Geometric Structure Extraction and Reconstruction

Inauguraldissertation
der Philosophisch-naturwissenschaftlichen Fakultät
der Universität Bern

vorgelegt von

Shihao Wu

von China

Leiter der Arbeit:

Prof. Dr. M. Zwicker
Universität Bern

Prof. Dr. M. Wand
Universität Mainz

Geometric Structure Extraction and Reconstruction

Inauguraldissertation
der Philosophisch-naturwissenschaftlichen Fakultät
der Universität Bern

vorgelegt von

Shihao Wu

von China

Leiter der Arbeit:

Prof. Dr. M. Zwicker
Universität Bern

Prof. Dr. M. Wand
Universität Mainz

Von der Philosophisch-naturwissenschaftlichen Fakultät
angenommen.

Bern, 6.7.2018 **Der Dekan:**
Prof. Dr. G. Colangelo

Abstract

Geometric structure extraction and reconstruction is a long-standing problem in research communities including computer graphics, computer vision, and machine learning. Within different communities, it can be interpreted as different subproblems such as skeleton extraction from the point cloud, surface reconstruction from multi-view images, or manifold learning from high dimensional data. All these subproblems are building blocks of many modern applications, such as scene reconstruction for AR/VR, object recognition for robotic vision and structural analysis for big data. Despite its importance, the extraction and reconstruction of a geometric structure from real-world data are ill-posed, where the main challenges lie in the incompleteness, noise, and inconsistency of the raw input data. To address these challenges, three studies are conducted in this thesis: i) a new point set representation for shape completion, ii) a structure-aware data consolidation method, and iii) a data-driven deep learning technique for multi-view consistency. In addition to theoretical contributions, the algorithms we proposed significantly improve the performance of several state-of-the-art geometric structure extraction and reconstruction approaches, validated by extensive experimental results.

Keywords. Points representation, meso-skeleton, consolidation, data consolidation, filtering, clustering, dimensionality reduction, manifold denoising, generative adversarial network, multi-view reconstruction, multi-view coherence, specular-to-diffuse, image translation

ii

Acknowledgements

First and foremost I want to thank my advisor Prof. Matthias Zwicker. It has been a great honor and pleasure working with him. I'm thankful for all his contribution of time, care, effort, ideas, joy, motivation, teaching, and funding that give me the best Ph.D. experience one could ask for. None of my research work could have been possible without his intellectual guidance and mental support. He is such an excellent role model of life-work balance, creative, curiosity, optimistic, generous, and kindness that teaches me how to become a happier, healthier, and more conscious person, which makes this journal truly invaluable.

I'm also grateful for working in Bern with those incredible CGG members. Dragana Esser gave me the warmest welcoming and help me settle down in Bern thoughtfully. She keeps our lab well organized and makes everything easy going. She is always there and walks me through many life troubles. Marco Manzi is friendly, productive, and full of interesting ideas. It's always pleasant being with and learning something from him, e.g. the time we spent together in Siggraph Asia still fresh to me. Daniel Donatsch is cheerful, dependable, and sporting, kind-hearted. He always offers his help before I ask. He is also my coach for indoor cycling and certainly convince me the benefits of doing sport. Peter Bertholet is so talented, humble, and being patient as my teacher of mathematics and geometry. He contributes to the elegant formulation for our theoretical analysis in the SAF work. We had a wonderful time attending conferences, hiking, and board gaming. Tiziano Portenier is quiet, efficient and cooperative. He teaches me a lot, especially the deep learning techniques. He wrote a professional technical description of S2Dnet shortly before the deadline. Daljit Singh is a close friend and his passion for life and research is contagious. Siavash A. Bigdeli has many good qualities I'm learning from, e.g., confidence, curiosity, and heartiness.

I would also like to offer my sincere thanks to my mentors over the years. Prof. Hui Huang is a co-supervisor of both my master and Ph.D. degree. She is amiable, determined, aspiring, being the backbone of

our projects, and keep encouraging and cultivating me to be a good researcher. Prof. Guiqing Li is my master supervisor who opened the door for me doing research. He is tireless, unselfish and popular to the students. We keep collaborating and published two works during my Ph.D. study. Prof. Daniel Cohen-Or is the top of the top researcher in the field, who inspires most of the ideas of my research work and keep lightening and driving the project to success with his dedication. He invited me to visit his lab in the table-via university and his hearty families, which is an unforgettable experience. Prof. Minglun Gong is brilliant, caring and skillful. The soul of my research works would have been incomplete without his great contributions.

I would like to take this opportunity to thank all the collaborators in my research works, including: Matan Sela, Xuequan Lu, Honghua Chen, Yinglong Zheng, Jie Mei, Liqiang Zhang, Wei Sun, Pinxin Long, Jiacheng Ren, Jinfeng Ou, Xiaohui Zhou, Simone Raimondi, Prof. Ron Kimmel, Prof. Oliver Deussen, Prof. Baoquan Chen, Prof. Uri Ascher, Prof. Hao Zhang, and etc..

We are also lucky of meeting great friends in these four years who gave us lots of help and joy: Niclas Scheuing, John Sivell, Vittorio Caggiano, Sukhbeer Kaur Sabharwal, Laura Höylä, Anita Sobe, Céline Dizerens, Rava Pourboghrat, Peppo Brambilla, Gian Calgeer, Stefan Moser, Fabrice Rousselle, Givi Meishvili, Simon Jenni, Adrian Wälchli, Qiyang Hu, Meiguang Jin, Zan Li, and etc.

The lists of people who helped me in the journey have no ends. In particular, I can't thanks enough to Prof. Michael Wand for being the co-referee of my thesis and Prof. Paolo Favaro being the head of defense community. Their generous support during my thesis writing period makes this work finish in time.

Last but not least, I thank my parents Qinghua Wu and Huaying Luo for their unconditional love. They offer me absolute freedom and support to pursue my goals. My heartiest gratitude goes to my beloved wife Xiaowei (Magenta) Zeng. Her smile and bravery make us overcome every day's challenges effortlessly. Her kindness and friendliness bring us the happiness and friends. Her artistic taste deeply flows into my works. Her love makes every impossible possible.

I thank everyone for everything and devote this thesis.

Contents

List of Figures

3

Chapter 1

Introduction

> Geometry existed before the creation.
>
> *Plato*

Geometric structure is beautiful and everywhere, see Figure 1.1 for a few examples in our nature. As one of the earliest forms of human art, our ancestors engraved geometric pattern on stones, see Figure 1.2, around 100,000 years ago. Our modern data acquisition techniques such as CT scanner, light field camera, and 3D laser scanner, as shown in Figure 1.3, allow us capturing massive real-world geometric data in a few seconds. However, given these raw input data, the extraction and reconstruction of the geometric structure remains a long-standing problem shared by research communities including computer graphics, computer vision, and machine learning. Researchers in different fields may hold a different view for this problem depending on what types of input data they are working with. For example, it can be viewed as a problem of skeleton extraction from the point cloud [130], surface reconstruction from multi-view images [118], and manifold learning from high dimensional data [19]. All these subproblems have gained much attention in recent years, driven by a wide range of modern applications, such

9

as scene reconstruction for AR/VR, object understanding for robotic vision, and structural analysis for big data, as shown in Figure 1.4. It is also tightly related to those big questions in artificial intelligence: how would a machine be truly aware of the geometric structure with their own sensing? Can it extract, reconstruct and understand the underlying structure of the information consciously, creatively, and deeply? What governs the structure of all? Our exploration of such a fundamental problem is an exciting and everlasting journey.

Figure 1.1: Examples of geometric structure in the natural. From top left to bottom right: a nice curve of our milki-way, a plant with spiral structure, an omega shape formed by birds, and the tree structure of our brain. Credit: WALLPAPERIA, Alanna, Alain, Pinterest.

Figure 1.2: A piece of red ochre with a deliberate zigzag engraving from Blombos cave, South Africa. Credit: Anna Zieminski/AFP/Getty and Chris Henshilwood.

Figure 1.3: Some modern data acquisition techniques. From left to right are: medical scanner, light field camera, 3D scanner. Credit: RAYMOND R. HOULE CONSTRUCTION, LYTRO, and [155].

1.1 Problem Statement

Let us first define the scope of the problem addressed in this thesis. We assume the raw input data to our problem are commonly used media like 2D images, 3D point cloud, or high dimensional scientific data. The output geometric structure we seek for is a continuous, intrinsic, and meaningful representation of the input data. The structure representation can be in the form of a skeleton, surface, manifold, etc. Ideally, the underlying geometric structure of the data can be best extracted and reconstructed when the input data is intact, clean and consistent. However, the reality is quite the opposite. All kinds of prac-

Figure 1.4: Applications of geometric structure extraction and reconstruction: AR/VR, autonomous car, data visualization and analysis. Credit: Sergey Nivens, OVERSQUARE AUTOMOTIVE, Kimo Quaintance.

tical and physical limitations make the input data incomplete, noisy, and inconsistent. Hence our research problem of geometric structure extraction and reconstruction from the imperfect raw input data is ill-posed. Knowing that a perfect solution to such ill-conditioned problem does not exist, we can discuss in the following how we approach these three challenges, i.e., the incomplete, noisy, and inconsistent nature of the data.

Incompleteness.   Almost everything in our universe, including a molecule, a stone, a tree, a forest and even a distribution of some high dimensional data, has its unique structure and contains a large amount of geometric information. Capturing and storing all these information is an ambitious yet unpractical goal, despite the fast development of advanced data acquisition techniques [155]. Besides, one common limitation of all data collection approaches is that the cost of the data acquisition goes up quickly as the increase of data completeness. For example, while it cost almost nothing to take a 2D selfie image, fully scanning a 3D surface or volume of the human body can be expensive and time-consuming. A sacrifice of completeness often has to be made for a low price of data. Therefore, it is desirable to complete the missing geometric information for those incomplete data, based on reasonable prior knowledge [11] such as symmetry,

skeleton, smoothness, correspondence, topology, functionality, etc. Different priors favor different types of input data.

We focus our study of the data completion problem on 3D point clouds, which are acquired from devices like laser scanners and depth cameras that are widespread. A raw point cloud data is a set of data points in space that represents the external surfaces of the object. As shown in Figure 1.5 (a), the point cloud is often incomplete, as a result of self or inter-object occlusion, inapplicable material, misalignment, unreachable viewpoints, or the acquisition of time-varying data from a single view [142]. Such data imperfection has greatly limited the application of this innovative data representation for many years and remains an open problem in research. As one direction approaching this problem, we proposed a new point set representation method in **Chapter 3** that allows a joint optimization of the skeleton extraction and surface completion of an object.

Noise and outliers. Like the data incompleteness, noise and outliers are well-known data imperfections that are caused by inevitable limitations of data observation and collection approaches, as shown in Figure 1.5 (b). Given such noisy and outlier-ridden input data, scientists in statics, signal processing, and machine learning have spent tremendous time and effort in designing robust denoising algorithms that reveal and preserve the intrinsic structure of the data. The key to a successful algorithm is an insightful observation of the input data. Therefore, different denoising algorithms have been developed for different types of media such as audio, images, and point clouds. We focus our study on one kind of denoising method that works particularly well for point-based data representation. This method, named Locally Optimal Projection (LOP), is proposed by Lipman et al. [85] in 2007 for point cloud denoising. The core insight of LOP is a regularization term for uniform points distribution that maintains the structure of data while denoising. We introduce the basic of LOP in **Chapter 2**. In **Chapter 4**, we made an important theoretical analysis of LOP about the convergence criterion, revealing its relation to classical algorithms like mean shift and manifold denoising, and demonstrating its power

in applications like clustering and dimensionality reduction.

Inconsistency. Caused also by the limitations of existing data acquisition techniques, our input data is usually captured by multiple observations of the same target but from different views. Given such multi-view data, the extraction of data correspondence and the reconstruction of the target model are difficult, if not impossible. The challenge is twofold. First, the information of the observer, e.g. the camera positions, are unknown or unreliable. Second, irrelevant environmental factors that affect the outcome of the observation form different views, e.g. illumination and material, are also unknown and complex.

While these two issues are common for many types of data, the data inconsistency challenge in our study is specifically referred to the appearance incoherency in 3D reconstruction from multi-view images, as shown in Figure 1.5 (c). One rather strong assumption shared by many multi-view 3D reconstructions is that the target objects are predominantly diffuse. This assumption suggests that the appearance of the model are mostly consistent across different views, which is important for the computation of image correspondences through feature matching and the improvement of the reconstruction quality through shape-from-shading [79]. However, when the object is not diffuse, the appearance of the object changes in different views and the shading cue is not usable without additional prior knowledge like global illumination or shape silhouette [42]. Therefore, the multi-view reconstruction of glossy surfaces is a challenging problem. We believe a promising way to approach this problem is data-driven machine learning, because of the fact that our human mind can be trained to recognize the geometric structures in the images regardless of the complexity of the scene. Encouraged by the success of deep learning algorithms in image processing, we addressed this problem in **Chapter 5** through a generative adversarial network (GAN) for multi-view specular-to-diffuse image translation, where some basic of GAN is covered in **Chapter 2**.

(a) Incompleteness   (b) Noise and outliers   (c) Inconsistency
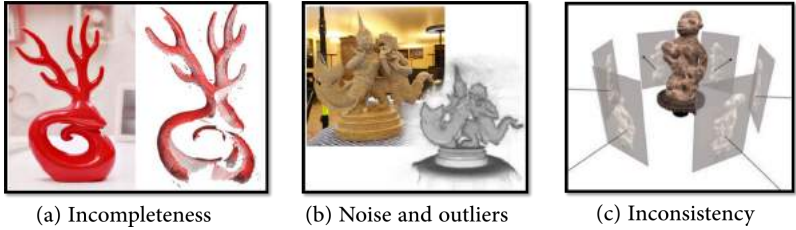
Figure 1.5: Main challenges of geometric structure extraction and reconstruction: incompleteness, noise and outliers, and inconsistency. Credit: [58], [151], and [36].

Goals. Aiming to advance research in dealing with the aforementioned main challenges, in this thesis, the goal we pursue in this thesis is to explore optimization and learning techniques to extract and reconstruct the underlying geometric structure from raw input such as point cloud, multi-view images, and high-dimensional data. Towards this goal, we first propose a new 3D point sets consolidation method that optimizes both the inner and surface structure simultaneously. Next, we extend our consolidation technique to high dimensional data with a theoretical analysis for parameter selection. Finally, we tackle the appearance inconsistency problem of multi-view images by leveraging the recent success of learning based image-to-image translation approaches. Experimental results show that all of our proposed methods outperform the corresponding state-of-the-art methods, especially in terms of dealing with noise, incompleteness, inconsistency, and outliers of the data.

1.2 Contributions

This thesis contributes in the following three problems:

Deep Points Consolidation [154]. We present a consolidation method that is based on a new representation of 3D point sets. The key idea

is to augment each surface point into a *deep point* by associating it with an inner point that resides on the meso-skeleton, which consists of a mixture of skeletal curves and sheets. The deep points representation is a result of a joint optimization applied to both ends of the deep points. The optimization objective is to fairly distribute the end points across the surface and the meso-skeleton, such that the deep point orientations agree with the surface normals. The optimization converges where the inner points form a coherent meso-skeleton, and the surface points are consolidated with the missing regions completed. The strength of this new representation stems from the fact that it is comprised of both local and non-local geometric information. We demonstrate the advantages of the deep points consolidation technique by employing it to consolidate and complete noisy point-sampled geometry with large missing parts.

Structure-aware Data Consolidation [153]. We present a structure-aware technique to consolidate noisy data, which we use as a pre-process for standard clustering and dimensionality reduction. Our technique is related to mean shift, but instead of seeking density modes, it reveals and consolidates continuous high density structures such as curves and surface sheets in the underlying data while ignoring noise and outliers. We provide a theoretical analysis under a Gaussian noise model, and show that our approach significantly improves the performance of many non-linear dimensionality reduction and clustering algorithms in challenging scenarios.

Specular-to-Diffuse Translation for Multi-View Reconstruction.
Most multi-view 3D reconstruction algorithms, especially when shape-from-shading cues are used, assume that object appearance is predominantly diffuse. To alleviate this restriction, we introduce S2Dnet, a generative adversarial network for transferring multiple views of objects with specular reflection into diffuse ones, so that multi-view reconstruction methods can be applied more effectively. Our network extends unsupervised image-to-image translation to multi-view "specular to diffuse" translation. To preserve object appearance across

multiple views, we introduce a Multi-View Coherence loss (MVC) that evaluates the similarity and faithfulness of local patches after the view-transformation. Our MVC loss ensures that the similarity of local correspondences among multi-view images is preserved under the image-to-image translation. As a result, our network yields significantly better results than several single-view baseline techniques. In addition, we carefully design and generate a large synthetic training data set using physically-based rendering. During testing, our network takes only the raw glossy images as input, without extra information such as segmentation masks or lighting estimation. Results demonstrate that multi-view reconstruction can be significantly improved using the images filtered by our network. We also show promising performance on real world training and testing data.

1.3   Thesis Organization

In **Chapter 2**, two technical backgrounds are introduced, one is LOP for local structure filtering, the other is GAN for global structure learning.

In **Chapter 3** we introduce the *deep point* representation of 3D point sets. We augment each surface point into a *deep point* by associating it with an inner point that resides on the meso-skeleton, which consists of a mixture of skeletal curves and sheets.

Chapter 4 present a structure-aware technique to consolidate noisy data, which we use as a pre-process for standard clustering and dimensionality reduction.

In **Chapter 5** we introduce S2Dnet, a generative adversarial network for transferring multiple views of objects with specular reflection into diffuse ones, so that multi-view reconstruction methods can be applied more effectively.

Chapter 6 concludes our research and results and briefly describes possible future work.

Chapter 2

Technical Background

2.1 Introduction

For a better understanding of the techniques in the later chapters, in this chapter, we briefly introduce two important background techniques: i) Locally Optimal Projection (LOP) for consolidation of low-level structure and ii) Generative Adversarial Network (GAN) for learning high-level structure.

LOP is the base of **Chapter 3** and **Chapter 4**, which was originally introduced by Lipman et al. [85] and improved by Huang et al. [57]. Please refer to these two works for a more detailed technical description of LOP. GAN is a trending learning method that **Chapter 5** is based on. It was introduced by Ian Goodfellow et al. [45] in 2014 and has attracted enormous attention in the past few years. For a more in-depth introduction of GAN, the reader is advised for other literature [44, 27, 54].

2.2 Locally Optimal Projection

The Locally Optimal Projection operator (LOP) was originally designed for surface approximation from point-set data. Given a set of original input points $Q = \{q_j\}_{j \in J} \subset \mathbb{R}^3$ that is typically unorganized, unoriented, unevenly distributed, noisy and outliers-ridden, the output of LOP should be a set of clean sampled points $X = \{x_i\}_{i \in I} \subset \mathbb{R}^3$, which in general is much smaller, i.e., $|I| << |J|$, and more organized than Q, representing the underlying surface of Q. Formulated as an optimization problem, the energy function of LOP is a combination of data term and repulsion term. The data term is a localized version of the L_1-median that robustly projects sample points onto the surface defined by the original points. Since the contraction force of data term tends to attract sampled points to the local density extrema, a repulsion term is introduced as a regularization of the points distribution by penalizing sampled points being too close to each other.

L_1-**median.**   The L_1-median is a simple and powerful statistical tool that can be regarded as the extension of the univariate median to the multivariate setting [150]. Unlike the usual mean average, L_1-medians is known to be robust to noise and outliers, as shown in Figure 2.1. By definition, the total Euclidean distance between the L_1-median location x and a set of input points is minimized, i.e.,

$$x = argmin \sum_{j \in J} \|x - q_j\|.$$

Localized L_1-median as data term.   The L_1-median is a global center of a set of points. It can be adapted locally to a point by adding a weight function θ, i.e.,

$$x = argmin \sum_{j \in J} \|x - q_j\| \theta(\|x - q_j\|).$$
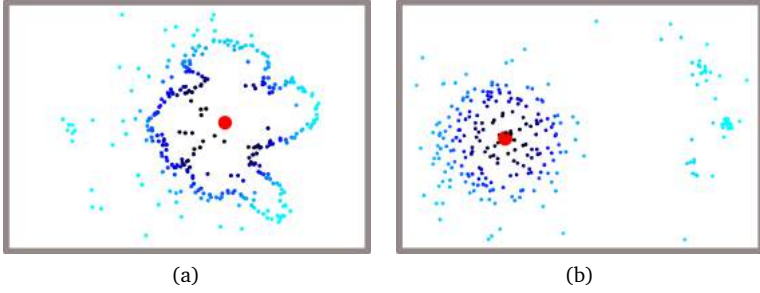
Figure 2.1: Insensitivity of the L_1-median (red dot) to noise and outliers in the data. The bluish shade of the points reflects the relative weight with respect to the L_1-median center.

The weight function $\theta(r) = e^{-r^2/(h/2)^2}$ is a fast decaying smooth function with support radius h defining the size of the supporting local neighborhood. The result of localized L_1-median depends on the initial position of x and the neighorhood size h. As shown in Figure 2.2, a single localized L_1-median can not represent a complex shape no matter what h we use. Therefore, we are seeking for a set of localized L_1-medians that best approximates the target shape as our data term, i.e.,

$$\arg\min_X \sum_{i \in I} \sum_{j \in J} \|x_i - q_j\| \theta(\|x_i - q_j\|).$$

Repulsion term. Applying the L_1-median alone tends to yield a sparse distribution, where local centers are likely accumulated into a set of point clusters; see Figure 2.3(a). To avoid such clustering and maintain a proper geometry representation, we need to counteract the attraction force by adding a repulsion force on the structure itself. This brings the final objective function of LOP with two terms:

$$h = h_0 \qquad\qquad h = 2h_0 \qquad\qquad h = 4h_0$$
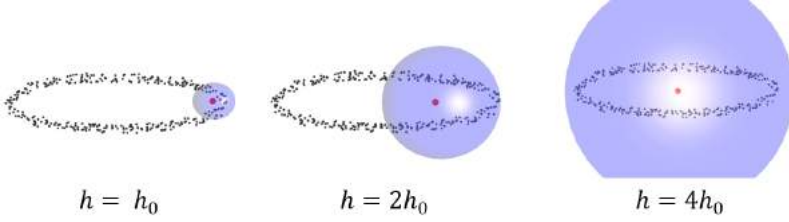
Figure 2.2: Localized L_1-Median using different size of local neighborhood.

$$\arg\min_X \sum_{i \in I} \sum_{j \in J} \|x_i - q_j\| \theta(\|x_i - q_j\|) - \sum_{i \in I} \mu_i \sum_{i' \in I \setminus \{i\}} \theta(\|x_i - x_{i'}\|),$$

where the $\{\mu_i\}_{i \in I}$ are balancing constants among X that can be set by the user. See Figure 2.3(b) for the effect of using the replusion term.

(a) Without repulsion   (b) With repulsion
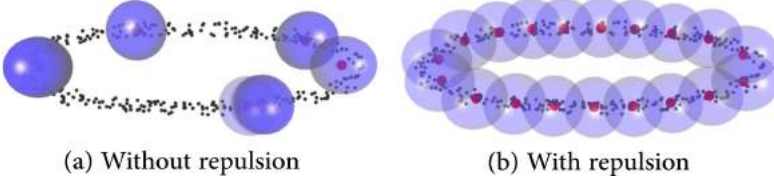
Figure 2.3: With and without repulsion term.

Pros and cons of LOP.   Besides its robustness to noise and outliers, the main advantage of the LOP operator is that it work directly on raw point clouds without additional information like mesh connectivity and point normal. The repulsion term allows LOP extract multi-scale structures with nice distribution by using different kernel sizes, as

demonstrated in Figure 2.4. Some disadvantages of LOP include over-smoothing, failing to deal with large missing data and sensitivity to the selection of parameters. We have proposed some methods that improve LOP to better preserve sharp features [59, 88, 166, 167]. In **Chapter 3**, we address the shape completion problem based on LOP. In terms of parameter selection, Lipman et al. [85] have proven that if the data set Q is sampled from a C^2-smooth surface S, LOP operator has an $O(h^2)$ approximation order to S, provided that the balancing parameters $\{\mu_i\}_{i\in I}$ are carefully chosen. We further analysis the relation between $\{\mu_i\}_{i\in I}$ and neighborhood size h in **Chapter 4**.

$$h = h_0 \qquad h = 2h_0 \qquad h = 4h_0$$
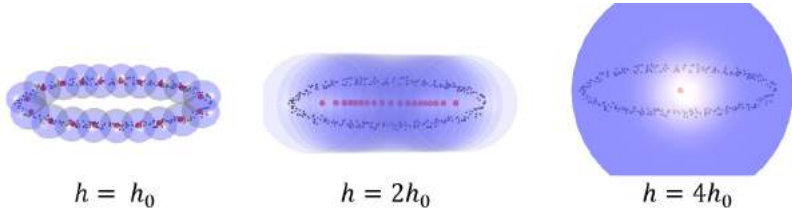
Figure 2.4: LOP using different local neighborhood sizes.

2.3 Generative Adversarial Network

In this section, we will first walk through some basic concepts of neural networks. Then, we focus on the theoretical and practical background of Generative Adversarial Networks. Last, we introduce some extension of GAN that serve as the baselines of **Chapter 4**.

Neural network. In 1943, Warren McCulloch and Walter Pitts [92] created a computational model for neural networks based on mathematics and algorithms called threshold logic. Nowadays, the neural network has become the most powerful machine learning tool in many complex tasks in areas like natural language processing and computer
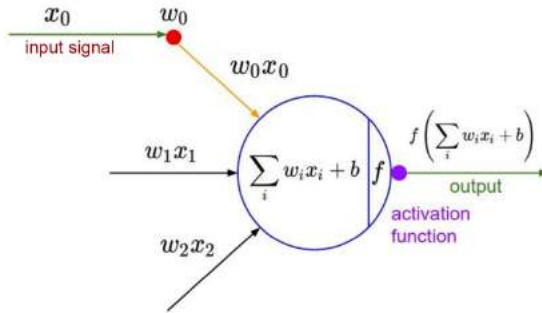
Figure 2.5: Illustration of a neuron activation with three input signals. Credit: Karpathy, CS231n, Standford, 2016.

vision, thanks to a) the great development of GPU computation power; b) the availability of huge amounts of training data, and c) some smart design of network architecture.

A neural network can be seen as a black box that transfers input data X into a desirable output Y, i.e., $F(X) = Y$. Such black box is similar to a biological brain of an animal or human being. From the viewpoint of neuroscience, we can look at this black box from both outside and inside. Looking from outside, we can see the brain process the input information X and give a response output Y. The response is based on the experience learned in the past and some projection for the future. Looking from inside, we can see the millions of neurons transport information through some chemical and physical activations. However, no matter from outside or inside, we do not know yet the mechanism how a thought is actually generated in our brain. It remains an on-going research in both neuroscience and computer science to form a theory for the explanation of the behavior of the neural network. Nevertheless, given enough training data, a neural network is a great tool in approximating a highly non-linear function that maps X to Y, e.g. determining the label of an image. Interestingly, such complex functions are built upon thousands of millions of base computational unit, called neuron.

Activation function. Given n input signals $x_0, x_1, ..., x_n$, a neuron is defined by a set of weights $w_0, w_1, ..., w_n$, where the activation a of a neuron can be computed as

$$a = \sum_{i=1}^{n} x_i \cdot w_i.$$

To introduce non-linearity, we apply a non-linear function called activation function f and a bias b, and get our output signal

$$y = f\left(\sum_{i=1}^{n} x_i \cdot w_i + b \right).$$

Here the weights are the parameters to be learned during training, as illustrated in Figure 2.5. There are various choices for f. For example, a conventionally used sigmoid function,

$$f(z) = 1/(1 + e^{-z}),$$

or a TanH function,

$$f(z) = (e^z - e^{-z})/(e^z + e^{-z}),$$

or rectified linear unit (ReLU) that is more frequently used because of its fast convergence and unlike TanH, ReLU dose not have the gradient vanishing issue, i.e.,

$$f(z) = max(0, z),$$

and its variant, leaky ReLU, that tries to avoid the dead activation issue, i.e.,

$$f(x) = \begin{cases} x & \text{if x} > 0 \\ \alpha x & \text{otherwise} \end{cases}.$$
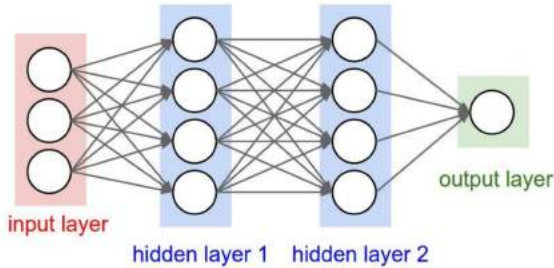
Figure 2.6: An example of a neural network structure. It consists of an input layer with three features, two hidden layer with 4 neurons of each, and one output layers. These are fully-connected in the sense that, for every layer, each neuron is connected to all neurons of the next layer. Credit: Karpathy, CS231n, Standford, 2016.

Building a network.   A neural network is created by connecting neurons in a certain way. Figure 2.6 show an example of building a fully-connected network upon some neurons. This is done by organizing neurons into several layers and establishing connections between consecutive layers, forming a multilayer perceptron. By connection, it means that the output value of a neuron is taken as the input signal of a neuron in the next layer.

As discussed, a neural network, defined by a collection of weights $\theta = w_0, w_1, ..., w_m$ of all neurons in the network, can be viewed as a complex function $F(x_i; \theta) = y_i$, where x_i is a sample on the feature domain, and y_i is the corresponding output of the network. The goal of training a network is to seek a set of optimal weights θ^*, such that the behavior of $F(x_i; \theta^*) = y_i$ meets our expectation.

To describe the expectation of our problem, we need at least $|K|$ sample data $x_1, e_1...x_k, e_k$ for computing our parameters, where $|K| > 1$ and e_i is the expected output of x_i. For example, x_i can be an RGB image and e_i can be an annotated label (e.g. whether it contains a cat) of that image x_i. A naive way to search the optimal parameter is by exhaustively testing out all possible combination of θ, which is
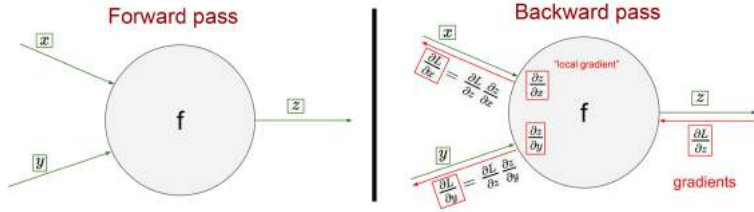
Figure 2.7: A visualization of the forward and backward passes of backpropagation. Credit: CS231n, Standford, 2018.

unfeasible and usually there exist more than one solution.

To make this problem more trackable, like most machine learning algorithms do, training a network is often done by optimization. First, we need to formulate the expectation of our task as an objective (or loss) function that can be minimized/maximized. For example, the loss can be formulated as the sum of the Euclidian distances between y_i and e_i, i.e.,

$$\mathcal{L} = \sum_{i=1}^{n} \|y_i - e_i\|.$$

When $\mathcal{L}$ is minimized, an "optimal" set of parameters θ^* is obtained, and hopefully for a new input x^{new}, the output $y^{new} = F(x^{new}; \theta^*)$ meets our expectation.

Although minimizing the loss $\mathcal{L}$ cannot guarantee a perfect behavior of the trained network on new data, this approach is computationally trackable, and often produces a satisfying result. Note that giving the same loss function, our transformation function $F(x_i; \theta)$ can also be defined by other forms like logistic regression [56] or support vector machines (SVM) [51]. In this thesis, we focus on the formulation using neural networks, which in general have a much bigger capacity in representing a complex transformation function, but requires a large amount of training data and a careful optimization to avoid overfitting.

Training a network.   Now that the structure and objective function $\mathcal{L}$ of the network is defined, we can minimize our objective function by optimizing the parameters θ using a well-studied optimization method called gradient descent. Gradient descent is known as a powerful first-order iterative optimization algorithm for finding the minimum of a differentiable objective function. In general, the gradient describes the slope of a function with respect to a variable. The slope always points to the nearest valley of the objective function. Given a function defined by a set of parameters, gradient descent starts with an initial set of parameter values and iteratively moves toward a set of parameter values that minimize the function, taking steps in the negative direction of the function gradient. For example, the gradient

$$\frac{\partial \mathcal{L}}{\partial \theta} = \left(\frac{\partial \mathcal{L}}{\partial w_0}, \frac{\partial \mathcal{L}}{\partial w_1}, ..., \frac{\partial \mathcal{L}}{\partial w_m}, \right)$$

describes the slope of the loss function with respect to the network parameters θ^0. Given a set of network parameters θ, the weights are iteratively updated by

$$\theta = \theta - \lambda \frac{\partial \mathcal{L}}{\partial \theta}$$

where the λ is the learning rate that controls the step size of θ moving along the negative gradient direction. This process is performed continuously until we reach a local minimal of $\mathcal{L}$.

Now we need to compute the partial derivative. Surprisingly, this can be readily done for an arbitrarily complex network using a simple technique called backpropagation, which is a way of computing gradients of expressions through recursive application of the chain rule. It consists of two stages: a forward and a backward pass, as visualized in Figure 2.7. In the feed-forward stage, the input features are processed from the first to the last layer of the network according to the pre-defined connections of neurons as described above. In this stage, the local gradient at each node can be computed and stored for later usage. The output of this stage is used to compute the error value of loss $\mathcal{L}$. Subsequently, the network is run backward, backprop-
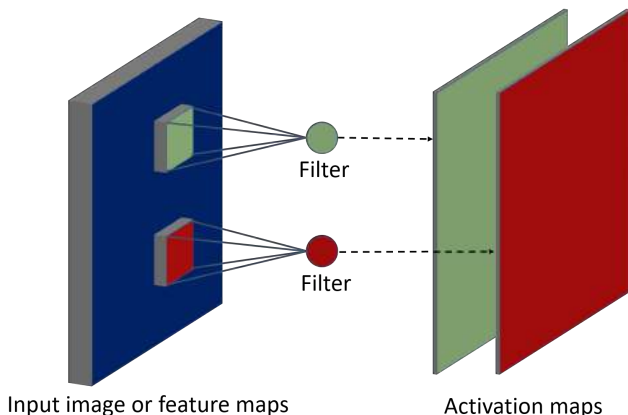
Figure 2.8: A visualization of a convolutional layer. Two kernels are applied on the entire input images and extracted two feature maps independently. The weights of each kernel are shared by the corresponding activation map.

agating the error of each neuron from the last to the first layer. Using the stored derivatives, each neuron only needs to take care of the local gradient coming from the direct upstream, applying the chain rule, and passing the modified gradient to its downstream. In the end, we get the $\frac{\partial \mathcal{L}}{\partial \theta}$ in the first layer and use it to update the network parameters.

Many advanced optimization techniques have been developed to improve the training process. For example, stochastic gradient descent is used to consider a mini-batch instead of all training examples at once. Adam optimizer [74] can adaptively estimate the learning rate based on first and second order momentum. Batch normalization [64] or its variants are widely used for stabilizing the learning. Regularization techniques such as L2 & L1 regularization, Dropout, early stopping, and data augmentation are frequently used in practice to cope with overfitting.
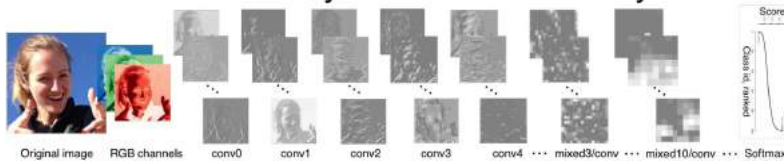
Figure 2.9: A visualization of the feature maps in different layers of a convolutional neural network. Credit: CS231n, Standford, 2018.

Convolutional Neural Networks   Applying the aforementioned network architecture for complex input features like RGB images can be problematic. It will consume lots of memory and give too many degrees of freedom. Therefore, LeCunn et al. [80] proposed a new kind of architecture that takes the semantic relations over a neighborhood into account and constrains the leaning problem by weights sharing. Such network architecture is generally refer to convolutional neural networks and its most distinguishing feature is the convolutional layer, as shown in Figure 2.8.

The input to a convolutional layer is an image or the feature maps from the previous layer. Unlike a fully-connected network that each activation must be connected to all features in the previous layer, in a convolutional layer, one activation map shares the weights of a filter. This is done by the convolution operation in image processing.

It works like this: a filter defined by a set of weights with a relatively small window size will slides through an image, marching with a certain step size called stride. In each step, the filter overlaps with a block of the input volume and a weighted average is computed for a pixel in the output activation map. As a result of this filtering operation, we connect each neuron to only a local region of the input volume and dramatically reduce the number of network parameters. The filter bank is invariant to location, since its weights are the same for the whole feature map. The spatial extent of this connectivity is a hyperparameter called the receptive field of the neuron, or equivalently the size of the filter.

The spatial size and depth of the output feature maps is deter-

mined by the window size, stride and number of the filters. In some early practice, a pooling layer is used after the convolutional layer, in order to reduce the spatial resolution quickly. But this can also be done by using a stride size > 1 in the convolutional layer. In general, the spatial size of the feature maps decrease as the layer number increases. This encourages the network to considers multi-scale information of the image, and further reduces the number of parameters. As shown in Figure 2.9, the beginning layers of the networks respond to the local edge information, while the later layers seem to capture some global information like illumination. For classification tasks, a fully-connected and softmax layer is usually used to generate a matching score for evaluating the loss.

Note that sometimes it is also desirable to increase the spatial resolution of the feature maps, e.g., in image super-resolution, segmentation, and autoencoders. A common technique to achieve that is a deconvolution layer (or transposed convolutional layer), proposed by Zeiler et al. [162]. The deconvolution can be then considered as an operation that allows recovering the input of a feature map that is originally obtained by applying a convolution on the input, as shown in Figure 2.10.

So far, the most influential design of convolutional neural networks include: LeNet [80], AlexNet [77], GoogleNet [128], VGGNet [127], and ResNet [50]. As the performance of the network appeared to be improved by increasing the number of convolutional layers, some people refer to this technique as deep learning.

Generative Adversarial Networks   The network architectures discussed above are mostly designed for supervised learning. The unsupervised learning problem is more challenging in machine learning. The input data to unsupervised learning have no labels, and we need to learn the underlying hidden structure of the data, e.g., clustering, dimensionality reduction, feature learning, and density estimation.

One architecture for learning generative models of data is the autoencoder, where the goal is to learn a representation (or encoding) for a set of data, typically for the purpose of dimensionality reduction.
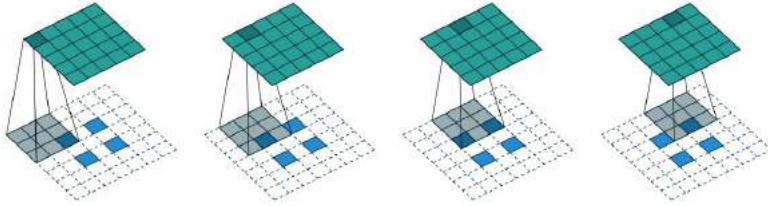
Figure 2.10: An example of deconvolutional layer. The transpose of convolving a 3×3 kernel over a 5×5 input using 2×2 strides (i.e., i = 5, k = 3, s = 2 and p = 0). It is equivalent to convolving a 3×3 kernel over a 2×2 input (with 1 zero inserted between inputs) padded with a 2×2 border of zeros using unit strides. Credit: [32].

An autoencoder network consists of two parts: an encoder to learn a compact latent space z for the abstract representation of the data, and a decoder to reconstruct the information from the compressed data, as illustrated in Figure 2.11. Once the training is converged, hopefully, new data can be generated by sampling on the latent space z. Variational autoencoder (VAE) [75] is theoretically sound and remains an active research field, but so far it tends to generate blurrier images compared to the GANs that we discuss next.

Generative adversarial networks (GANs) are deep neural net architectures comprised of two nets: i) a generator network G that tries to fool the discriminator by generating real-looking data, and ii), a discriminator network D that tries to distinguish between real and fake data, as shown in Figure 2.12. Ideally, the probability distribution $P_G(x)$ defined by generator G should be as close to the true data distribution $P_{data}(x)$ as possible and the discriminator D can serve as a measurement of the JS divergence (or similarity) between $P_G(x)$ and $P_{data}(x)$.

The training process of GAN can be viewed as a min-max game between generator G and discriminator D:
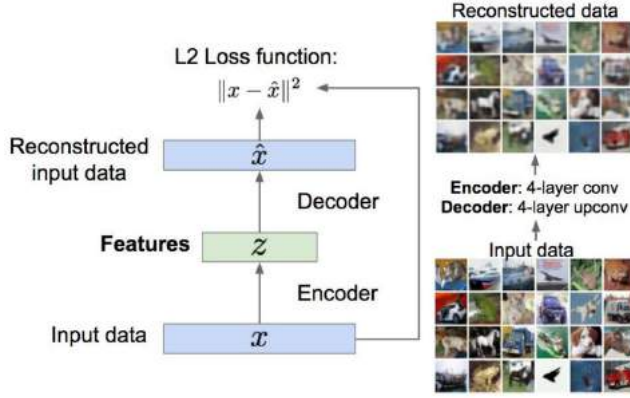
Figure 2.11: The structure of an autoencoder. The encoder network tries to find a compact latent space of data compression, while the decoder network aims at a full reconstruction of the original data. Credit: CS231n, Standford, 2018.

$$\min_{\theta_g} \max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p_z} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right],$$

where x is real data (randomly sampled from the training set) and z is random noise. The discriminator D defined by θ_d wants to maximize objective such that $D_{\theta_d}(x)$ is close to 1 (real) and $D_{\theta_d}(G_{\theta_g}(z))$ is close to 0 (fake). The generator G defined by θ_g wants to minimize the objective such that $D_{\theta_d}(G_{\theta_g}(z))$ is close to 1, i.e., the discriminator is fooled into thinking the generated $G_{\theta_g}(z)$ is real.

The training of GAN is performed by alternate between i) a gradient ascent on the discriminator

$$\max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p_z} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right],$$

and ii) a gradient ascent on the generator
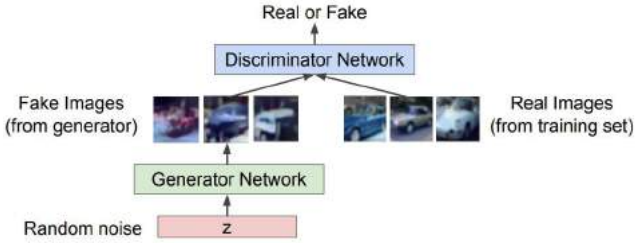
Figure 2.12: An overview of GAN. The generator tries to forge fake images from a random noise input, and the discriminator needs to judge whether an image is true or fake. Credit: CS231n, Standford, 2018.

$$\max_{\theta_g} \mathbb{E}_{z \sim p_z} \log(D_{\theta_d}(G_{\theta_g}(z))).$$

When training D, the parameters of G, i.e., θ_g, are fixed, and vice versa. In practice, some people find updating the discriminator k iterations before an update of the generator will lead to better results. But others find $k = 1$ works the best. Jointly training two networks simultaneously can be difficult and unstable. Recent work like DC-GAN [109], Wasserstein GAN [4], progressive GAN [69], and Spectral GAN [95] focus on improving the stability of the GAN training and the quality of the output images.

Overall, the output of GAN are generally crisper than those produced by other generative models like VAE, and the latent space learned by GAN is meaningful. One possible explanation is that GAN balances the pros and cons of generator and discriminator. Generator approaches the problem bottom-up, and it can easily generate complex data with a deep model, but it can not imitate the global appearance, and hard to learn the semantic correlation between components. On the other hand, the discriminator is top-down, being good at the big picture, but lacks the ability of generation.

Supervised and unsupervised conditional GAN. Besides image generation, GAN works surprisingly well in a visual task named image-to-image translation that transforms an image from domain X to Y. This is challenging because GANs needs to produce images that agree with the target domain Y, while the underlying structure in the o-riginal image should be preserved. Isola et al. [65] proposed using a conditional GAN to approach this problem and achieve stunning results. For the generator, the input is a combination of the source image and random noise. For the discriminator, the input image and the output of the generator are concatenated for examination. A conditional L1 regularization term is added to penalize big differences between the input and output image. Denoting the output image in domain B as y, the objective of a conditional GAN can be expressed as

$$\min_{\theta_g} \max_{\theta_d} \left\{ \mathbb{E}_{x \sim p_{data(x)}, y \sim p_{data}(y)} \left[\log D_{\theta_d}(x, y) \right] + \right.$$
$$\mathbb{E}_{z \sim p_z} \left[\log(1 - D_{\theta_d}(x, G_{\theta_g}(x, z))) \right] + \tag{2.1}$$
$$\left. \lambda \mathbb{E}_{x \sim p_{data}(x), y \sim p_{data}(y), z \sim p_z} \left[\| y - G_{\theta_g}(x, z) \|_1 \right] \right\},$$

where λ is a hyperparameter setting the weight of the regularization term.

Conditional GAN achieves great success in image translation but requires many paired images for training. Interestingly, people quickly find GAN can do the job almost as good without paired images, i.e., with unsupervised learning [169, 158, 73]. The core technique is a cycle-consistency loss that encourages the network to learn an image transformation from domain X to Y and then back to X. To do so, the GAN must somehow leans to keep the intrinsic information of the image during the transformation, e.g., from X to Y. Otherwise, the subsequent transformation from Y to X will violate the cycle-consistency.

The goal is to learn two mappings: $G_{X2Y} : X \rightarrow Y$ and $G_{Y2X} : Y \rightarrow X$, and two discriminator D_X and D_Y. For the mapping of

$G_{X2Y} : X \rightarrow Y$, the GAN loss can be expressed as:

$$\mathcal{L}_{GAN}(G_{X2Y}, D_Y, X, Y) = \mathbb{E}_{y \sim p_{data}(y)}\left[\log D_Y(y)\right] + \atop \mathbb{E}_{x \sim p_{data}(x)}\left[\log(1 - D_Y(x, G_{X2Y}(x)))\right]. \qquad (2.2)$$

Similar to the L1 conditional loss in equation (2.1), the cycle consistency loss can be written as:

$$\mathcal{L}_{cyc}(G_{X2Y}, G_{Y2X}) = \mathbb{E}_{x \sim p_{data}(x)}\left[\|G_{Y2X}(G_{X2Y}(x)) - x\|_1\right] + \atop \mathbb{E}_{y \sim p_{data}(y)}\left[\|G_{X2Y}(G_{Y2X}(y)) - y\|_1\right]. \qquad (2.3)$$

The final objective of cycleGAN is:

$$\mathcal{L}(G_{X2Y}, G_{Y2X}, D_X, D_Y) = \mathcal{L}_{GAN}(G_{X2Y}, D_Y, X, Y) + \atop \mathcal{L}_{GAN}(G_{Y2X}, D_X, Y, X) + \atop \lambda\mathcal{L}_{cyc}(G_{X2Y}, G_{Y2X}), \qquad (2.4)$$

where λ controls the relative importance of the two objectives.

In **Chapter 5**, we extend the GAN-based image-to-image translation methods from handing single view to multi-view images by introducing a multi-view coherence loss. Specifically, we perform specular-to-diffuse image translation and the 3D reconstruction based on the transformed diffuse image is significantly improved.

Chapter 3

Deep Points Consolidation

3.1   Introduction

Objects in computer graphics are commonly represented only by their surface. However, objects are typically volumetric and their analysis and processing should consider also their volume. To account for the volume, skeletal shape representations have been widely used for shape modeling, analysis and editing. Skeletal representations usually keep their linkage to the surface. One of the best known examples is the medial axis transform (MAT), which is the set of centers of tri-tangent spheres. Each point on the surface is then represented by a point and a radius on the MAT's skeleton.

   In this paper, we introduce a new representation for point sets that, similarly to the MAT, makes a link between the surface and its local volume. Each surface point is associated with an inner point that resides on a meso-skeleton [129], which consists of skeletal curves in cylindrical regions and skeletal sheets (i.e., medial axes) elsewhere. The augmented representation is a set of line sections, each with one end on the surface and the other on the meso-skeleton. We term these
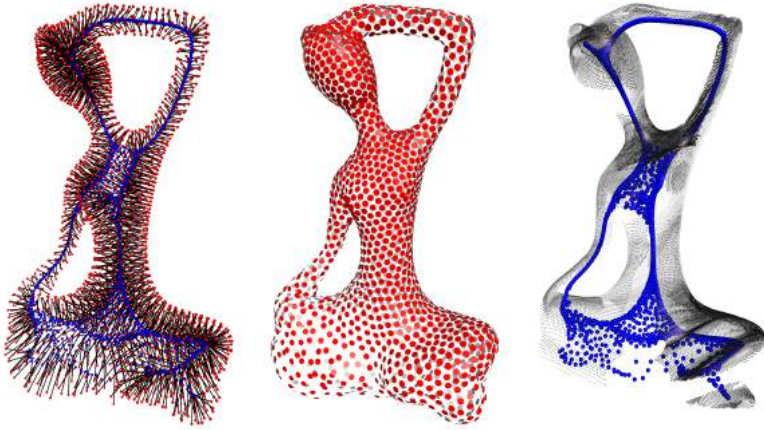
Figure 3.1: The deep points representation (left) is a set of line sections, each with one end (red) on the surface (middle) and the other (blue) on the meso-skeleton (right).

augmented points, *deep points*, or *dpoints* for short. See Figure 3.1 for an illustration of the deep points representation. The deep points form a smooth mapping between the surface and meso-skeleton, where their orientations agree with their corresponding surface normal directions. It is worth noting that unlike the MAT, dpoints can be computed robustly from noisy and highly incomplete input.

The deep points representation is a result of an optimization applied to the raw input point set. The key idea is to jointly optimize the surface and skeletal points so they form a valid deep points set. As we will show, the optimization converges where the inner points form a meso-skeleton, and the surface points are consolidated. The strength of the dpoints representation stems from the fact that it is comprised of both local and non-local geometric information. We demonstrate the advantages of dpoints by employing them to consolidate and complete noisy point clouds with large missing parts.

Surface normal vectors have a critical role in surface reconstruc-

tion. Advanced consolidation techniques [46, 102, 165, 59, 15] that can deal well with artifacts such as noise, outliers, irregular sampling, and sharp features rely on the availability of accurate normals. However, vector normals as a second order differential feature remain noisy, especially near open boundaries. Thus, unreliable normals make it challenging to complete missing surface data in the proximity of boundary points. Our consolidation technique can complete the surface without assuming the availability of accurate surface normals.

Figure 3.2 illustrates the main benefits of surface reconstruction using our dpoints representation. The raw scan in Figure 3.2(a) is non-uniform, many regions are sparse, and large parts are completely missing. Consequently, boundaries are not clearly defined and normal data is unreliable. Our dpoints representation computes a topologically correct meso-skeleton for the input shape, which provides non-local geometric information and guides the completion of missing regions on the surface. The result of our consolidated point set surface is shown in Figure 3.2(e), to which we apply Poisson reconstruction in Figure 3.2(f); see also the accompanying video.

In contrast, directly applying state-of-the-art reconstruction methods to the noisy and incomplete input, such as Poisson reconstruction [71] in Figure 3.2(b), does not provide plausible results. The WLOP technique [85, 57] excels in that it can consolidate raw and imperfect data without relying on normals. However, WLOP does not complete missing regions as is obvious in Figures 3.2(c-d).

It should be noted that surface completion is, by its nature, an ill-posed problem. We therefore guide it by a coherent meso-skeleton, resulting in natural-looking reconstructions even for highly incomplete scans; see an evaluation in Figure 3.17. Furthermore, our completion is contextual as it extrapolates the existing surface, rather than completing it with context-oblivious data, like circular [132] or elliptical [58] cross sections; see comparisons in Figure 3.13.

The rest of this chapter is organized as follows: we discuss the background and previous work in Section 3.2.
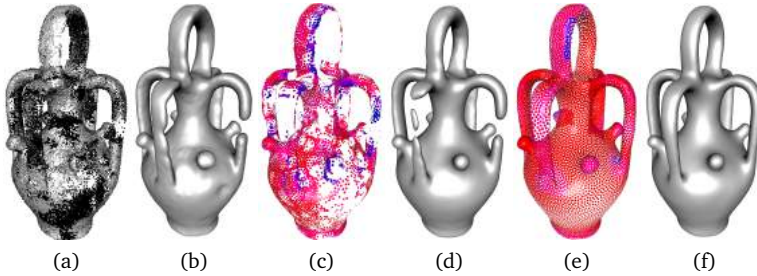
Figure 3.2: The input point cloud (a) contains noise and large missing regions. Applying Poisson surface reconstruction [71] on either the input (a) or the WLOP consolidation [57] result (c) does not yield satisfactory models; see (b) and (d), respectively. The surface points shown in (e) are consolidated and completed by our dpoints technique. This leads to a much better Poisson surface reconstruction (f). In (c) and (e), the errors of the surface point normals estimated by local PCA are evaluated based on the ground truth and color coded (blue means higher error).

3.2 Related work

Surface reconstruction. In a broader context, our work is related to the vast literature on surface reconstruction [55, 139, 3]. Noise is a major challenge in handling real scanned data. Assuming local smoothness, methods based on signed distance functions (SDF) [18, 70, 71] can reconstruct watertight surfaces. These techniques assume the whole model is scanned and when missing data is significant the reconstructed surfaces are often overly smooth and may contain topological errors. Please refer to Berger et al. [10, 12] for a more comprehensive survey of state-of-the-art methods.

Surface Consolidation. Consolidation is an important preprocessing step, such as normal estimation, denoising, smoothing and regu-

larization, which works directly on a point set itself. Early work by [2] defines a point set surface through moving least squares (MLS) projection. Later works focus on remitting the over-smoothing problem. Representative approaches include anisotropic smoothing [78], point-sampled cell complexes [1], algebraic point set surfaces [46], robust implicit MLS [102] and point set resampling [59]. To preserve sharp features, Avron et al. [7] use a ℓ_1-sparse method to compute piecewise smooth surfaces, whereas Huang et al. [59] generate piecewise smooth point set surfaces through point projection. Recent work by Calderon and Boubekeur [15] preserves sharp features under a point morphology framework. All of these techniques depend on oriented normals for the projection control.

The most related work to our approach is Weighted Locally Optimal Projection (WLOP) [85, 57], which generates a uniform distributed point set with oriented normals. Preiner et al. [107] develop an accelerated version of WLOP by using a more compact representation of the original input points. WLOP-based consolidation methods are robust since they do not rely on the normals of the input points. However, none of them was designed to complete point clouds. In that sense, our method fuses consolidation and completion into one coherent technique.

Completion. Missing data, caused by self-occlusion, light absorption, or challenging surface materials [155], is one of the most challenging problems in surface reconstruction. Diffusion-based methods [28] are able to fill small holes with complex boundaries. To fill large holes, context-based [122, 48], and repetition-based [165] methods are proposed, with the assumption that the missing parts can be replaced with other parts of the input itself. Other methods infer missing data through exploiting high-level knowledge and priors, such as symmetry relationships [103], volumetric smoothness [131], canonical regularities [82], and global parity measurement [121]. Nevertheless, these techniques cannot avoid erroneous topological reconstructions. Hence, interactive methods were developed to allow guiding the reconstruction with topology control [124, 159].
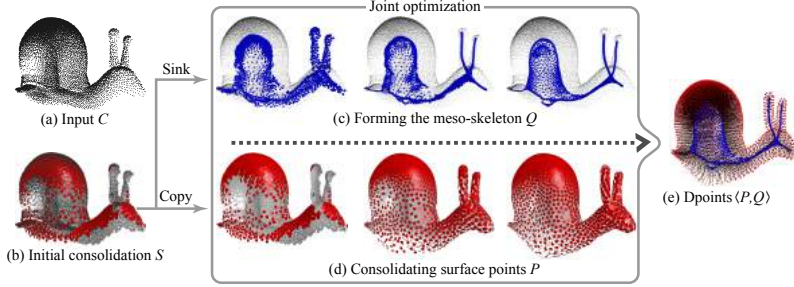
Figure 3.3: Deep points consolidation. Given the input point cloud (a) and its initial consolidation results (b), our approach creates deep points by sinking the inner points to form a meso-skeleton (c) and moving the outer points along the surface to complete missing areas (d). The final representation consists of a set of coherent vectors that connects the surface with the meso-skeleton.

Skeletonization.   Skeletonization has been intensely studied in computer graphics, e.g., [6, 17, 14, 98]. In particular, curve skeleton techniques were developed in the context of surface reconstruction [123, 132, 81, 87, 58]. Tagliasacchi et al. [132] introduce a curve skeleton extraction mechanism by the use of the rotation symmetry axis. Huang et al. [58] compute L_1-medial skeletons with conditional regularization.

The work of Tagliasacchi et al. [129] computes meso-skeletons, which combine medial sheets and curve skeletons. We adapt their notion of meso-skeletons. However, our method differs in that we do not assume having a complete surface with reliable normals. We generate the meso-skeleton as a by-product of the point cloud consolidation from noisy and incomplete input that lacks reliable normals. Miklos et al. [94] introduce a generalization of the MAT that provides a medial representation at different levels of abstraction. While this approach is more robust to noise, it still does not address the issue of incomplete data.

To complete the missing parts along the recovered curve skeleton,

these methods [132, 58] typically run cross-sectional curve interpolation and assume that the geometry of the cross-sectional curve is circular or elliptical. Our approach for computing meso-skeletons is inspired by the skeletonization scheme of [58]. However, we introduce new adaptive and anisotropic neighborhoods (Figure 3.5), which allow us to generate skeletons of different topologies (2D sheets & 1D curves) under the same framework.

3.3 Deep Points

A deep point consists of a pair of points $\langle p_i, q_i \rangle$, an outer point $p_i \in \mathbb{R}^3$ that resides on the surface and a corresponding inner point $q_i \in \mathbb{R}^3$ that resides on a meso-skeleton. We refer to the outer points $P = \{p_i\}_{i \in I}$ as surface points and the inner ones $Q = \{q_i\}_{i \in I}$ as skeletal points, where the index set I is the same. We represent points in space and their orientations as column vectors, and denote orientation vectors with bold fonts.

The deep points representation $\langle P, Q \rangle = \{\langle p_i, q_i \rangle\}_{i \in I} \subset \mathbb{R}^6$ shall satisfy the following criteria:

- the surface points P reside on the latent surface and are regularly distributed;

- the collection of the skeletal points Q forms a meso-skeleton of the surface, which may consists of both 2D surface sheets and 1D curves;

- the dpoint orientation $\mathbf{m}_i = (p_i - q_i)/\|p_i - q_i\|$ agrees with the surface normal $\mathbf{n}_i$ at the corresponding surface point p_i.

These objectives yield a joint optimization of the surface and the skeletal points under both *position* and *orientation* constraints. As the optimization converges, the deep points are generated.

We provide an overview of our algorithm in Figure 3.3. Let us denote the input point cloud with $C = \{c_j\}_{j \in J} \subset \mathbb{R}^3$ (Figure 3.3(a)). The point cloud can be unoriented and unevenly distributed, and it

may have large missing parts. We start by applying WLOP consolidation [57] on C to yield a denoised, oriented, and downsampled point set $S = \{s_i, \mathbf{n}_i\}_{i \in I}$ (Figure 3.3(b)). Through replicating the consolidated input S and sinking these replicated points down into the shape interior (Section 3.4.1), an initial meso-skeleton (Figure 3.3(c), left) is formed. We refer to points on this initial meso-skeleton as anchor points and denote them by $H = \{h_i\}_{i \in I}$.

With set S serving as the initial surface points (Figure 3.3(d), left) and set H serving as the initial skeletal points, we optimize the set $\langle S, H \rangle$ (Figure 3.3(d) and (c), left to right) to form a valid deep points representation that satisfies the three criteria introduced above. In Section 3.4.2, we describe the optimization of the skeletal points into a connected and topologically correct meso-skeleton of the surface. The corresponding surface point consolidation is presented in Section 3.4.3. We show a sequence of optimized point clouds in Figure 3.3(c) and (d), including both skeletal and surface point sets, to demonstrate how this joint optimization converges to a valid state of deep points $\langle P, Q \rangle$ as presented in Figure 3.3(e).

3.4   Joint Optimization on Dpoints

The joint optimization of dpoints consists of three stages: sinking the consolidated points S (Figure 3.3(b)) to obtain a set of anchor points H (Figure 3.3(c), left), and then consolidating these anchor points H to form a meso-skeleton (Figure 3.3(c), left to right), and finally consolidating a second copy of the surface points S to complete the missing areas and refine the connection to the meso-skeleton (Figure 3.3(d), left to right). At the end, a good dpoints representation is obtained, which consists of a meso-skeleton with proper topology, a complete and consolidated point set surface, and an one-to-one mapping between the two, where the mapping (dpoint) orientation agrees with the surface normal (Figure 3.3(e)).

3.4.1 Sinking Consolidated Points

Our algorithm to obtain an initial meso-skeleton is loosely inspired by the grassfire transform for binary images. Intuitively, we set the consolidated points "on fire", causing the fire to burn from the surface into the volume of the shape. When opposing fire fronts meet, they stop propagating and form the initial meso-skeleton. We consider points that have met an opposing front as "anchored". A key challenge in this step is that we work with point sets that may have large missing parts, i.e., are not closed, which may lead to regions that never meet an opposing front. We handle this problem by exploiting the dpoints representation, that is, the connection between moving inner points and their corresponding surface points.

We initialize outer and inner points of each dpoint $\langle p_i, q_i \rangle$ by placing both at the corresponding consolidated point s_i. This gives us dpoints of zero length, where the length means the distance between the outer and inner point. To move inner points into the volume of the shape and anchor them at proper locations, we apply an iterative procedure. For each dpoint, we compute a static neighborhood $\mathcal{P}_i$ on the surface points, and a dynamic neighborhood $\mathcal{Q}_i$ on the inner points, which we update at each iteration. We use the static neighborhood to determine the amount of movement into the volume for each inner point, and to maintain smoothness among neighboring inner points. On the other hand, we use the dynamic neighborhoods to determine whether opposing fronts meet and inner points settle.

In a preprocessing step, we first compute the average sparsity r of the consolidated point set S. That is, r is the average distance to the closest neighbor,

$$ r = \frac{1}{|S|} \sum_{i \in I} \min_{i' \in I \setminus \{i\}} \|s_i - s_{i'}\|, \tag{3.1} $$

where operator $\|\cdot\|$ computes the Euclidean norm of a vector and $|\cdot|$ measures the size of a set. We compute the static neighborhood $\mathcal{P}_i = \{p_{i'} \mid \|p_{i'} - p_i\| < \sigma_p r\}$ for each surface point p_i where the parameter σ_p defaults to 5, and denote its set index (the set of indices belonging to $\mathcal{P}_i$) by $I_i^{\mathcal{P}}$.

Within each iteration, we compute the dynamic neighborhood $\mathcal{Q}_i = \{q_{i'} \mid \|q_{i'} - q_i\| < \sigma_q r\}$ with the parameter $\sigma_q = 2$ by default, and denote a set index $I_i^{\mathcal{Q}}$ for each inner point q_i. We use this neighborhood to detect whether an inner point q_i should settle. We say opposing fronts meet within the dynamic neighborhood if

$$\max_{i' \in I_i^{\mathcal{Q}}} \mathbf{n}_{i'} \cdot \mathbf{n}_i \leq \cos(\omega), \tag{3.2}$$

where $\mathbf{n}_i$ are the normals of surface points, which we obtained in the initial consolidation step [57]. Intuitively, the above condition means that an inner point stops sinking once it meets other inner points with sufficiently different normals. The criteria for being sufficiently different is based on the parameter ω. It controls how deep the inner points sink and it defaults to 45°.
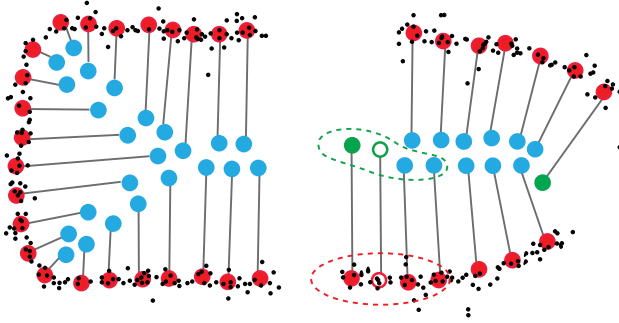
Figure 3.4: Generating anchor points by sinking consolidated (red) points. Most inner points (blue) stop sinking as they get close to other inner points with different normals, but some (green) need to be stopped through bilateral smoothing. The dashed green curve shows the neighborhood used for one inner point (hollow green) during smoothing. This neighborhood is static and is determined based on the corresponding surface point (hollow red).

Next, we move each unsettled inner point by a small distance along the direction opposite to its normal, i.e., $q_i = q_i - t\mathbf{n}_i$. In the

(a) Input (C').   (b) WLOP (S).   (c) Contraction of (b).   (d) Anchor points (H).   (e) Contraction of (d).   (f) Meso-skeleton.
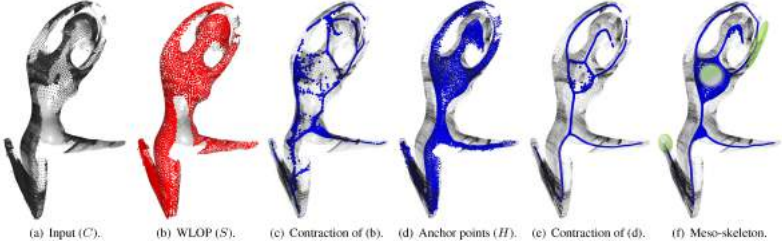
Figure 3.5: Given a highly incomplete raw scan (a) and its WLOP consolidated points (b), direct point contraction fails to provide a good set of skeletal points (c). Sinking the inner points first until they settle into anchor points (d) and then applying contraction yields cleaner skeletal curves (e), but does not form skeletal surface sheets. Using anisotropic repulsion, our approach generates the meso-skeleton (f) with mixed curves and surface sheets in a uniform framework. The green ellipsoids in (f) show the anisotropic neighborhoods used in three areas.

first iteration, the moving distance t is set to $r/2$ for all unsettled inner points. In the following iterations, we adaptively compute t for each inner point q_i. For each q_i, we set t as the average moving distance in the previous iteration of all inner points (settled and unsettled) in its static neighborhood $\{q_{i'}\}_{i' \in I_i^{\mathcal{P}}}$.

After the inner points advance to the new locations, we further adjust their positions to maintain the smoothness among neighboring inner points. We smooth inner point positions using a form of cross-bilateral filtering, where the filter weights are determined based on the surface points $\mathcal{P}_i$ in the static neighborhood of each inner point. In particular, the weights measure the proximity and normal similarity of neighboring surface points. This leads to the cross-bilateral filter updates of inner point positions q_i as:

$$q_i = \frac{\sum_{i' \in I_i^{\mathcal{P}}} \theta(p_i, p_{i'}) \phi(\mathbf{n}_i, \mathbf{n}_{i'}) q_{i'}}{\sum_{i' \in I_i^{\mathcal{P}}} \theta(p_i, p_{i'}) \phi(\mathbf{n}_i, \mathbf{n}_{i'})},$$

with the bilateral weighting functions:

$$\theta(p_1, p_2) = e^{-\left(\frac{\|p_1 - p_2\|}{r}\right)^2}, \phi(\mathbf{n}_1, \mathbf{n}_2) = e^{-\left(\frac{1 - \mathbf{n}_1^T \mathbf{n}_2}{1 - \cos(\omega)}\right)^2}. \qquad (3.3)$$

Here the parameter r is computed in Eq (3.1) and the parameter ω is the same one used in Eq (3.2).

It is worth noting the importance of using the static neighborhood $\mathcal{P}_i$ over surface points, and the positions and normals of surface points in this neighborhood, to compute the weights in Eq (3.3). As shown in Figure 3.5(d), this strategy helps the inner points to maintain their structure as they move into the volume. In addition, since both settled and unsettled points are used for smoothing, an unsettled point may stop traveling and be marked as settled after most of its neighbors are settled. This feature is very important for handling highly incomplete point clouds because there may be points that will never run into other points as determined by Eq (3.2). They hence would never stop traveling along their anti-normal directions; see e.g., the green points in Figure 3.4.

3.4.2   Forming the Meso-Skeleton

Once all the inner points settle at the interior of the 3D shape, they form a set of fixed anchor points $H = \{h_i\}_{i \in I}$. As shown in Figure 3.5(d), these anchor points do not necessarily form an intuitive meso-skeleton consisting of connected curves and surfaces. In addition, due to the missing data, there are sparse areas where the anchor points still leave gaps. We compute a point-based meso-skeleton with topologically-correct connections by further consolidating the inner points. We formulate an optimization with two objectives: first, to keep the inner points close to the L_1-median of their neighboring anchor points; second, to uniformly distribute them along connected curves and surface sheets.

The key issue in implementing both objectives is to determine a proper anisotropic neighborhood for each inner point. In cylindrical areas, we are aiming to obtain skeletal curves, hence neighborhoods of

long and thin prolate ellipsoid shapes are appropriate. For plane-like areas, we want to generate skeletal surface sheets, and flat neighborhoods of large oblate ellipsoid shapes would work best. Finally, in areas near the endpoints of skeletal curves or at boundaries of skeletal surfaces, a small ellipsoid should be used to avoid shrinkage of the skeletal curves or surfaces.

To automatically determine the appropriate elliptical neighborhood at an inner point q_i, we first apply PCA on inner points within q_i's neighborhood $\mathcal{Q}_i$ (as computed in Section 3.4.1). We use the three principle axes $\{\mathbf{v}_i^1, \mathbf{v}_i^2, \mathbf{v}_i^3\}$ as the local coordinates of the ellipsoid. Instead of using the PCA eigenvalues to determine the semi-axis lengths of the ellipsoid, however, we develop a different procedure exploiting our dpoints representation. We will discuss the benefits of this below. Let us denote the semi-axis lengths by scalars $\{l_i^1, l_i^2, l_i^3\}$. We compute them by first projecting the normal directions of all inner points within the neighborhood $\mathcal{Q}_i$ to the corresponding principle axis and then computing the average normal projection length. The larger the projection length, the shorter the semi-axis length l is set to. To be precise, we set

$$l_i^m = \left(\frac{\sum_{i' \in I_i^{\mathcal{Q}}} |\mathbf{n}_{i'}^T \mathbf{v}_i^m|}{|\mathcal{Q}_i|} + \epsilon \right)^{-1}, \quad m = 1,\ 2,\ 3.$$

The constant parameter ϵ is small and set to 0.1 by default.

As shown in Figure 3.5(f) and illustrated in Figure 3.6, the ellipsoids computed can automatically adapt to the local topology of the inner points. In the center regions of the skeletal curves or skeletal surface sheets, where surface normals are perpendicular to the dominant PCA directions, the normal projection length is small, resulting in a large semi-axis length. This helps to connect gaps on skeletal curves and close holes on skeletal surface sheets. On the other hand, at the end points of skeletal curves and the borders of skeletal surface sheets, the normal projection length is large even along the dominant PCA directions. Hence, a small semi-axis length is used to avoid shrinkage in these areas, because the inner points are constrained to smaller regions.

With the anisotropic neighborhood defined, we now formulate the meso-skeleton as the result of an optimization that attempts, first, to keep the inner points close to the L_1-median of their neighboring anchor points, and second, to consolidate the inner points to form regularly sampled, connected skeletal curves and surfaces. Hence, the optimization is a sum of a data and a regularization term:

$$argmin_Q \sum_{i \in I} \sum_{k \in I} \vartheta(q_i, h_k) \|q_i - h_k\| + R(Q). \qquad (3.4)$$

The first term (data term) is the weighted L_1-median, where the weight function $\vartheta(q_i, h_k) = e^{-d_e^2(q_i, h_k)/r^2}$ is defined based on the ellipsoid (Mahalanobis) distance between the neighboring anchor point h_k and the center point q_i. This distance function returns the same constant number for all points on the ellipsoid surface defined at q_i; and hence gives them equal weights. It is computed as

$$d_e(q_i, h_k) = \|\mathbf{A}_i^T (q_i - h_k)\|,$$

where the 3×3 column matrix $\mathbf{A}_i = [\mathbf{v}_i^1/l_i^1; \mathbf{v}_i^2/l_i^2; \mathbf{v}_i^3/l_i^3]$ consists of the semi-axes of our elliptical neighborhoods.

Applying the L_1-median term alone tends to yield a sparse but non-regular point distribution; see Figure 3.5(e) where local L_1-median centers are accumulated in clusters. We avoid clustering using the regularization term $R(Q)$, which represents additional repulsion forces. Our approach is derived from previous ones [85, 57], but with important differences. The key idea is that to allow the inner points to form thin skeletal curves or surfaces, we cannot repulse points along the directions that are perpendicular to the skeletal curves or surfaces. Hence, we use our elliptical neighborhoods to define novel anisotropic repulsion forces. The regularization term is a sum of these forces at each inner point q_i, summed over all inner points $i \in I$ with $\{\lambda_i\}$,

$$R(Q) = \sum_{i \in I} \lambda_i \sum_{i' \in I \setminus \{i\}} \frac{\vartheta(q_i, q_{i'})}{\|d_e(q_i, q_{i'})\|^3}. \qquad (3.5)$$

Intuitively, the repulsion force between an inner point $q_i \in \mathcal{Q}$ and a neighbor $q_i', i' \in I \setminus \{i\}$ is given by the projection of the neighbor
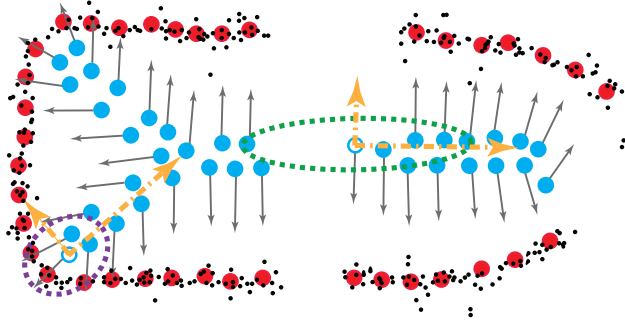
Figure 3.6: The anisotropic neighborhoods (dashed ovals) defined for two inner points along their local PCA axes (orange arrows). Even though the eigenvalues along the dominant PCA direction are high in both cases, the normal projection lengths are different, resulting in the green oval being much longer than the purple one.

into the local coordinates of the ellipsoid $\{\mathbf{v}_i^1, \mathbf{v}_i^2, \mathbf{v}_i^3\}$, and applying stronger repulsion along the direction where the semi-length of the ellipsoid is longer. The balancing weight λ_i is determined by a regularization parameter μ that we define next in this subsection.

It is also worth noting that our anisotropic regularization is different from that of the L_1-medial skeleton [58]. There the conditional regularization stops to push points whenever the skeletal points are forming a curve structure. Under the same situation, our anisotropic repulsion still pushes points along the curve direction, allowing them to connect broken skeletal curves and to obtain a more uniform point distribution. This not only removes the need of finding bridge points to connect broken curves, but also handles different topologies (curves and surface sheets) under a uniform framework.

When the gradient of the energy in Eq (3.4) equals to zero, the following relation is satisfied at each q_i with fixed coefficients:

$$\sum_{i' \in I \setminus \{i\}} \alpha_{ii'} (q_i - q_{i'}) - \lambda_i \sum_{i' \in I \setminus \{i\}} \beta_{ii'} \mathbf{A}_i^T \mathbf{A}_i (q_i - q_{i'}) = 0,$$

(a) meso-skeleton

(b) Repulsion force   (c) Fitting with prior   (d) normal update
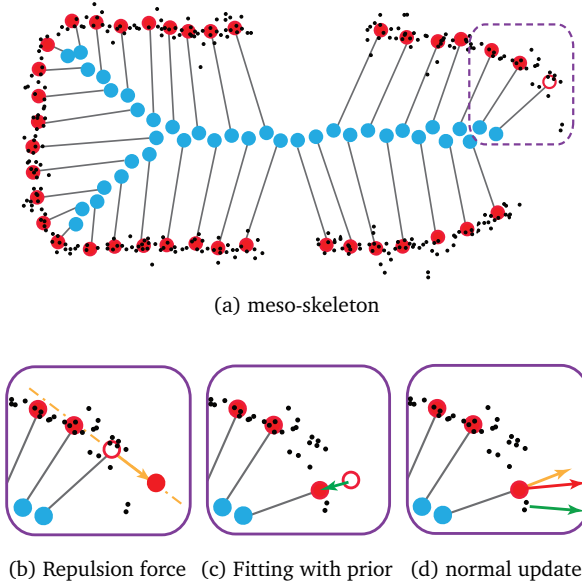
Figure 3.7: Once the meso-skeleton (a) is formed, we consolidate the surface points through alternatively optimizing surface point locations and updating their normals. The repulsion force introduced by the regularization term pushes the point along the tangential (orange line) direction (b), whereas the data fitting term with the shape prior moves the point towards the local L_1-median with adjustment on the dpoint length (c). (d) Once the optimal location is found, the normal direction is updated based on dpoint orientation (orange arrow) and PCA normal (green arrow).

where $\alpha_{ii'} = \frac{\vartheta(q_i, q_{i'})}{\|q_i - q_{i'}\|}$ and $\beta_{ii'} = \frac{\vartheta(q_i, q_{i'})}{\|d_e(q_i, q_{i'})\|^5}$.

Applying a fixed point iteration, we update inner points by

$$q_i = \frac{\sum_{i' \in I \setminus \{i\}} \alpha_{ii'} q_{i'}}{\sum_{i' \in I \setminus \{i\}} \alpha_{ii'}} + \mu \|\ell_i\|^2 \frac{\sum_{i' \in I \setminus \{i\}} \beta_{ii'} \mathbf{A}_i^T \mathbf{A}_i (q_i - q_{i'})}{\sum_{i' \in I \setminus \{i\}} \beta_{ii'}}.$$

Here we have $\mu\|\ell_i\|^2 = \lambda_i \sum_{i' \in I \setminus \{i\}} \beta_{ii'} / \sum_{i' \in I \setminus \{i\}} \alpha_{ii'}$ and the column vector $\ell_i = [l_i^1; l_i^2; l_i^3]$. We empirically set the regularization parameter $\mu = 0.4$ by default to control the global level of anisotropic penalty applied on accumulated points.
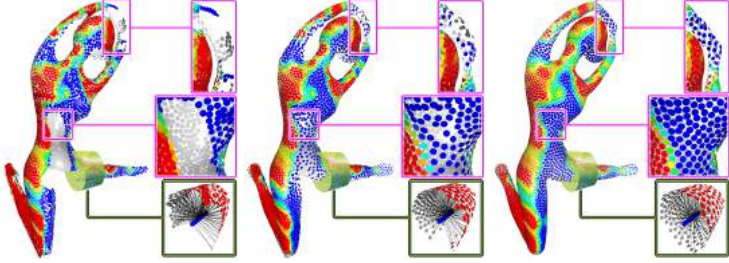
Figure 3.8: The initial consolidated points contain large missing areas (left). The surface consolidation process gradually closes the gaps (middle) and finally converges to a set of complete surface points (right). The color of surface points encodes the density of local input points.

3.4.3 Surface Point Consolidation

As shown in Figure 3.5, after the second stage, the inner points form a complete meso-skeleton (2D sheets & 1D curves) with a regular point distribution. The meso-skeleton now provides the non-local information that can guide surface point consolidation to complete large missing areas. In practice, the location of each p_i is adjusted through optimization based on the following objectives: i) p_i resides on the underlying surface and hence shall be close to the L_1-median of nearby input points; ii) the points $\{p_i\}$ shall be regularly distributed on the underlying surface; and iii) in incompletely scanned areas where input points are missing, additional shape priors are needed to guide surface completion. See also Figure 3.7.

Based on the above objectives, we define the optimization function

for surface points as the sum of three terms: a L_1-median data term, a regularization term $\hat{R}$, and a shape prior term G:

$$argmin_P \sum_{i \in I} \eta(p_i) \sum_{c_j \in \mathcal{C}_i} \theta(p_i, c_j)\|p_i - c_j\| + \hat{R}(P) + G(P). \quad (3.6)$$

The θ function in the data term is the same as the one defined in Eq (3.3), and $\mathcal{C}_i = \{c_j \mid \|c_j - p_i\| < \sigma_p r\}$ is a subset of input points in p_i's neighborhood. The data term is weighted using a function $\eta(p_i) = 1 + \sum_{c_j \in \mathcal{C}_i} \theta(c_j, p_i)$, which outputs high weight when the density of the input points in $\mathcal{C}_i$ is high. Hence, when there are sufficient data in the neighborhood of p_i, the data term has higher weight to ensure the position of p_i fits the data.

The second regularization term applies repulsion forces introduced from all neighboring surface points. To avoid points being pushed away from the surface, here the repulsion is performed along the local tangent plane at p_i. That is:

$$\hat{R}(P) = \sum_{i \in I} \sum_{i' \in I \setminus \{i\}} \frac{\theta(p_i, p_{i'})}{\|\hat{\mathbf{A}}_i^T(p_i - p_{i'})\|^3}, \quad (3.7)$$

where the 3×2 projection column matrix $\hat{\mathbf{A}}_i = [\mathbf{u}_i^1; \mathbf{u}_i^2]$, with $\{\mathbf{u}_i^1, \mathbf{u}_i^2\}$ being two arbitrary orthogonal directions on p_i's tangential plane, ensures $\hat{R}(p_i)$ only introduces repulsion forces perpendicular to the surface normal $\mathbf{n}_i$.

When the point set surface in the neighborhood of p_i is complete, repulsion forces from all directions maintain the regularity of the point distribution. Otherwise, if p_i is near the open boundary of missing areas, its neighboring points will push it towards the hole, allowing small gaps on the surface being naturally filled. However, when the scan contains large missing areas, e.g., the object is only scanned from one side as in Figure 3.17(a), additional shape priors are needed to further guide the completion.

A big benefit introduced by the dpoints representation is the notion of volume, i.e., the length of a given dpoint $\|p_i - q_i\|$ indicates how far the surface is away from the local meso-skeleton at location p_i.

This allows us to incorporate different types of shape priors with ease. For example, under the assumption that surface points in areas with missing data shall preserve a similar volume as their neighbors, we can define a *volume preserving shape prior* term as:

$$G(P) = \frac{1}{2} \sum_{i \in I} \gamma(p_i)(\|p_i - q_i\| - L(p_i))^2, \tag{3.8}$$

where $L(p_i) = \frac{\sum_{i' \in I_i^P} \theta(p_i, p_{i'})\|p_{i'} - q_{i'}\|}{\sum_{i' \in I_i^P} \theta(p_i, p_{i'})}$ computes the average dpoint length within p_i's neighborhood, and the weight function $\gamma(p_i) = (1 + var(\{\|p_{i'} - q_{i'}\|\}_{i' \in I_i^P}))^{-1}$ assigns high weight when the dpoint length variance within the neighborhood is low.

$$G(P) = \frac{1}{2} \sum_{i \in I} \gamma(p_i)(L(p_i))^2. \tag{3.9}$$

Similarly as solving Eq (3.4), we can solve Eq (3.6) and update surface points by applying a fixed point iteration. For example, when the volume preserving shape prior from Eq (3.8) is used, the optimization can be solved by:

$$
\begin{aligned}
p_i &= \frac{\eta(p_i) \sum_{c_j \in \mathcal{C}_i} \alpha_{ij} c_j}{\gamma(p_i) + \eta(p_i) \sum_{c_j \in \mathcal{C}_i} \alpha_{ij}} + \frac{\gamma(p_i)(q_i + L(p_i)\mathbf{m}_i)}{\gamma(p_i) + \eta(p_i) \sum_{c_j \in \mathcal{C}_i} \alpha_{ij}} \\
&+ \hat{\mu}\frac{\sum_{i' \in I \setminus \{i\}} \hat{\beta}_{ii'} \hat{\mathbf{A}}_i^T \hat{\mathbf{A}}_i (p_i - p_{i'})}{\sum_{i' \in I \setminus \{i\}} \hat{\beta}_{ii'}},
\end{aligned}
$$

with weights $\alpha_{ij} = \frac{\theta(p_i, c_j)}{\|p_i - c_j\|}$ and $\hat{\beta}_{ii'} = \frac{\theta(p_i, p_{i'})}{\|\hat{\mathbf{A}}_i^T(p_i - p_{i'})\|^5}$. The parameter $\hat{\mu} = \sum_{i' \in I \setminus \{i\}} \hat{\beta}_{ii'} / (\gamma(p_i) + \eta(p_i) \sum_{c_j \in \mathcal{C}_i} \alpha_{ij})$ that controls the tangential repulsion forces defaults to 0.4.

Once the location of surface point p_i is adjusted, its corresponding surface normal $\mathbf{n}_i$ is updated. Using a similar idea as above, when dense input data is available in the local neighborhood, higher weight is given to the conventional oriented PCA normal $\tilde{\mathbf{n}}_i$; otherwise, we

set $\mathbf{n}_i$ closer to the dpoint orientation $\mathbf{m}_i$:

$$\mathbf{n}_i = \frac{(\eta(p_i) - 1)\tilde{\mathbf{n}}_i + \mathbf{m}_i}{\|(\eta(p_i) - 1)\tilde{\mathbf{n}}_i + \mathbf{m}_i\|}.$$

Alternatively optimizing the locations of surface points and updating their normals allows these surface points to redistribute along the underlying surface, converging to a regular distribution with missing areas completed. In addition, since the normal calculation considers the dpoint orientation $\mathbf{m}_i$, the two orientations generally agree with each other in the final dpoints representation.

3.5   Results

We test our deep point consolidation technique on various incomplete scans of both physical and virtual objects. The input data in Figures 3.1, 3.5, 3.9, 3.11, 3.13, 3.15 and 3.22 was acquired using a laser scanner, the ones in Figure 3.14(a) were downloaded from the SHREC 2015 dataset [99], and those used in Figure 3.17 are digital scans of a synthetic model.

Parameters and timing.   On average, the initial WLOP consolidation needs 30-40 iterations to converge, whereas the iterations needed for generating deep points (consisting of sinking WLOP points into anchor points, forming meso-skeleton, and consolidating the surface) vary from 60 to 120 depending on how large the missing data regions are. Table 3.1 presents actual computation times on an Intel(R) Core(TM) i7-5930K CPU@3.5GHz with 32GB RAM.

There are two key controllable parameters in our approach: the point settling threshold ω and the neighborhood size parameter σ_p used for both sinking and surface point consolidation. All other parameters $\{\sigma_q, \mu, \hat{\mu}, \epsilon\}$ are not sensitive to the inputs, and are fixed at the aforementioned default values for all experiments.

Figures 3.9 and 3.10 illustrates the impact of ω on the results. Generally speaking, since the inner points settle sooner under smaller

ω, the meso-skeleton follows the surface details more closely but is also more sensitive to noise. With larger ω, the inner points often travel deeper before they settle. When they cannot meet the opposing front due to missing data, the final skeleton may deviate from the proper medial position, causing the final consolidated shape having a bigger volume. In addition, some small branches may not be represented in the meso-skeleton as well. Figure 3.11 and 3.12 illustrates that larger σ_p value often leads to more uniform surface point distribution and makes the representation more robust against input noise and large missing parts. However, on the other hand, the consolidated surface may not represent the surface details as well.

Table 3.1: Timing for dpoints computation on all examples presented. $|C|$ and $|S|$ denote numbers of raw input points and deep points, respectively. T_w and T_d are the time (in sec.) needed for the initial WLOP consolidation and then dpoints consolidation, respectively.

	$|C|$	$|S|$	T_w	T_d
Fig. 3.1	84772	2143	3.1	26.3
Fig. 3.5 & 3.8	58710	6870	4.8	41.2
Fig. 3.9	22148	5234	2.7	30.5
Fig. 3.11 T	43545	3996	4.1	83.9
Fig. 3.11 B	78630	6727	8.4	87.3
Fig. 3.13 T	24292	13812	3.2	112.9
Fig. 3.13 B	32614	7252	5.5	61.5
Fig. 3.14 T	40509	40509	10.5	259.6
Fig. 3.14 B	67131	67131	18.9	408.8
Fig. 3.15 T	175312	19721	17.8	197.3
Fig. 3.15 B	300283	57791	47.3	449.9
Fig. 3.22 T	292910	32142	30.3	355.3
Fig. 3.22 B	483583	47083	134.9	723.7

Comparison to skeletonization and reconstruction. Figure 3.8 shows how the consolidation process gradually closes missing areas

by adjusting the location of surface points. The benefit of having con-
solidated surface points is shown in Figures 3.2, 3.13, 3.14, and 3.15,
where the models reconstructed from these points are noticeably more
accurate than the ones generated directly from the input or leveraging
previous skeletonization techniques [132, 58]. In particular, through
pushing surface points in an anisotropic manner, our reconstruction
results better preserve the connectivity of different parts (Figures 3.2,
3.13 and 3.15 (bottom)), while avoiding improperly fusing them to-
gether (Figure 3.14). The notion of local volume of dpoints also al-
lows the reconstruction to better preserve thin structures (Figure 3.15
(top)).

In terms of the meso-skeleton generated, Figure 3.13 shows that
our inner skeleton can better respect the topology of the input shape
than existing skeletonization approaches [132, 58]. Unlike the ex-
isting approaches, which produce skeletal curves regardless surface
topology, the meso-skeletons that we generated for non-tubular shapes
nicely form 2D skeletal surface sheets; see, e.g., the flat surfaces in
Figures 3.13 and 3.22, the cylindrical and disk shapes in Figures 3.15,
3.16 and 3.23.

Reconstruction accuracy.   To quantitatively evaluate the reconstruc-
tion accuracy, digital virtual scans of a synthetic model are used. With
the input model serving as the ground truth, the reconstruction er-
ror is measured using the average distance between all 40K vertices
on the input model and their closest points on the reconstructed sur-
face. As shown in Figure 3.17(a), the models reconstructed using
dpoints are noticeably more accurate than the one reconstructed from
input points directly. Also, as expected, the improvement is more
pronounced for incomplete scans (i.e., one or two scans are used)
than complete scans. Note that since the virtual scan is noise free,
employing consolidation methods such as WLOP, would not improve
the direct Poisson reconstruction.

Figure 3.17(b) further evaluates the reconstruction accuracy over
noise corrupted scans. Here, Gaussian noise with 2% magnitude of
the model dimension is added to the scan data. The accuracy of

Poisson reconstruction over WLOP consolidation results and dpoints consolidation are plotted. The comparison again demonstrates the benefits of using our dpoints consolidation.

Sharp features and fine details. Since the dpoints representation is computed based on a small set of WLOP points downsampled from the input, it cannot fully preserve fine geometry details and sharp features on the original model; compare e.g., examples in Figures 3.22(b) and (d). Nevertheless, additional post-processing can be applied to reconstruct these sharp features and high-frequency details from the input. Here, the edge-aware point resampling technique (EAR) [59] is applied to upsample the surface points and then project the new samples that are close to the input data to the underlying surface using bilateral projection. This can effectively recover small geometry details; compare Figures 3.22(d) and (f) bottom. Furthermore, we can run bilateral normal smoothing to adjust the normals of surface points so that sharp features can be better preserved on the consolidated surface; compare Figures 3.22(d) and (f) top.

Shape priors. When the input scans contain open boundaries, e.g., the base of the models in Figures 3.9 and 3.11, our approach with the volume preserving shape prior (3.8) tries to close it through pushing surface points around the meso-skeleton and maintaining dpoints length $\|p_i - q_i\|$ to be similar to its neighbors. While such a strategy minimizes local volume variation, the reconstructed surface tends to have a "blobby" shape and may not accurately represent the real geometry of the missing surface, which is often flat and connects known surface through edges with $\mathcal{C}^1$ discontinuities.

To address this problem, different shape priors shall be applied. For example, the minimal volume shape prior in Eq (3.9) described in Section 3.4.3 tries to close the surface so that the total volume of the shape is minimized. The dpoints representations generated using these two shape priors are compared in Figure 3.25, which demonstrates the difference between the consolidated point set surfaces, as well as the flexibility of our surface completion framework.

Limitations.   In an effort to design a simple and elegant algorithm, our approach handles all operations through local point projections. No global smoothness is enforced for either points on the meso-skeleton or the consolidated surface points. Consequently, as shown in Figures 3.5 and 3.8, although our approach successfully connects the arms around the top of the model, the skeletal curve and the reconstructed point set surface do not appear to be smooth enough. Additional post-processing may therefore be needed.

Our method fails if the initial WLOP consolidation or the initial sinking stage fails. This may happen generally if either the data has too large missing parts, or the input contains very strong noise; see e.g., stress test results shown in in Figure 3.26. Our method always generates closed watertight point sets while preserving holes as shown in Figure 3.15 (top). On the other hand, we might mistakenly preserve holes due to large missing areas as boxed in Figure 3.22 (bottom). We believe that more accurately detecting and closing holes is a separate problem that would require stronger priors or user involvement.

3.6   Conclusions and future work

We present a novel representation for point sets, which facilitates the consolidation of noisy points and the completion of missing regions. The strength of the deep points representation comes from the fact that it is comprised of both local and non-local geometric information. For example, the dpoint orientations are used to adaptively determine the anisotropic neighborhoods when generating meso-skeletons and the dpoint lengths are used to control the local volume of the shape when consolidating surface points.

Deep points encode a consolidated point set surface and its meso-skeleton, and the mapping relations between the two in a highly concise representation. We believe that the non-local information that deep points carry is powerful for many applications. This is evident by the competence of skeletons and medial axis structures. In the future, we would like to explore the potential of using deep

points representation for applications such as retrieval of point-based geometries and deformation of point set surfaces.

(a) WLOP.   (b) Anchor points ($\omega = 25°$). (c) Meso-skeleton from (b). (d) Consolidation using (c). (e) Anchor points ($\omega = 45°$).

(f) Meso-skeleton from (e).   (g) Consolidation using (f). (h) Anchor points ($\omega = 90°$). (i) Meso-skeleton from (h).   (j) Consolidation using (i).
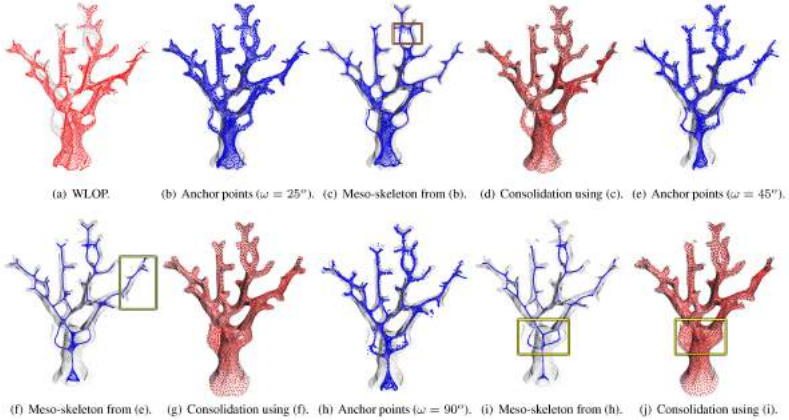
Figure 3.9: The impact of the parameter ω. With a small ω value
(b-d), the inner points settle as anchor points sooner (b), the meso-
skeleton (c) consists of many 2D sheet structures and a small number
of isolated points (orange box), and the final surface point set (d)
follows the input data reasonably well. In contrast, under a large ω
value (h-j), the inner points have to sink deeper before they settle (h),
the meso-skeleton (i) consists of mostly 1D skeletal curves and may
miss fine branches (green box in f). In addition, when a large part of
a cylindrical surface is missing, the location of the skeletal curve may
drift away from the center (yellow box in i), causing the consolidated
surface points (j) having a much thinker cylinder and small holes on
the model being improperly filled (yellow box in j).
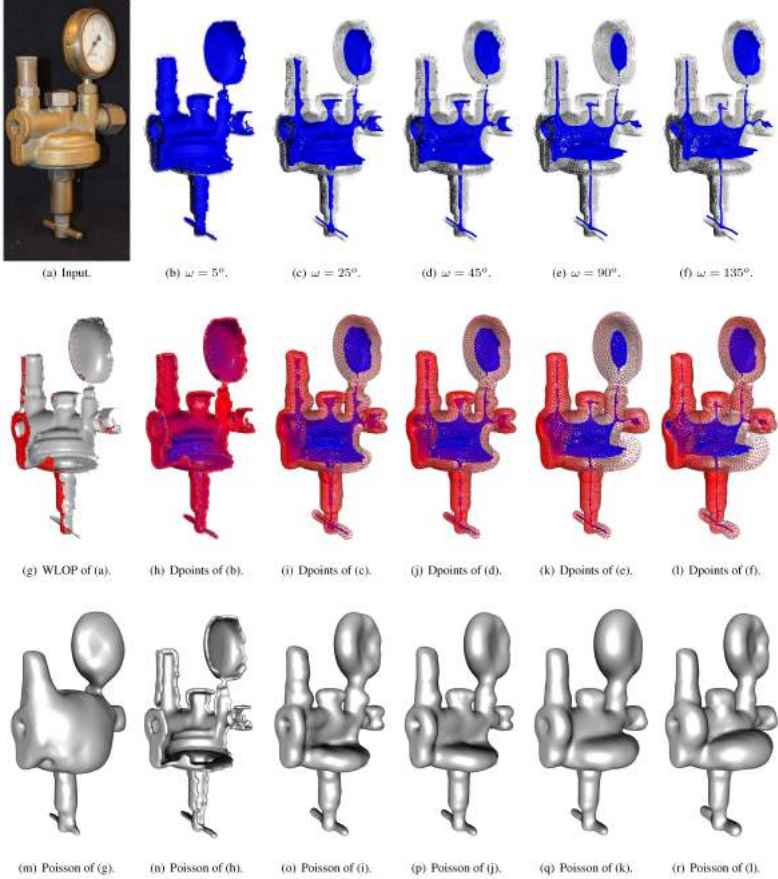
Figure 3.10: For a real object (a) that is scanned from one side only, the Poisson reconstruction result (m) generated using WLOP (g) does not accurately model the object. The results generated using our approach under different values for parameter ω are shown in (n-r). They show that, with a small value of $\omega = 5^o$, the skeletal points converge prematurely before reaching the center of the shape (b), leading to the shape not being completed properly. On the other hand, with a large value of $\omega = 135^o$, the skeletal points travel too far from the surface, resulting in inaccurate blob shapes.

(a) Photo.   (b) Raw scan.   (c) Meso-skeleton.   (d) Dpoints with $\sigma_p = 5$.   (e) Dpoints with $\sigma_p = 3$.

(f) Photo.   (g) Raw scan.   (h) Meso-skeleton.   (i) Dpoints with $\sigma_p = 5$.   (j) Dpoints with $\sigma_p = 8$.
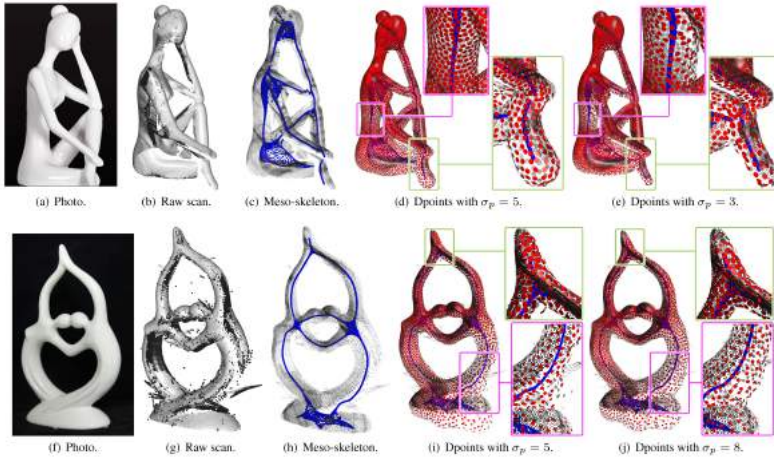
Figure 3.11: The impact of the neighborhood size parameter σ_p. For shapes with complex topology, our approach generates proper meso-skeletons (c) and (h). In (d) under the default value $\sigma_p = 5$, the partially scanned foot (green box) is noticeable thicker than the original model shown in (a). Lowering the value of σ_p in (e) makes the foot area closer to the original model, but the surface points on the body (pink box) are not as evenly distributed. In (i) due to noise in the raw scan (g), some surface points (pink box) deviate away from the underlining surface under the default value. Increasing the value of σ_p in (j) makes the point set surface more robust against input noise, but does not represent the surface details well (green box).

(a) Input mesh.

(b) Virtual scan under two different views.

(c) Meso-skeleton of input.

(d) Dpoints with $\sigma_p = 2$.

(e) Dpoints with $\sigma_p = 5$.

(f) Dpoints with $\sigma_p = 10$.

(g) Poisson reconstruction using scan (b) directly.

(h) Poisson reconstruction using dpoints in (d).

(i) Poisson reconstruction using dpoints in (e).

(j) Poisson reconstruction using dpoints in (f).
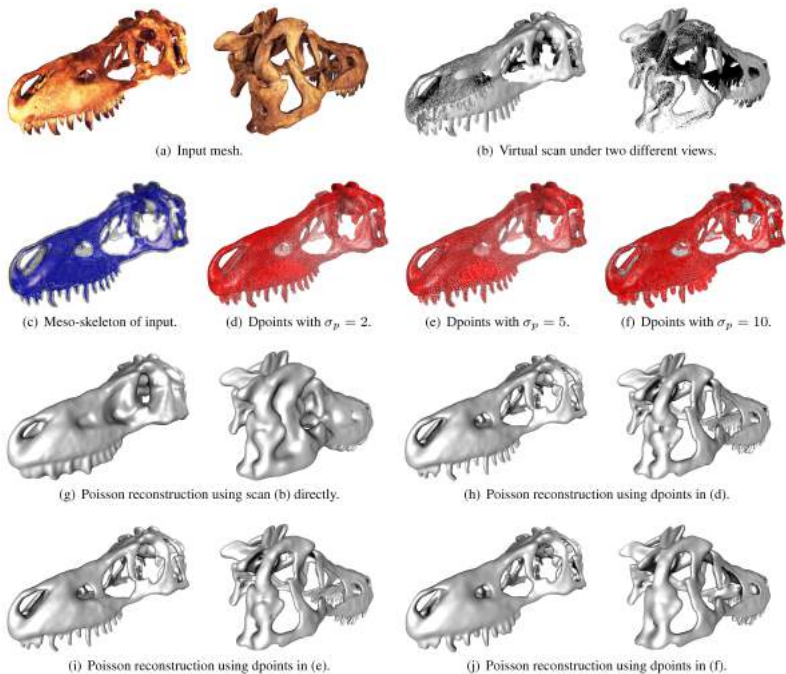
Figure 3.12: Reconstructions for a synthetic model (a) under different neighborhood size parameters σ_p. Directly applying Poisson reconstruction on the scan data (b) cannot model all the topology details (g). Our results generated under different σ_p values are visually similar, which suggests the algorithm is relatively robust against this parameter setting.

(a) Input + Poisson.   (b) ROSA + Poisson.   (c) L_1-medial + Poisson.   (d) Deep points + Poisson.
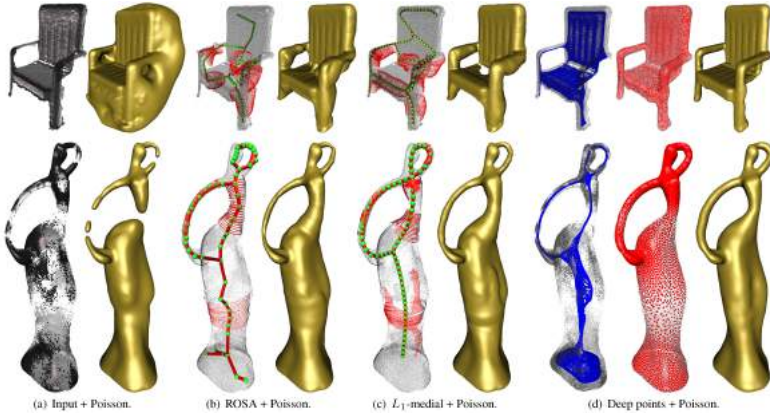
Figure 3.13: A comparison among the Poisson surface reconstructions [71] obtained using input points directly (a), ROSA skeleton [132] (b), L_1-medial skeleton [58] (c), and our dpoints consolidation (d).

(a) Input scan.   (b) Poisson of (a).   (c) Skeletal points.   (d) Surface points.   (e) Poisson of (d).
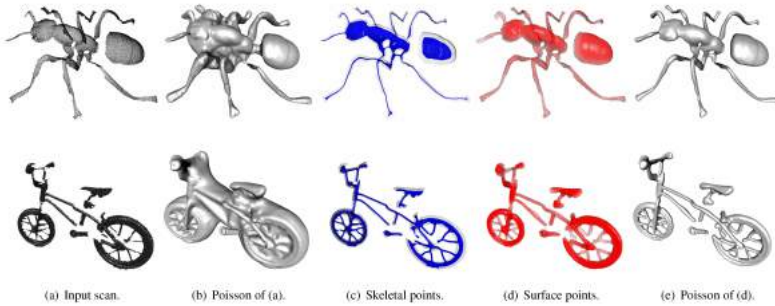
Figure 3.14: Results on standard benchmark 3D scans (a), which are downloaded from the SHREC 2015 dataset [99]. The direct Poisson reconstruction results (b) incorrectly fused multiple parts together. Using the consolidated dpoints (c & d), the thin and adjacent structures are better preserved.

(a) Photo.   (b) Input scan.   (c) Dpoints.   (d) Poisson of (c).   (e) EAR of (c).   (f) Poisson of (e).
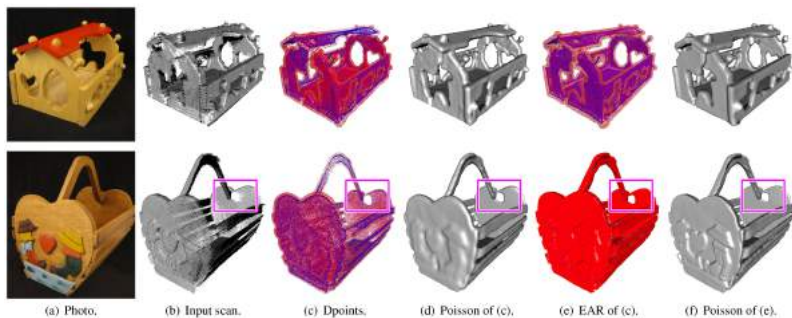
Figure 3.15: Handling objects (a) with complicate thin and non-tubular structures. Directly applying Poisson reconstruction over WLOP (b) failed to provide satisfying results (d). Our reconstruction results (e) based on dpoints consolidation (c) better preserve the thin and non-tubular structures while maintaining the correct connectivity of different parts.

(a) Photo.   (b) Incomplete scan.   (c) WLOP.   (d) EAR.

(e) ROSA.   (f) L_1-medial.   (g) Skeletal points.   (h) Surface points.
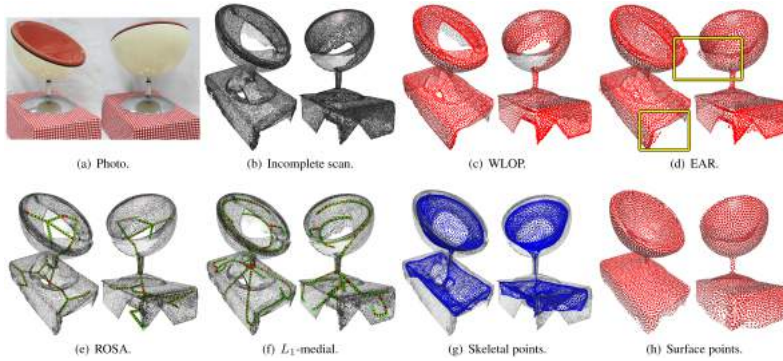
Figure 3.16: Consolidation for an incomplete scan. The input scan (b) contains large holes, which WLOP consolidation (c) cannot complete. Through inserting additional points, edge-aware resampling (EAR) (d) can fill large interior holes with some artifacts (see regions within yellow boxes), but cannot close open boundaries. Due to the non-tubular shape of the object, existing skeletonization algorithms fail to produce meaningful skeletons (e & f). In comparison, the meso-skeleton we computed (g) consists both 1D curve and 2D surface sheet. It better respects the object topology, while at the same time fills the large holes. As a result, the surface point consolidation step (h) can complete both interior holes and open boundaries.

(a) 1 scan.   (b) Poisson of (a). (c) Dpoints of (a). (d) Poisson of (c).

(e) Under clean scans.   (f) Under noise-corrupted scans.
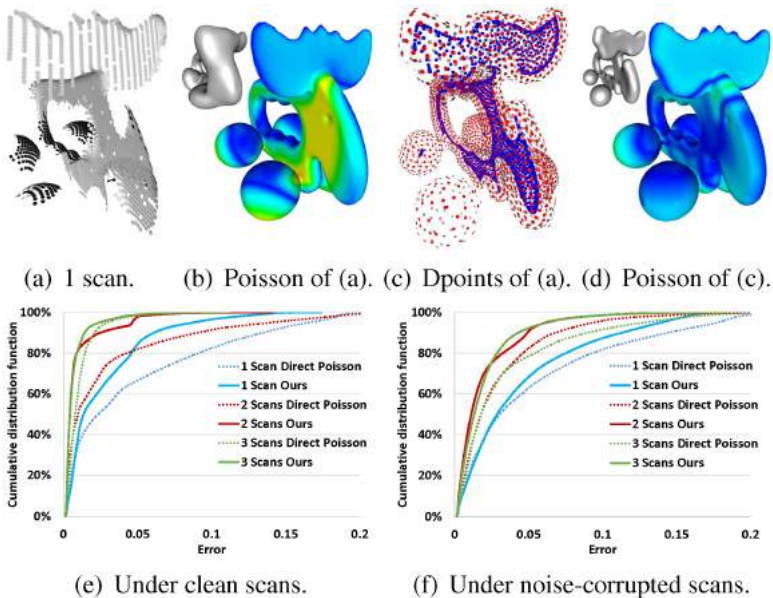
Figure 3.17: Quantitative evaluation on reconstruction accuracy using virtual scans of a ground truth synthetic model. When a single scan (a) is used, the direct Poisson reconstruction result (inset in (b)) does not resemble the model (shown in (b)). In comparison, the Poisson reconstructed model (inset in (d)) based on dpoints (c) is visually much more accurate. The reconstruction errors, measured using the distances between vertices on the ground truth model and their closest points on the reconstructed surface, are visualized in (b) and (d). The error distributions under clean and noise-corrupted scans are plotted in (e) and (f), respectively.

(a) 1 scan.   (b) Dpoints of (a).   (c) 2 scans.   (d) Dpoints of (c).   (e) 3 scans.   (f) Dpoints of (e).

(g) Poisson of (a).   (h) Poisson of (b).   (i) Poisson of (c).   (j) Poisson of (d).   (k) Poisson of (e).   (l) Poisson of (f).

(m) Error of (g).   (n) Error of (h).   (o) Error of (i).   (p) Error of (j).   (q) Error of (k).   (r) Error of (l).

(s) Error distribution

Figure 3.18: Quantitative evaluation on reconstruction accuracy using a synthetic toy model. Under the same number of digital scans, the Poisson reconstructed model using dpoints consolidation (h, j, l) is more accurate than using the raw input directly (g, i, k). The reconstruction errors, measured using the distances between vertices on the ground truth model and their closest points on the reconstructed surface, are visualized in (m-r). The error distributions are plotted in (s).

(a) 1 scan. (b) Dpoints of (a). (c) 2 scans. (d) Dpoints of (c). (e) 3 scans. (f) Dpoints of (e).

(g) Poisson of (a). (h) Poisson of (b). (i) Poisson of (c). (j) Poisson of (d). (k) Poisson of (e). (l) Poisson of (f).

(m) Error of (g). (n) Error of (h). (o) Error of (i). (p) Error of (j). (q) Error of (k). (r) Error of (l).

(s) Error distribution

Figure 3.19: Quantitative evaluation on reconstruction accuracy for noise corrupted scans. Gaussian noise with magnitude of 2% of the model dimension is added to the digital scans shown in Figure 3.18. Performing WLOP consolidation (a, c, e) before Poisson reconstruction helps to suppress noise, but it does not complete missing areas, resulting in poor reconstructions (g, i, k). In comparison, the Poisson reconstructed model using dpoints consolidation (h, j, l) is more accurate. The error distribution plot (s) also demonstrates the difference.
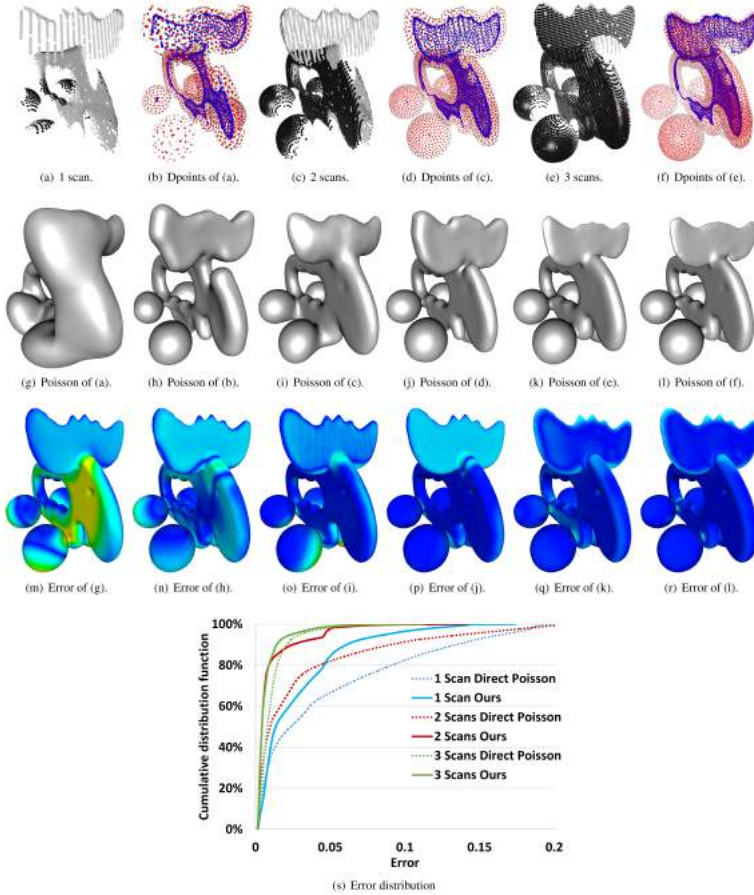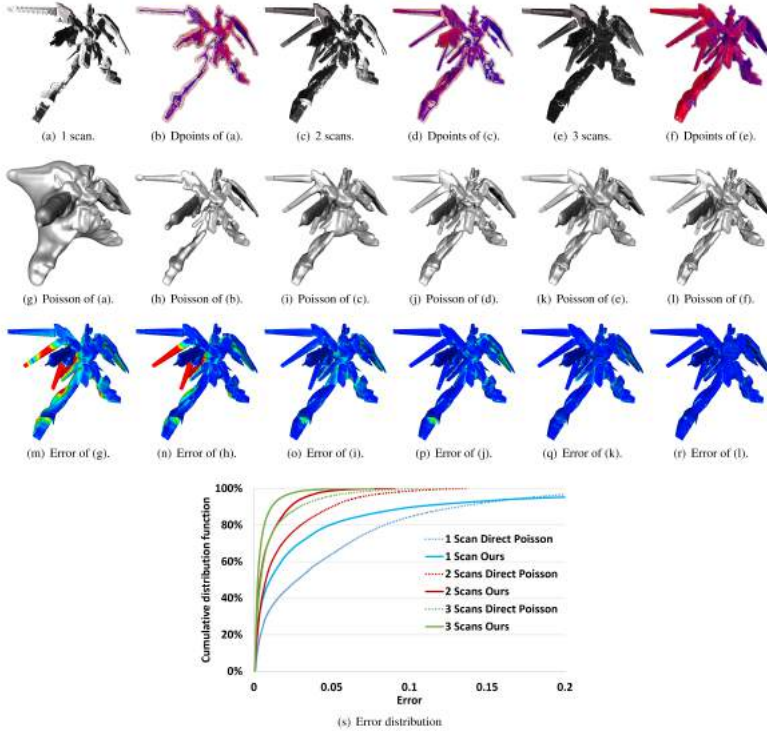
Figure 3.20: Quantitative evaluation on reconstruction accuracy using a synthetic gundam model. Under the same number of digital scans, the Poisson reconstructed model using dpoints consolidation (h, j, l) is more accurate than using the raw input directly (g, i, k). The reconstruction errors, measured using the distances between vertices on the ground truth model and their closest points on the reconstructed surface, are visualized in (m-r). The error distributions are plotted in (s).
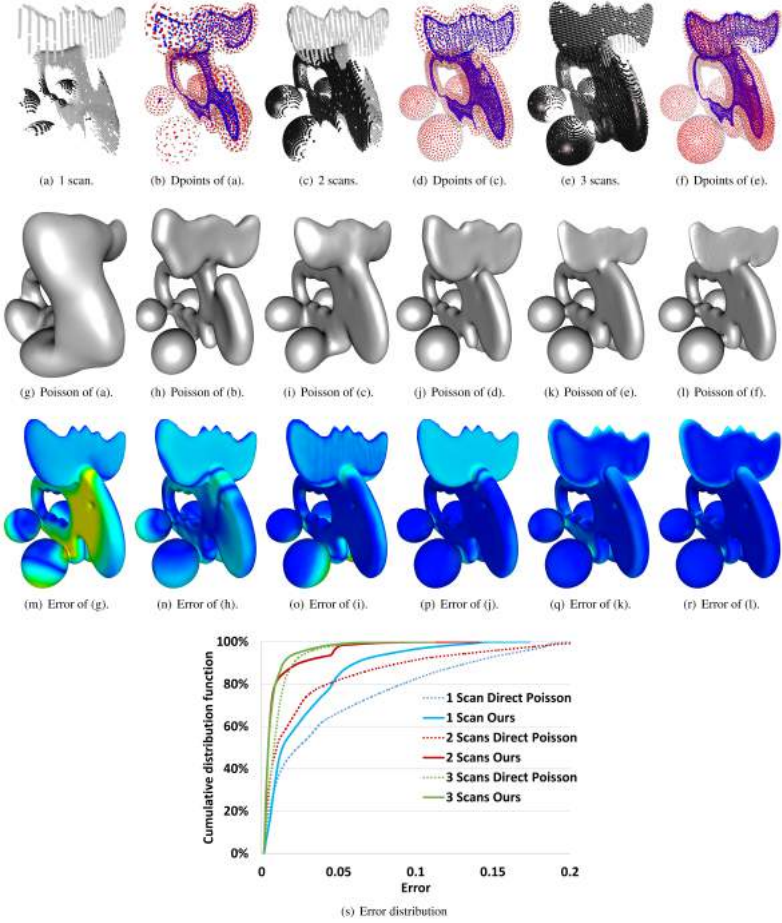
(a) WLOP of 1 scan.   (b) Dpoints of (a).   (c) WLOP of 2 scans.   (d) Dpoints of (c).   (e) WLOP of 3 scans.   (f) Dpoints of (e).

(g) Poisson of (a).   (h) Poisson of (b).   (i) Poisson of (c).   (j) Poisson of (d).   (k) Poisson of (e).   (l) Poisson of (f).

(m) Error of (g).   (n) Error of (h).   (o) Error of (i).   (p) Error of (j).   (q) Error of (k).   (r) Error of (l).

(s) Error distribution

Figure 3.21: Quantitative evaluation on reconstruction accuracy for noise corrupted scans. Gaussian noise with magnitude of 1% of the model dimension is added to the digital scans shown in Figure 3.20. Performing WLOP consolidation before Poisson reconstruction helps to suppress noise, but it does not complete missing areas, resulting in poor reconstructions (g, i, k). In comparison, the Poisson reconstructed model using dpoints consolidation (h, j, l) is more accurate. The error distribution plot (s) also demonstrates the difference.
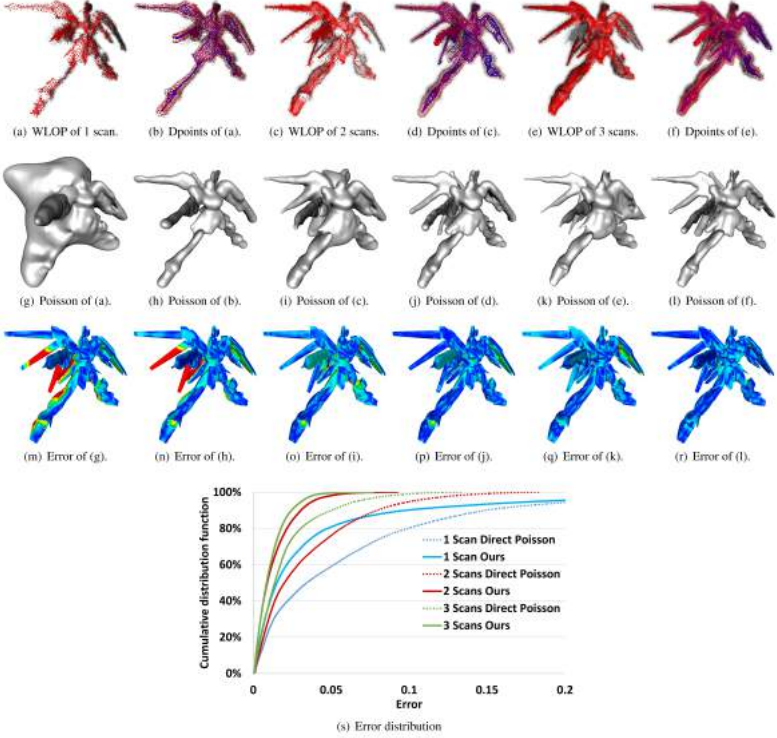
(a) Photo.   (b) Initial WLOP.   (c) Dpoints.   (d) Poisson of (b).   (e) Poisson of (c).
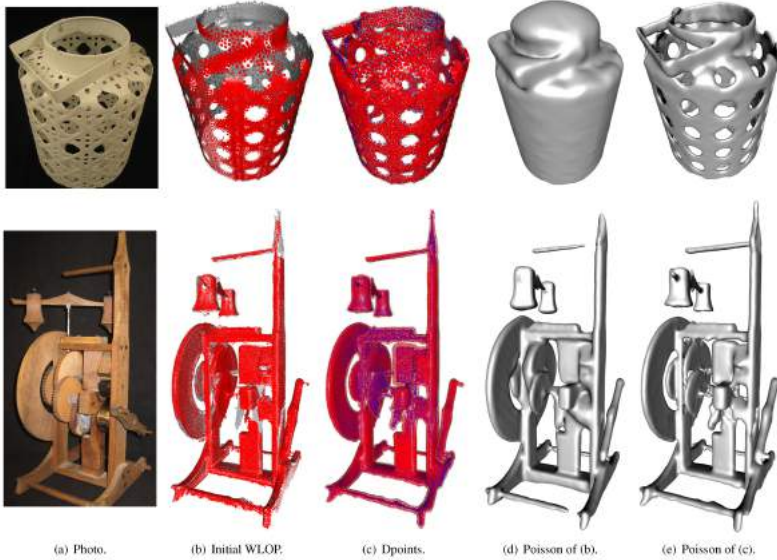
Figure 3.22: Post-processing for reconstructing fine geometry details and sharp features. While due to downsampling, the Poisson reconstruction results (d) on dpoints (c) cannot preserve fine details and sharp features as well as on the original shapes (a, b), the post EAR [59] step (e) effectively helps to recover them (f) through inserting and projecting additional dpoints.

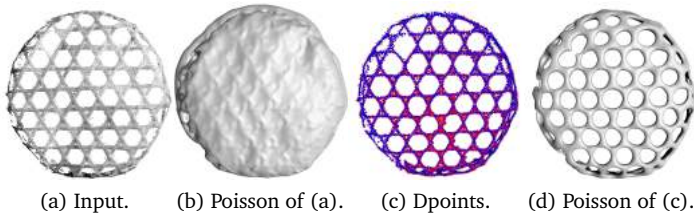(a) Input.   (b) Poisson of (a).   (c) Dpoints.   (d) Poisson of (c).

Figure 3.23: With dpoints consolidation (c) the Poisson reconstruction (d) can well restore the object's thin structures and close-by surface sheets, even with input points only from a single scan from one side (a).

(a) Input scan.   (b) Dpoints.   (c) Poisson of (c).
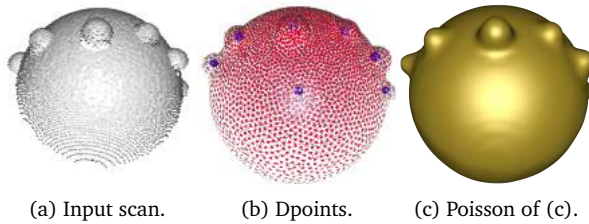
Figure 3.24: Dpoints representations under different scale of detail.

Figure 3.25: Dpoints representations under different shape priors. When there are $\mathcal{C}^1$ discontinuities between known and missing surfaces, using the volume preserving shape prior (left) results in "blobby" shapes. The minimal volume shape prior can be applied to alleviate this problem (right).

(a) Photo.

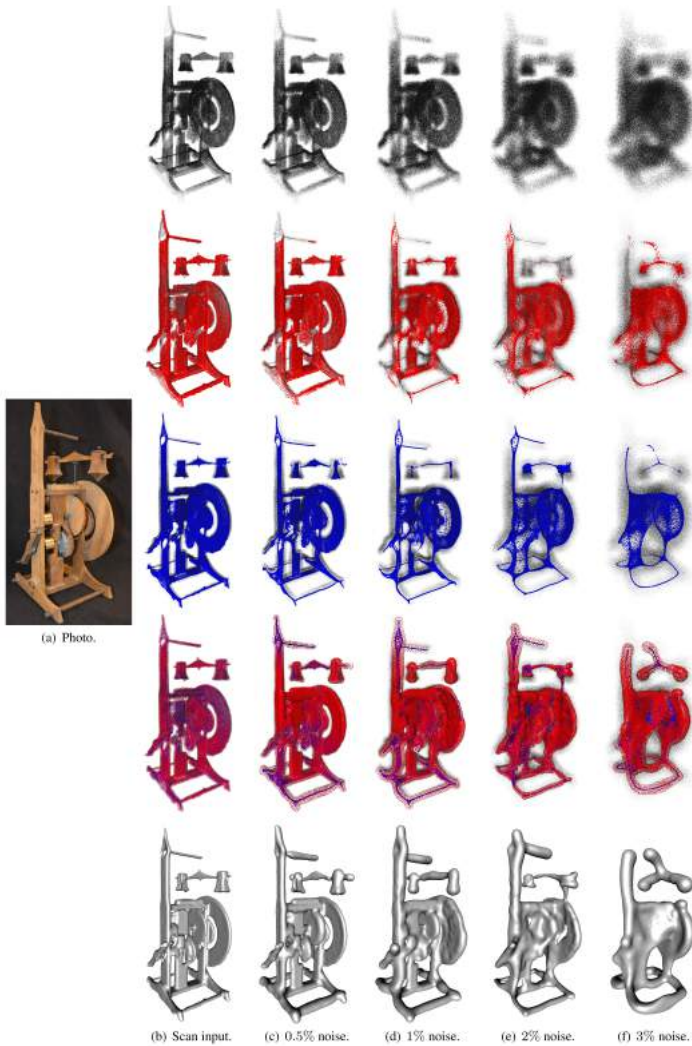(b) Scan input.   (c) 0.5% noise.   (d) 1% noise.   (e) 2% noise.   (f) 3% noise.

Figure 3.26: Stress test performed by corrupting the scan data of a real complex object (a) with Gaussian noise of different magnitudes. From top to bottom we show in row: raw data, initial WLOP results, meso-skeletons, dpoints representations, and Poisson reconstruction results after the dpoints consolidation.

Chapter 4

Structure-aware Data Consolidation

4.1 Introduction

We present a structure-aware filtering (SAF) method that consolidates noisy data by projecting it onto underlying, lower dimensional structures. To reveal structures in noisy inputs, SAF concentrates sample points toward latent and lower dimensional data manifolds, while maintaining an even distribution of samples across these manifolds. We achieve this by adding a regularization to the weighted data averaging in conventional mean shift [26]. A theoretical analysis under a Gaussian noise model is provided, which reveals the parameter settings needed to balance between data concentration on the manifolds and even distribution across them. Empirical experiments show that SAF can significantly boost the performance of state-of-the-art clustering and dimensionality reduction approaches.

In clustering applications, data may form arbitrary, lower dimensional structures embedded in a feature space. A general strategy to address this problem is to project the data into lower dimensional subspaces where the clusters are more apparent. Often numerous

subspaces are required, for example if each cluster manifests itself in a different subspace. The problem is more challenging, however, when the clusters form non-linear structures as demonstrated in Figure 4.1. Here each cluster has a curvy non-convex structure and the two clusters are intertwined such that they are not separated in any linear subspace. It becomes even more difficult in the presence of irrelevant features or data measurement uncertainties, which appear as noise. As shown in Figure 4.1, standard techniques such as spectral clustering or DBSCAN may fail to cluster such data.

Our structure-aware filtering technique (SAF) excels when the data forms low-dimensional structures that are contaminated by higher-dimensional noise. It is most effective when the low-dimensional manifolds are highly non-linear, like the curvy clusters in Figure 4.1, which SAF recovers succinctly (red points). After processing the data with SAF, standard clustering techniques are successful as demonstrated in Figure 4.1. A key advantage is that SAF does not require any local parametric representations of the underlying manifolds. Testing and evaluating our method on various benchmarks shows that SAF improves performance of many standard clustering techniques.
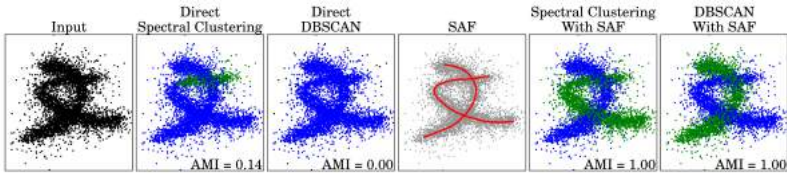
Figure 4.1: A challenging example with two intertwined clusters, corrupted with noise. After projecting the input data (left) onto the underlying structure using our structure-aware filtering (SAF) approach (red points in the middle), spectral clustering or DBSCAN (right) detect the proper clusters.

Clustering is also related to (nonlinear) dimensionality reduction, as many clustering techniques strive to construct a lower dimensional embedding of the data before clustering. In comparison, our approach

does not project the data to lower dimensional spaces directly. Instead, it projects noisy data onto lower dimensional manifolds, but each data point still maintains its high dimensional features. It can therefore be viewed as a "dimension consolidation" technique. We demonstrate that our strategy can be used as a pre-process to standard dimensionality reduction, and our consolidation indeed leads to more robust results of a number of standard clustering approaches as well.

In summary, the main contribution of this paper is to demonstrate how to boost the performance of common clustering and dimensionality reduction techniques by applying a novel structure-aware data consolidation approach as a pre-process. Our theoretical analysis proofs that, under simplifying assumptions (isotropic Gaussian noise with known variance, planar manifolds in arbitrary dimensions), the proposed structure-aware filtering converges to the underlying data manifolds. Empirical evidence shows that our analysis provides valid guidance to the selection of algorithmic parameters in practical applications.

4.2 Related Work

For an overview of research on clustering we refer the reader to recent surveys [76, 66] and discuss only selected works here. Many clustering techniques are derived from k-means clustering, including for example k-medians [22] and many others. Gaussian mixture models (GMMs) [84] and the expectation-maximization (EM) algorithm are also closely related to k-means clustering. One of the main limitations of most k-means related algorithms is their assumption that clusters exhibit simple shapes, such as isotropic (k-means clustering) or ellipsoid (GMMs) distributions. In contrast, hierarchical clustering does not require the user to specify the number of clusters and can naturally produce arbitrary cluster shapes. We refer readers to standard textbooks for an overview [49]. These techniques either proceed top-down (divisive) or bottom-up (agglomerative). DBSCAN [33] is one of the most popular agglomerative algorithms. It greedily aggre-

gates points in high density neighborhoods to clusters, which may form arbitrary shapes.

Clustering by seeking modes of an underlying density distribution is another popular approach. The best known example is the mean shift algorithm [26] and its variations. These techniques do not require the specification of the number of clusters and can also find clusters with arbitrary shapes. The underlying kernel density estimation, however, also suffers from the curse of dimensionality, which restricts them to lower dimensions.

Subspace clustering is a general strategy to work around the curse of dimensionality, and we refer the reader to Kriegel et al.'s recent survey of these techniques [76]. Dimensionality reduction maps the data to a lower dimensional space, often using non-linear techniques [114], before further processing such as visualization or clustering. Spectral clustering [126] relies on an embedding given by the spectral analysis of the similarity matrix of the data. It is highly related to dimensionality reduction techniques using Laplacian eigenmaps [8] and diffusion maps [25]. Typically, spectral clustering techniques produce their final output by applying k-means clustering after dimensionality reduction. Kannan et al. [68] provide a thorough analysis of spectral clustering using a novel quality criterion. Dhillon et al. [31] made an interesting connection between kernel k-means [117] and multiclass spectral clustering [160].

The method proposed in this paper is not a clustering or dimensionality reduction technique on its own, but it can significantly improve the performance of many approaches mentioned above by consolidating the data before clustering or embedding. Technically, our consolidation algorithm falls into the locally optimal projection (LOP) framework [85, 57, 59] in its discretized form. Huang et al. further introduce a L_1-*medial skeleton* [58] as a curve skeleton representation for 3D point clouds using such locally optimal contraction. Wu et al. augment each surface point to a *deep point* [154] by associating it with an inner point that resides on a structural mixture of skeletal curves and sheets. The common objective of these works is to seek a proper interpretation of the noisy input using a data fitting term

complemented with a repulsion term. In a similar spirit, we consider a large sample scenario and have made the first attempt on analyzing continuous densities, resulting in a convergence proof for a special case, and a structure-aware data consolidation method that greatly assists many data mining applications.

Our technique also shares similarities with manifold denoising algorithms [52, 147, 43, 149, 143, 30]. Manifold blurring mean shift [147] restricts mean shift directions to be parallel to manifold normals estimated using local PCA. Sparse subspace denoising [143] builds on sparse subspace clustering and includes a subspace reconstruction error by estimating locally linear subspaces using PCA to achieve denoising. Robust PCA [16] suppresses outliers by decomposing a highly corrupted measurement matrix into a low-rank and a sparse matrix. Hein and Maier [52] propose Manifold Denoising (MD) using a neighborhood graph Laplacian of the data. Laplacian smoothing, however, shrinks the manifold and ultimately collapses it to a single point. Hence, manual tuning of the desired amount of smoothing is in general required. Most recently, Deutsch et al. [30] propose a Manifold Frequency Denoising (MFD) algorithm by removing the high frequency bands in the spectral graph wavelet domain. It is a global method and can produce clean output when there exists only one underlying manifold. However, when the noise is severe and the underlying manifolds are nearby, the results of MFD degenerate.

4.3 Method

Our goal is to improve existing clustering and dimensionality reduction algorithms by developing a data consolidation technique as a pre-process, which we call structure-aware filtering (SAF). Here we first introduce the SAF approach (Section 4.3.1) by starting with an intuitive continuous formulation, where input and output data are modeled as continuous density functions. This facilitates a theoretical analysis that allows us to explicitly derive the behavior of SAF (Section 4.3.2). Finally, we discuss a discrete implementation (Sec-

tion 4.3.3).

4.3.1   Structure-aware Filtering

SAF consolidates noisy data densities by contracting them locally to remove noise and reveal high density structures. We first formulate SAF by modeling the noisy input data densities as continuous functions, and expressing SAF as a continuous flow given by a time dependent velocity field $v(z, t) : (\mathbb{R}^n, R) \to \mathbb{R}^n$. Denoting the input data density $f_p(z)$, we initialize a time-dependent output density $f_x(z, t)$ as $f_x(z, 0) = f_p(z)$. Then, the goal of the SAF flow is to advect $f_x(z, t)$ to gradually remove noise while revealing the underlying structures in the input density $f_p(z)$.

We model the noisy input data density $f_p(z)$ by adding noise to an underlying m-dimensional data manifold M. Let us assume the data is mapped to $\mathbb{R}^n$ via an embedding $i : M \to \mathbb{R}^n$, and we have a probability density p_M on M. Then we express the data-generating process in $\mathbb{R}^n$ as $X = i(\theta) + \epsilon$, where $\theta \sim p_M$ and we assume isotropic Gaussian noise $\epsilon \sim N(0, \sigma)$. Hence, the noisy input density is represented as

$$f_p(z) = (2\pi\sigma^2)^{-\frac{n}{2}} \int_M e^{-\frac{||z - i(\theta)||^2}{2\sigma^2}} p_M(\theta) d\theta. \qquad (4.1)$$

While noise distributions other than Gaussian could be used, we will focus on Gaussian noise in our theoretical analysis. Note that Equation (4.1) can be considered a generalization of the Gaussian latent variable model used in Probabilistic PCA [137], where θ is Gaussian and $i(\cdot)$ is linear.

The SAF velocity field consists of two components: the first one "pulls" along the gradients of the noisy input data density. This term tries to accumulate output density in local extrema of the noisy input density, and we call it the data term. The second term "pushes" output density along its negative gradients, hence we call it a repulsion term. This term makes sure that the output density does not "clump" around weak density extrema in the noisy input data density. The

repulsion term allows us to consolidate and enhance latent continuous structures in the input data, such as one-dimensional (curve) or higher-dimensional (surface) manifolds. More precisely, we define the SAF flow with the velocity field $v(z,t)$ as

$$v(z,t) = \nabla(f_p * K)(z) - \lambda(z,t)\nabla(f_x * L)(z,t).$$

Here the smoothing kernel K serves to remove noise from the input density, and L smooths the output density itself with a balancing weight function $\lambda(z,t)$. The output density $f_x(z,t)$ is time dependent, and related to the velocity field via the continuity equation

$$\frac{\partial f_x(z,t)}{\partial t} = -\nabla \cdot (f_x(z,t)v(z,t)).$$

Let us assume the smoothing kernels K and L are radially symmetric, so we can write them as $K(\xi) = k(\frac{1}{2}\|\xi\|^2)$ and $L(\xi) = l(\frac{1}{2}\|\xi\|^2)$, $\xi \in \mathbb{R}^n$. Further assuming k and l are differentiable, we have

$$\nabla K(\xi) = \xi k'(\frac{1}{2}\|\xi\|^2), \quad \text{and} \quad \nabla L(\xi) = \xi l'(\frac{1}{2}\|\xi\|^2),$$

where k' and l' are derivatives with respect to the argument $\frac{1}{2}\|\xi\|^2$. In addition, we define the weighting as

$$\lambda_x(z,t) = \mu \frac{\left(f_p(\xi) * k'(\frac{1}{2}\|\xi\|^2)\right)(z)}{\left(f_x(\xi) * l'(\frac{1}{2}\|\xi\|^2)\right)(z,t)},$$

with a global user parameter $\mu > 0$. Here ξ is the integration variable in the convolution, which we may omit in the following for clarity. After rearranging and scaling the velocity field we obtain the final SAF formulation

$$v(z,t) = \frac{f_p * \left(\xi k'(\frac{1}{2}\|\xi\|^2)\right)}{f_p * k'(\frac{1}{2}\|\xi\|^2)}(z) - \mu \frac{f_x * \left(\xi l'(\frac{1}{2}\|\xi\|^2)\right)}{f_x * l'(\frac{1}{2}\|\xi\|^2)}(z,t). \quad (4.2)$$

4.3.2 Theoretical Analysis

Given a noisy input density representing an underlying manifold, the data term of SAF attracts output density towards local maxima of the noisy input density, while the repulsion tries to maintain a smooth output density that is evenly distributed over the underlying manifold. These two terms need to be properly balanced: if the data term is too strong, data may be concentrated at isolated density modes; if the repulsion force is too strong, data may diffuse away. Here we provide a theoretical analysis of this process to understand under which circumstances SAF manages to attract density to the underlying manifold.

To make analysis tractable, we consider the special case where the smoothing kernels k and l are Gaussian, and the underlying data manifold is a hyperplane in $\mathbb{R}^n$, i.e., the noisy input data density f_p and the initial output density are degenerate Gaussians. We first analyze the one-dimensional case, and then generalize to arbitrary dimensions.

One-dimensional Case

Let $G_{0,\sigma^2}(\xi)$ denote a zero-mean, univariate Gaussian distribution with variance σ^2. We introduce the notation $g_{0,\sigma^2}(\frac{1}{2}\|\xi\|^2) = G_{0,\sigma^2}(\xi)$, and note that $g'_{0,\sigma^2} = -g_{0,\sigma^2}$. In the 1D case, the data manifold is represented by a Dirac impulse, the noisy input density is given by the 1D Gaussian $f_p = G_{0,\sigma^2}$, and the smoothing kernels are $k = l = g_{0,h^2}$. The following theorem describes how SAF converges to the noise free input (the Dirac impulse) in this scenario.

Theorem 1. *Let $f_p(z) = f_x(z,0) = G_{0,\sigma^2}(z)$, then for any $t \geq 0$ the output density is a Gaussian with some standard deviation $\omega(t)$, i.e., $f_x(z,t) = G_{0,\omega(t)}(z)$. In addition, if*

$$\sigma^2 < \frac{1-\mu}{\mu}h^2, \tag{4.3}$$

then $\omega(t) \to 0$ as $t \to \infty$. In other words, the output density converges to a Dirac impulse, i.e., the true 1D data manifold.

Proof. The initialization $f_p(z) = f_x(z,0) = G_{0,\sigma^2}(z)$ ensures that the first part of the theorem holds at $t = 0$. Assuming it is also true at time $t > 0$, we note that

$$
\begin{aligned}
\frac{f_p * (\xi \cdot k')}{f_p * k'}(z) &= \frac{G_{0,\sigma^2} * (\xi \cdot g'_{0,h^2})}{G_{0,\sigma^2} * g'_{0,h^2}}(z) \\
&= -\frac{h^2}{\sigma^2 + h^2} z, \\
\frac{f_x * (\xi \cdot l')}{f_x * l'}(z,t) &= \frac{G_{0,\omega(t)^2} * (\xi \cdot g'_{0,h^2})}{G_{0,\omega(t)^2} * g'_{0,h^2}}(z,t) \\
&= -\frac{h^2}{\omega(t)^2 + h^2} z,
\end{aligned}
$$

and hence the velocity from Equation (4.2) becomes

$$
v(z,t) = \left(-\frac{h^2}{\sigma^2 + h^2} + \mu \frac{h^2}{\omega(t)^2 + h^2} \right) z. \tag{4.4}
$$

This suggests that $v(z,t)$ corresponds to a uniform scaling of space, which means that the output density stays Gaussian at a time $> t$ and proofs the first part of the theorem by induction. To show the second part, let us denote the instantaneous scaling factor at time t as

$$
\tau(t) = 1 - \frac{h^2}{\sigma^2 + h^2} + \mu \frac{h^2}{\omega(t)^2 + h^2}. \tag{4.5}
$$

Space is monotonically contracted (scaled down) if $0 \leq \tau(t) < 1$ for all t, which is guaranteed by Equation 4.3. $\square$

As a key result, the theorem above provides a simple equation that characterizes the user parameters μ (repulsion strength) and h^2 (size of smoothing kernel) that lead to guaranteed convergence.

Hyperplanes in Arbitrary Dimensions

We generalize this analysis to the n-dimensional case by considering axis-aligned hyperplanes with uniform density $p_M \equiv 1$ as the data

manifolds[1]. Let us define an axis aligned hyperplane by the set H of coordinate axes that lie in the hyperplane. That is, H is a subset of the indices $\{1, \ldots, n\}$, and its cardinality $|H| = m$ corresponds to the dimensionality of the hyperplane. Using Equation (4.1), this leads to noisy input densities represented by degenerate, axis aligned multivariate Gaussians with zero-means, which are equivalent to products of 1D Gaussians,

$$f_p(z) = \prod_{i \in \bar{H}} G_{0,\sigma^2}(z_i),$$

where $\bar{H} = \{\{1, \ldots, n\} \setminus H\}$, z denotes an n-dimensional vector, and z_i is the i-th element in the vector.

Similar as in 1D, we assume the intermediate distribution is initialized as $f_x(z, 0) = f_p(z)$, and the smoothing kernels are $k = l = g_{0,h^2}$. In this special setting, the n-dimensional case directly reduces to the 1D case as all involved functions are separable into products of 1D functions, which means that also the convolutions are separable. As a consequence, the velocity $v_i(z, t)$ in each dimension $i \in \bar{H}$ is analogous to the 1D case in Equation (4.4),

$$v_i(z, t) = \left(-\frac{h^2}{\sigma^2 + h^2} + \mu \frac{h^2}{\omega(t)^2 + h^2} \right) z_i, \quad \text{for } i \in \bar{H}, \qquad (4.6)$$

and Theorem 1 applies to each dimension $i \in \bar{H}$ separately. On the other hand, the velocities parallel to the hyperplane are zero, $v_i(z, t) = 0$ for $i \in H$. Note that our analysis includes arbitrarily oriented hyperplanes, since we can simply rotate the coordinate system to align with the hyperplane, and then define the hyperplane as above.

---

[1] p_M cannot be considered a proper probability density in this case, but this is not an issue for our analysis.
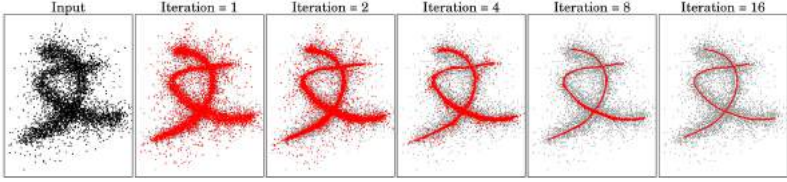
Figure 4.2: Consolidation process using anisotropic SAF.

Curved Manifolds

We can rewrite Equation (4.2) as

$$v(z,t) = \underbrace{(1-\mu)(i(\theta_{\min}) - z)}_{\text{(I)}}$$

$$-\underbrace{\left(i(\theta_{\min}) - \frac{f_p(\xi)\xi * k'(\frac{1}{2}\|\xi\|^2)}{f_p(\xi) * k'(\frac{1}{2}\|\xi\|^2)}(z)\right)}_{\text{(II)}}$$

$$+\mu\underbrace{\left(i(\theta_{\min}) - \frac{f_x(\xi)\xi * l'(\frac{1}{2}\|\xi\|^2)}{f_x(\xi) * l'(\frac{1}{2}\|\xi\|^2)}(z,t)\right)}_{\text{(III)}}. \qquad (4.7)$$

where $i(\theta_{\min}) = \arg\min_{i(\theta)} \|z - i(\theta)\|$ denotes the closest point to z on the data manifold M. The first term (I) represents motion towards the manifold M, as desired. Hein et al. [52] show that the second term (II) approximates $-mH - \frac{2}{p_M}\langle\nabla p_M, \nabla i\rangle$, where H is the mean curvature normal of M. The mean curvature term here smooths and shrinks the manifold, and the gradient of the data density ∇p_M leads to "clumping". Both these undesirable effects can be observed in practice.

In contrast, the repulsion in SAF is represented by a third, similar term (III), but with opposite sign compared to (II), and for an evolving manifold f_x. Hence repulsion in SAF counteracts the mean curvature smoothing and shrinkage, and it leads to more uniform densities on
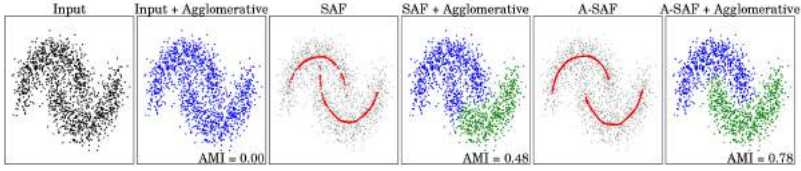
Figure 4.3: Clustering without (SAF) and with anisotropic repulsion (A-SAF).

the manifold due to tangential diffusion. In practice (Section 4.3.3), our approach converges to stable structures without collapsing. In addition, if we choose the parameters according to Theorem 1, we obtain thin manifolds without noise. Nonetheless, the analysis provided here can only give an intuition; a thorough proof for non-linear cases is left for future work.

4.3.3   Discrete SAF with Anisotropic Repulsion

We implement a discretized version of SAF following a Lagrangian approach, i.e., we represent densities by sets of sample points. The input density f_p is given by points $\{p_j\}$, and f_x by points $\{x_i(t)\}$. Then the data term from Equation (4.2) (without normalization) is

$$f_p * \left(\xi k'(\frac{1}{2}\|\xi\|^2)\right)(z) = \sum_j (p_j - z) k'(\frac{1}{2}\|p_j - z\|^2).$$

In addition, let us generalize the continuous formulation from Equation (4.2) to anisotropic kernels for repulsion, which will allow for more effective repulsion in practice as discussed below. We implement an anisotropy by including a matrix A to linearly deform the repulsion kernel, that is $L(A\xi) = l(\frac{1}{2}\|A\xi\|^2)$. This leads to the generalized

repulsion term and its discretized form

$$f_x * \left(A^T A \xi k'(\frac{1}{2}\|A\xi\|^2) \right)(z)$$

$$= \sum_j A^T A(p_j - z)k'(\frac{1}{2}\|A(p_j - z)\|^2).$$

Now we evaluate the velocity field only at the sample points $\{x_i(t)\}$, and advect the samples with unit time steps. Their updated positions $\{x_i(t+1)\}$ in the next time step immediately define f_x for the next iteration. For simplicity of notation, denote $x_i = x_i(t)$ and $x_i' = x_i(t+1)$. Then the discretized version of SAF defined at each sample point is

$$x_i' = x_i + \frac{\sum_j (p_j - x_i)k'(\frac{1}{2}\|p_j - x_i\|^2)}{\sum_j k'(\frac{1}{2}\|p_j - x_i\|^2)}$$
$$- \mu \frac{\sum_{i'} A^T A(x_{i'} - x_i)l'(\frac{1}{2}\|A(x_{i'} - x_i)\|^2)}{\sum_{i'} l'(\frac{1}{2}\|A(x_{i'} - x_i)\|^2)}$$
$$= \frac{\sum_j p_j k'(\frac{1}{2}\|p_j - x_i\|^2)}{\sum_j k'(\frac{1}{2}\|p_j - x_i\|^2)}$$
$$- \mu \frac{\sum_{i'} A^T A(x_{i'} - x_i)l'(\frac{1}{2}\|A(x_{i'} - x_i)\|^2)}{\sum_{i'} l'(\frac{1}{2}\|A(x_{i'} - x_i)\|^2)}.$$

The motivation behind the anisotropic repulsion is that, if the data forms a lower dimensional structure embedded in a higher dimensional space, we would like to direct repulsion to move points around on the structure itself [58, 154]. In practice, for each output point x_i we perform a PCA analysis on its k nearest neighbors and get the corresponding eigenvectors $\{v_i^1, v_i^2, \cdots, v_i^n\}$ and eigenvalues $\{\lambda_i^1, \lambda_i^2, \cdots, \lambda_i^n\}$, where n is the dimension of the input data. We denote the $n \times n$ column matrix $A_i = [\lambda_i^1 v_i^1; \lambda_i^2 v_i^2; \cdots; \lambda_i^P v_i^P]$, and use it to adjust the shape of the repulsion kernel. Note that our analysis from Section 4.3.2 also applies to anisotropic repulsion. Anisotropic repulsion simply means that the variances h_i in the repulsion term in
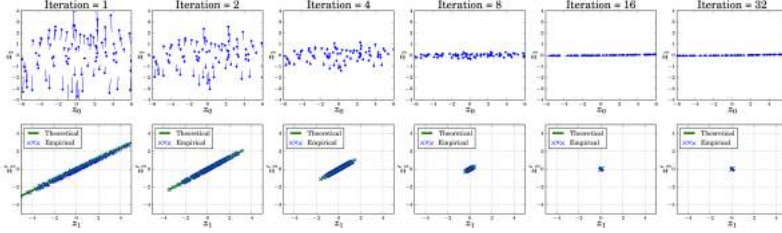
Figure 4.4: We illustrate the iterative consolidation process in a 2D example with a data distribution $f_p = G_{0,(\infty,4)}$. The user parameters are $h = 4$ and $\mu = 0.5$, which leads to convergence according to Equation (4.3). The first row shows the actual point movements where the dots are current positions and the vectors are pointing to the next locations. The second row shows both empirical and theoretical (Equation (4.6)) update ratios of z_1'/z_1.
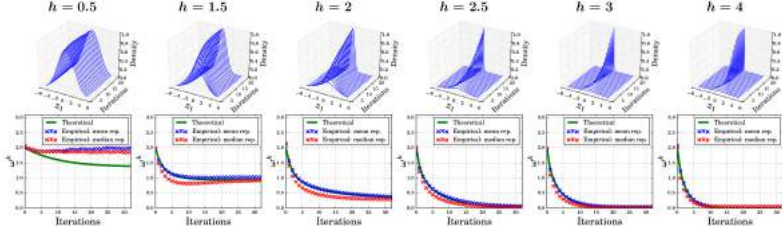

Figure 4.5: Top row: we illustrate changes of point densities during the iterative consolidation process with different h values. Bottom row: empirical estimate and theoretical prediction of the variances ω of the intermediate point distributions. We show results using mean and median repulsion with blue and red crosses, respectively. While theoretical analysis with median repulsion is difficult, it empirically follows our prediction.

Equation (4.6) are scaled with the PCA eigenvalue along the corresponding coordinate axis.

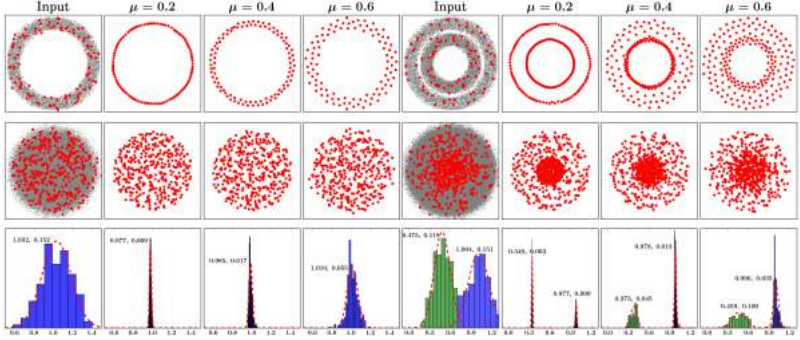Figure 4.2 illustrates the consolidation process using anisotropic

Figure 4.6: Convergence in different dimensions and with median repulsion. We add Gaussian noise ($\sigma = 0.15$) to 2D circles (first row, single circle and two concentric ones) and 4D spheres (second row, single sphere and two concentric ones). We set the kernel size $h = 0.1$, and show results with different μ values. Equation (4.3) predicts convergence for $\mu < 0.31$. The third row shows histograms of distances to the center of the 4D spheres. The values next to the red bell curves are their empirical means and variances, demonstrating convergence to thin manifolds as predicted for $\mu < 0.31$.

SAF. The anisotropic repulsion force mainly pushes points along the local major PCA directions, which improves the regularity of data distribution and eventually can lead to better clustering, as demonstrated in Figure 4.3.

Kernel Selection

The kernel k for the data term should be smooth to eliminate noise in the input density, hence we use a (multidimensional) Gaussian $k(\frac{1}{2}\|\xi\|^2) = g_{0,h^2}(\frac{1}{2}\|\xi\|^2)$. The kernel l for the repulsion term should have large derivatives around the origin, such that close-by points are effectively pushed away from each other. Therefore, in practice we

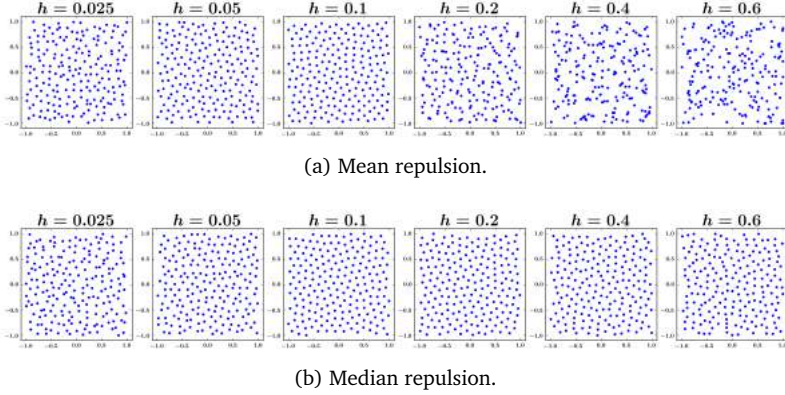(a) Mean repulsion.

(b) Median repulsion.

Figure 4.7: We compare mean (a) and median (b) repulsion. We regularize a set of 2D points using only repulsion without data term, and with periodic boundary conditions. We compare the results with different h values. While both mean and median repulsion have similar convergence behavior, median repulsion is more robust and less sensitive to the neighborhood size h, which helps to generate locally uniform distributions.

also use a modified repulsion kernel defined by its derivative

$$l'(\tfrac{1}{2}\|\xi\|^2) = \begin{cases} -g_{0,h^2}(\tfrac{1}{2}\|\xi\|^2)/\|\xi\| & \xi \neq 0, \\ 0 & \xi = 0. \end{cases} \qquad (4.8)$$

In the discrete setting, using Gaussian repulsion kernels means that the repulsion term vanishes if each output point x_i minimizes a locally weighted sum of square distances, i.e., if each point is the mean of its locally weighted neighbors. We refer to it as "*mean repulsion*". With modified repulsion kernels defined in Equation (4.8), the repulsion term vanishes if each point minimizes a locally weighted sum of absolute distances, i.e., if each point is the median of its locally weighted neighbors. We refer to this as "*median repulsion*".

(a) 2D plane + 4D noise.
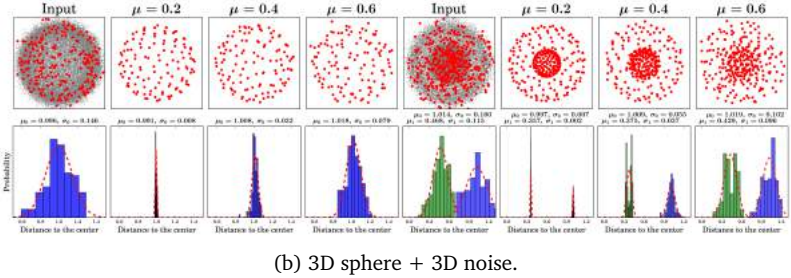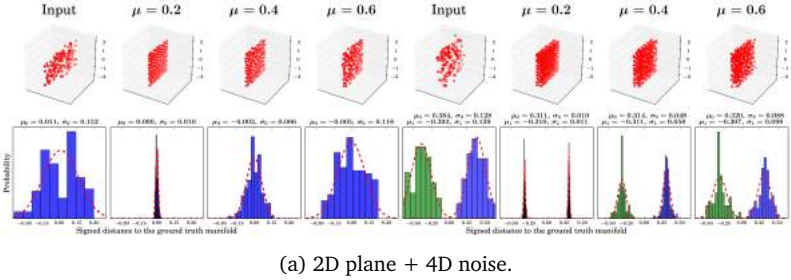

(b) 3D sphere + 3D noise.

Figure 4.8: Our theory can predict the convergence for high dimensional data. We add Gaussian noise with $\sigma = 0.15$ to 2D planes (a) and 3D spheres (b), set the kernel size $h = 0.1$, and show the results with different μ values applied. Our theoretical analysis predicts the convergence when $\mu < 0.31$. In each experiment, we test two types of input, i.e., single and double manifolds. At the bottom rows, we also demonstrate histograms of distances to the center of 2D planes and 3D spheres.

Empirical Validation

We empirically validate the theoretical results, that is, the velocity from Equation (4.6) and the convergence criterion from Equation (4.3), in a 2D setup with axes (z_0, z_1), following the notation in Equation (4.6). The input points are uniformly distributed along z_0, and normally distributed along z_1, that is, $\sigma_0 = \infty, \sigma_1 = 2$ (sub-
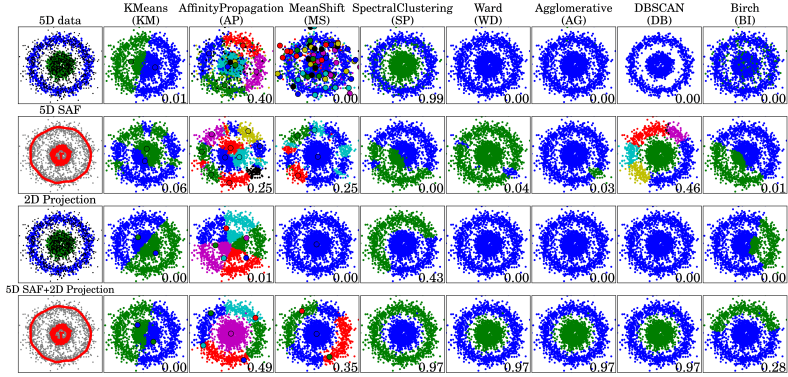
Figure 4.9: Dimensionality reduction and clustering: the input consists of two concentric 2D circles corrupted with 5D noise (the black dots). We elevate 2D rings to 5D by appending zeros, and then add standard Gaussian noise. The leftmost column shows ground truth labeling (green and blue, outliers black), the consolidated points (red) and the input points (gray). We use PCA to project the 5D data to 2D. We compare four strategies (from the top row to the bottom), "direct clustering", "consolidation + clustering", "projection + clustering" and "consolidation + projection + clustering", with different clustering algorithms (from left to right). The second row reveals the sensitivity of most clustering techniques to higher dimensional data, which is not clustered well even though the consolidation exposes the structure of the data. The third row shows that reducing the dimensionality of the data does not solve the problem due to the noisy data. In general, the "consolidation + projection + clustering" strategy in the bottom row gives the best performance (AMI scores in bottom right of subfigures). Some techniques (k-means, affinity propagation, mean shift) are not suitable to cluster this type of data, and they do not benefit from consolidation.

script indices are dimensions as in Equation (4.6)) and $f_p = G_{0,(\infty,4)}$. This is a degenerate 2D Gaussian modeling a 1D line in 2D. We set the

repulsion strength to $\mu = 0.5$. According to Equation (4.3) we need $h^2 > 4$ (omitting the subscript index 1 for clarity) for convergence. We visualize the convergence process for $h = 4$ in Figure 4.4, where mean repulsion is used.

In Figure 4.5 we show evolution of the point density over the iterations for different values of h^2. This shows that for $h^2 < 4$ the distribution fails (or stops) to contract because of the repulsion term. For $h^2 > 4$ the distribution continuously sharpens at a rate that is well predicted by the theory. Deviations of empirical behavior from the theory can be explained by the fact that the discrete point sets do not exactly correspond to continuous Gaussian distributions.

Figure 4.6 illustrates that our theory well predicts the convergence behavior with median repulsion for data in different dimensions. While there is some shrinkage due to the curved manifolds, SAF converges to stable structures because of repulsion. In Figure 4.7 we also compare empirical results for mean and median repulsion, which generates more uniform point distributions in practice.

Comparison with Mean Shift and LOP

The SAF data term is equivalent to mean shift, which accumulates output density at local extrema of the input. We are not interested in finding modes of the input, however. Instead, we want to produce an output density that removes noise from the input and consolidates and reveals its continuous structures. For this, the repulsion term is crucial.

The discrete formulation reveals that SAF is also a generalization of LOP operators [85, 57, 59], where the LOP weights correspond to the derivatives of certain radial kernels k and l. LOP with isotropic repulsion based on Euclidean distances [57] corresponds to median repulsion (Equation 4.8), while anisotropic SAF puts more effort on revealing and consolidating continuous high density structures in the underlying noisy data; see e.g., Figures 4.2 and 4.3.

4.4   Results

Here we discuss the application of our approach to dimensionality reduction and clustering.
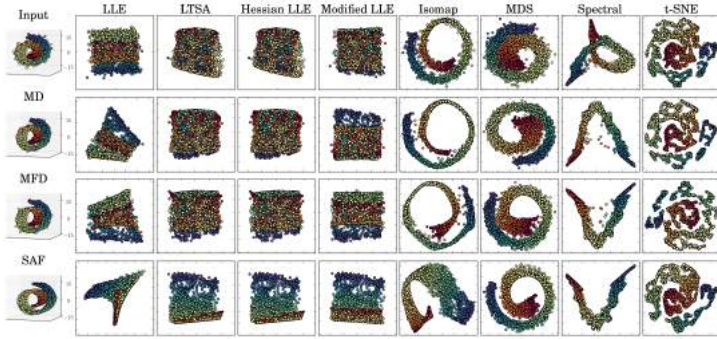
4.4.1   Dimensionality Reduction

Dimensionality reduction, or manifold learning, is an indispensable tool for data analysis, such as visualization or clustering. Existing techniques, however, often suffer from noise present in the high dimensional data. Our SAF can serve as a dimension consolidation tool that removes noise in high dimensional space, which greatly improves the performance of subsequent dimensionality reduction. In Figure 4.10 we test some of the most common dimensionality reduction methods with and without SAF consolidation.

We also compare SAF with two recent manifold denoising methods, Manifold Denoising (MD) [52], and Manifold Frequency Denoising (MFD) [30]. The resulting 2D embeddings show that the intrinsic shape of the data can be best preserved when the data is consolidated with SAF before dimensionality reduction.
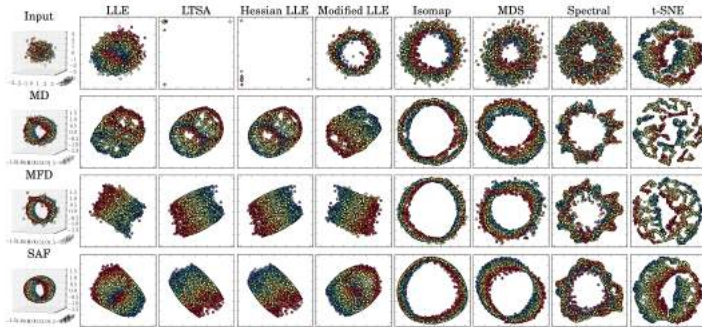
4.4.2   Clustering

We evaluate our method by comparing the performance of selected clustering techniques with and without our data consolidation approach. As baseline techniques we selected clustering algorithms that are commonly used, widely available with source code, and representative for various clustering strategies: KMeans clustering (KM) [5], Affinity propagation (AP) [35], Mean Shift clustering (MS) [26], Spectral clustering (SP) [141], Ward clustering (WD) [97], Agglomerative clustering (AG) [34], DBSCAN clustering (DB) [33], and Birch clustering (BI) [163], all implemented in the scikit-learn library [104]. In each experiment, we tune the parameters for all the selected algorithms to achieve optimal consolidation results, and we compare results with and without our data consolidation approach, see Fig-

(a) 3D Swiss roll to 2D.
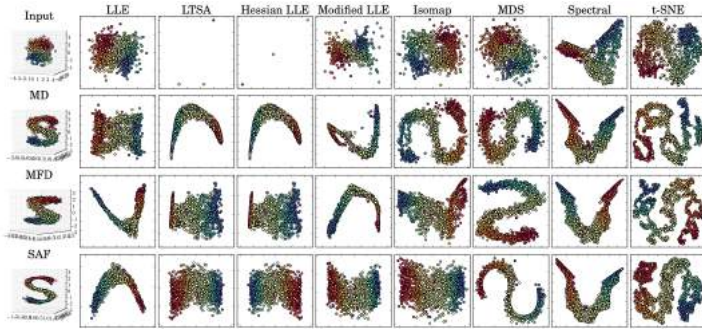
(b) 6D to 2D: We lift an 'S' shape to 6D, add noise, and project back to 2D.

(c) 10D to 2D: We lift a ring from 3D to 10D, add noise, and project to 2D.

Figure 4.10: Performance of dimensionality reduction without (top) and with MD [52] (second row), MFD [30] (third row), and SAF (bottom) consolidation.

ure 4.11 for an example. For each clustering algorithm, however, we use the same parameters regardless of using consolidation or not. When the ground truth labeling is given, we compute the Adjusted Mutual Information (AMI) to evaluate the clustering results. Note that we exclude the extremely noisy points (depicted as small black dots in the figures) from the calculation of AMI scores, because assigning ground truth labels for those points could be very ambiguous.
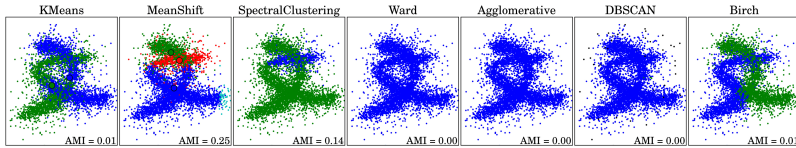
(a) Direct clustering.
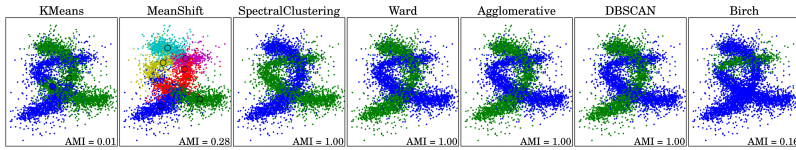
(b) Clustering with SAF.

Figure 4.11: Improved performance of additional clustering techniques on the data from Figure 4.1 in the paper. Directly using standard clustering techniques (a) may generate incorrect results when the clusters are intertwined and corrupted with noise. By projecting data points to the lower dimensional curve structures and filtering out the noise, we significantly improve clustering performance (b). Some techniques, such as k-means and mean shift clustering, do not profit from our approach, however, because they inherently cannot detect clusters with such elongated, curvy shapes. We also provide the adjusted mutual information (AMI) of the clustering results.

Dimensionality Reduction and Clustering

Many clustering algorithms cannot cope with high dimensional data well. In Figure 4.12, although our method can successfully clean up high dimensional noise and expose the low dimensional structure, the clustering algorithms do not benefit from the consolidation because the consolidated points remain in high dimension. Projecting the input data directly to a lower dimensional space often does not solve the problem if the data is noisy. Once we project the consolidated data into a lower dimensional space, however, the improvement of clustering is significant. More examples are shown in Figure 4.13.

This suggests that our consolidation method neither stops working in high dimensional space, nor does it solve the high dimensionality problem on its own. Consolidation using SAF followed by dimensionality reduction and finally clustering, however, is an effective scheme, as shown in Figure 4.14.

Different Dimensionalities

In Figure 4.16 we investigate how our consolidation performs with increasing dimensionality. We test on data consisting of two concentric hyperspheres with different radii, corrupted by Gaussian noise. We keep the radii, the number of sample points, and the noise level constant independent of the data dimensionality. Note that we do not apply any dimensionality reduction in this experiment.

As shown in Figure 4.15 and 4.16, the clusterings degrade as the dimensionality increases up to 6D. One reason is that many clustering algorithms themselves cannot handle high dimensional data well. And, in higher dimensions, the data points become sparser, which inevitably affects the robustness of our consolidation. In other words, our method requires denser data points or a better neighborhood definition (distance metric) as the dimensionality increases. We also observe similar effects with MD [52] and MFD [30]. In general, however, SAF performs better in high dimensional cases, as shown in Figure 4.17.
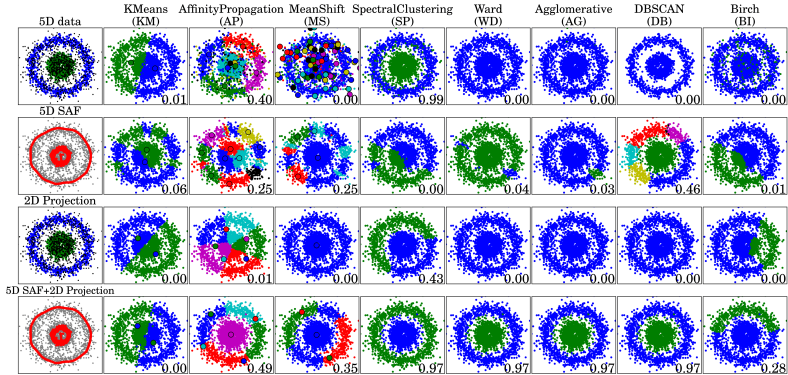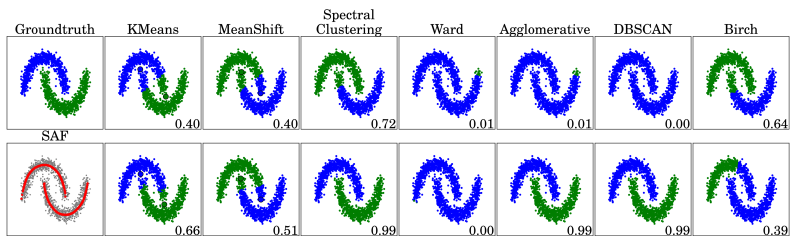
Figure 4.12: Dimensionality reduction and clustering: the input consists of two concentric 2D circles corrupted with 5D noise (the black dots). We elevate 2D rings to 5D by appending zeros, and then add standard Gaussian noise. The leftmost column shows ground truth labeling (green and blue, outliers black), the consolidated points (red) and the input points (gray). We use PCA to project the 5D data to 2D. We compare four strategies (from the top row to the bottom), "direct clustering", "consolidation + clustering", "projection + clustering" and "consolidation + projection + clustering", with different clustering algorithms (from left to right). The second row reveals the sensitivity of most clustering techniques to higher dimensional data, which is not clustered well even though the consolidation exposes the structure of the data. The third row shows that reducing the dimensionality of the data does not solve the problem due to the noisy data. In general, the "consolidation + projection + clustering" strategy in the bottom row gives the best performance (AMI scores in bottom right of subfigures). Some techniques (k-means, affinity propagation, mean shift) are not suitable to cluster this type of data, and they do not benefit from consolidation.

(a) 2D moons + 5D noise. Gaussian noise with $\sigma = 0.03$, kernel size $h = 0.055$, repulsion $\mu = 0.4$.
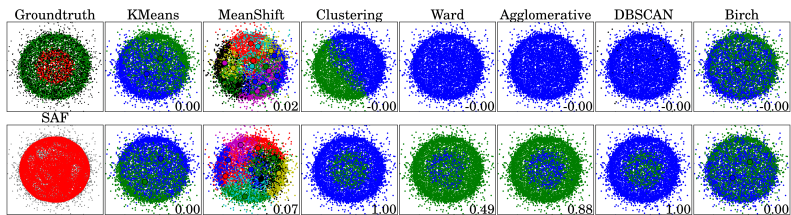
(b) 3D spheres + 6D noise. Gaussian noise with $\sigma = 0.04$, kernel size $h = 0.05$, repulsion $\mu = 0.2$.
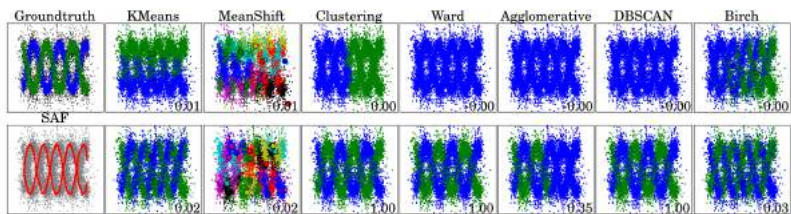
(c) 3D spirals + 10D noise. Gaussian noise with $\sigma = 0.025$, kernel size $h = 0.04$, repulsion $\mu = 0.25$.

Figure 4.13: Additional clustering examples of low dimensional structures corrupted with high dimensional noise. Top row: clustering without consolidation; bottom row: with consolidation. Leftmost column: consolidated data in red. Here the kernel size h is relative to bounding box diagonal. The consolidation outputs are projected to 3D via PCA, followed by clustering in 3D.
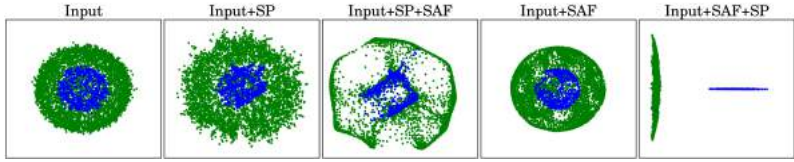
(a) Visualization of embeddings using ground truth segmentation for color coding.
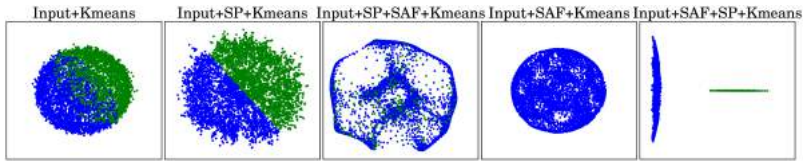
(b) Visualization of different clustering strategies.

Figure 4.14: We illustrate how spectral clustering, implemented via spectral embedding followed by k-means clustering, benefit from our data consolidation technique. The input data consists of two noisy, concentric 3D spheres with different radii corrupted with 6D noise. The top left shows an orthogonal projection of the input data to 2D. We compare five strategies for preprocessing and embedding the data before clustering (from left to right; the color codings in the top row show the ground truth clusters, the second row shows actual clustering results). All strategies rely on k-means clustering as their last step, and the results are shown in the bottom row. The first column, "Input" applies k-means directly to the input data; "Input+SP" is traditional spectral clustering, i.e., embedding using two eigenvectors of the affinity matrix followed by k-means; "Input+SP+SAF" uses the same spectral embedding, but consolidates before k-means clustering; "Input+SAF" consolidates and applies k-means both in 6D; "Input+SAF+SP" consolidates in 6D, and then uses a spectral embedding, following with k-means clustering. The experiment shows that the spectral embedding is sensitive to noise, and without our consolidation (second column from left) it fails to separate the clusters. In contrast, consolidation followed by spectral embedding (rightmost column) clearly separates the clusters and leads to correct results. Clustering in 6D without spectral embedding (first and fourth column from left) fails because k-means cannot separate the spherical shells in 6D.

(a) 2D hyperspheres. Kernel size $h = 0.034$, repulsion $\mu = 0.45$.

(b) 3D hyperspheres. Kernel size $h = 0.035$, repulsion $\mu = 0.4$.

(c) 4D hyperspheres. Kernel size $h = 0.024$, repulsion $\mu = 0.35$.
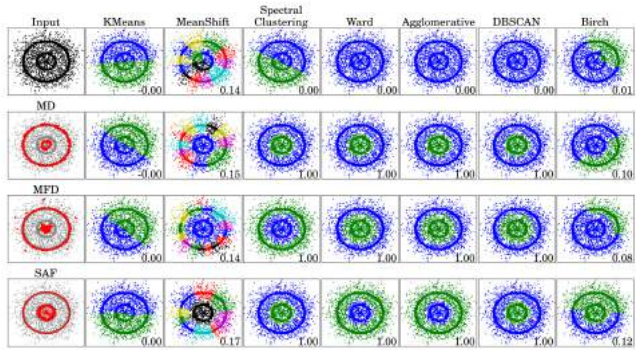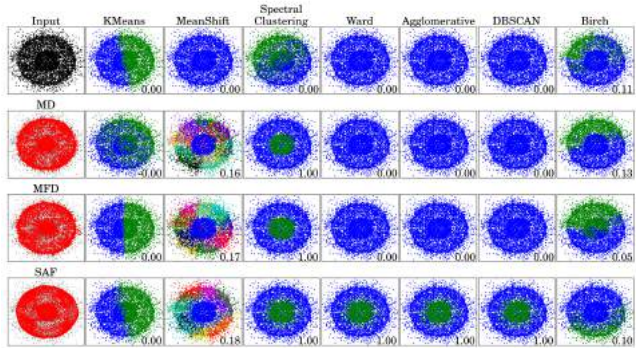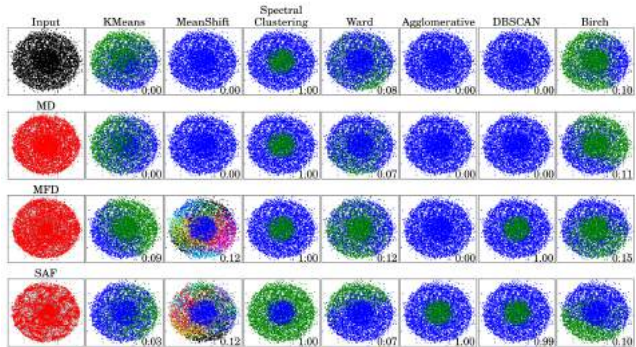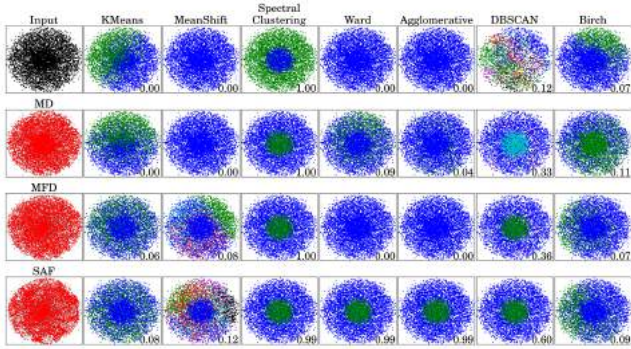
(a) 5D hyperspheres. Kernel size $h = 0.03$, repulsion $\mu = 0.2$.
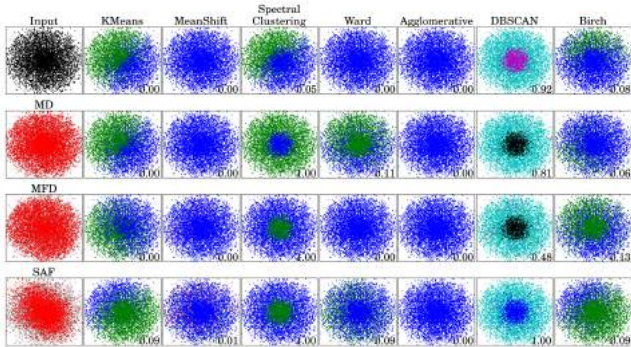
(b) 6D hyperspheres. Kernel size $h = 0.011$, repulsion $\mu = 0.15$.

Figure 4.15: SAF performance with increasing dimensionality of the data. The top and bottom rows show clustering without and with consolidation, respectively. SAF improves the performance of approaches including spectral clustering, Ward, agglomerative, and DBSCAN as shown, but its effectiveness decreases with data dimensionality increasing. We always use Gaussian noise with $\sigma = 0.05$ (relative to bounding box diagonal), and cluster without dimensionality reduction.
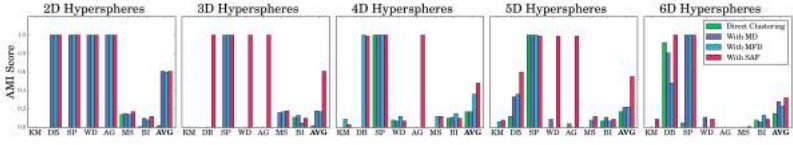
Figure 4.16: Performance of SAF with increasing dimensionality, compared with MD [52] and MFD [30]. The data consists of two concentric hyperspheres with different radii, corrupted with Gaussian noise, as shown in Figure 4.15. The rightmost bars "AVG" show the average over all clustering methods.
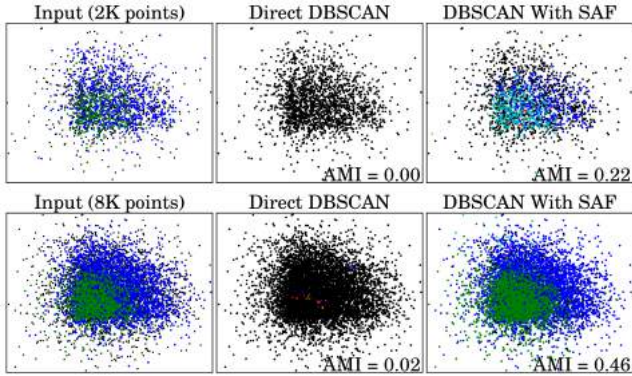
Figure 4.17: We illustrate the performance of our consolidation with data density increasing. The data consists of two patches of two 6D concentric hyperspheres with different radii and corrupted with Gaussian noise. The two rows show clustering on sparser and denser inputs, respectively. As the dimensionality increases, the denser input is required to realize the effectiveness of consolidation. We use Gaussian noise $\sigma = 0.05$, kernel size $h = 0.024$ (relative to bounding box diagonal), $\mu = 0.25$, anisotropic SAF, and cluster in 6D without dimensionality reduction.

Different Noise Levels

In Figure 4.18, 4.19, 4.20 and 4.21 we evaluate the performance of consolidation under different input noise levels. We use the data generator in the scikit-learn library to generate test data with desired Gaussian noise. As shown in the figure, our consolidation approach can substantially improve the clustering performance under a wide range of input noise levels. MD [52] and MFD [30] also improve the clustering performance under low noise levels, but SAF better preserves underlying structures when noise levels are high.

Different Embedding Spaces

As discussed above, we found that dimensionality reduction is important when dealing with high dimensional data. In the experiment in Figure 4.22 and 4.23 we investigate the influence of using different embedding spaces before clustering. We test on the MINST data set and project the 96 dimensional input data into 3D. Results suggest that using different dimensionality reduction techniques will not make a big impact on our consolidation method, as long as the underlying structure can be preserved in the embedding space. While MD [52] and MFD [30] also improve clustering performance, SAF shows an overall performance advantage.

Different Target Cluster Numbers

We test the clustering performance for different target cluster numbers using the extended Yale Face Dataset B [38]. This dataset contains 38 individuals and around 64 frontal images. We randomly selected 2, 4, 6 ,8 and 10 individuals from the dataset and report the AMI scores in Figure 4.25 and 4.24. The original face image is 1024 dimensions, which is projected onto a 9D affine subspace via PCA. The subspace is constructed by randomly selecting 1900 training images from the dataset. This pre-processing step was adapted and justified by Wang et al. [148]. In average, our method can best improve the clustering performance for different numbers of clusters compared to MD [52]
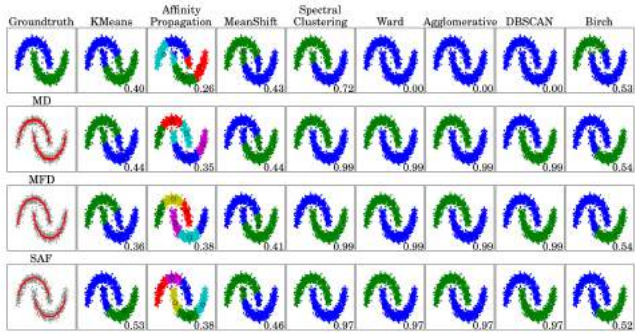
(a) Noise $\sigma = 0.03$, kernel size $h = 0.047$, repulsion $\mu = 0.4$.
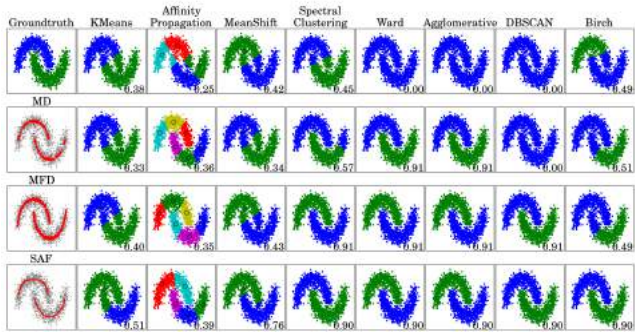
(b) Noise $\sigma = 0.045$, kernel size $h = 0.052$, repulsion $\mu = 0.4$.
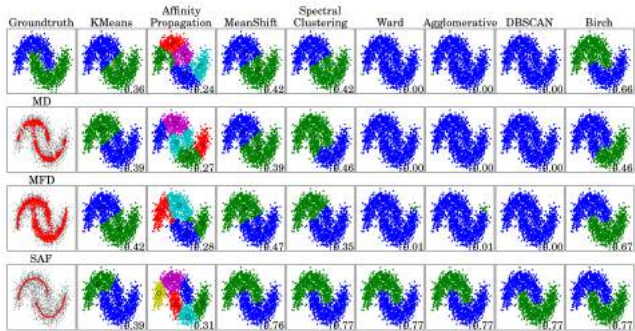
(c) Noise $\sigma = 0.06$, kernel size $h = 0.053$, repulsion $\mu = 0.35$.

(a) Noise $\sigma = 0.075$, kernel size $h = 0.057$, repulsion $\mu = 0.4$.

(b) Noise $\sigma = 0.09$, kernel size $h = 0.073$, repulsion $\mu = 0.35$.

Figure 4.18: We evaluate our approach for different noise levels using 2D data. Top row: clustering without SAF consolidation; bottom row: with SAF consolidation. We use anisotropic SAF and cluster in 2D. All σ and h values relative to bounding box diagonal.

(a) Noise $\sigma = 0.035$, kernel size $h = 0.039$, repulsion $\mu = 0.5$.

(b) Noise $\sigma = 0.053$, kernel size $h = 0.05$, repulsion $\mu = 0.35$.

(c) Noise $\sigma = 0.071$, kernel size $h = 0.056$, repulsion $\mu = 0.25$.
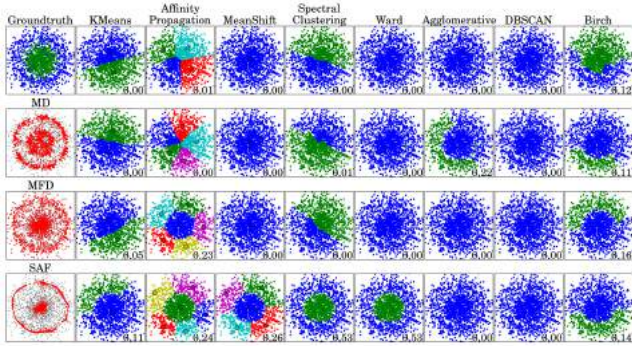
(a) Noise $\sigma = 0.088$, kernel size $h = 0.063$, repulsion $\mu = 0.2$.
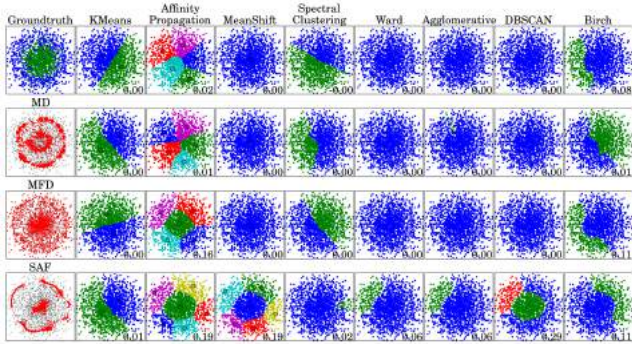
(b) Noise $\sigma = 0.11$, kernel size $h = 0.069$, repulsion $\mu = 0.15$.

Figure 4.19: Comparison of different noise levels using 2D concentric circles, similar to Figure 4.18.
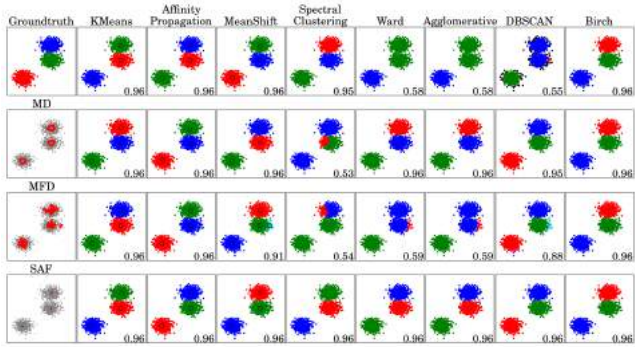
(a) Noise $\sigma = 0.063$, kernel size $h = 0.048$, repulsion $\mu = 0.3$.
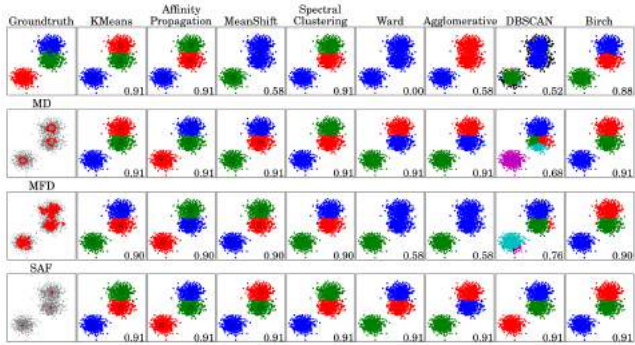
(b) Noise $\sigma = 0.079$, kernel size $h = 0.04$, repulsion $\mu = 0.2$.
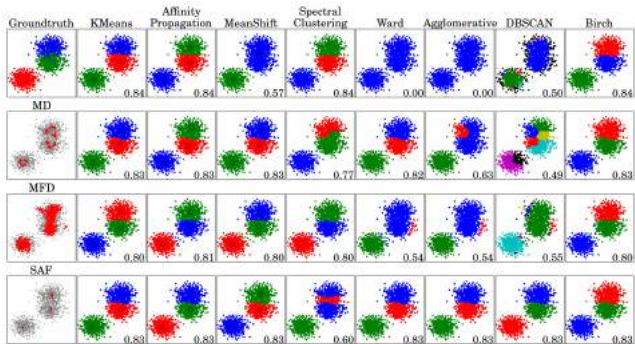
(c) Noise $\sigma = 0.094$, kernel size $h = 0.045$, repulsion $\mu = 0.2$.

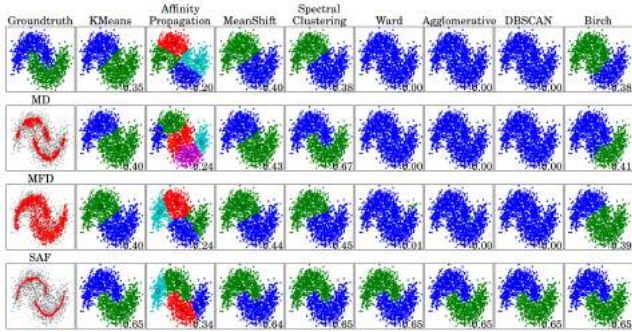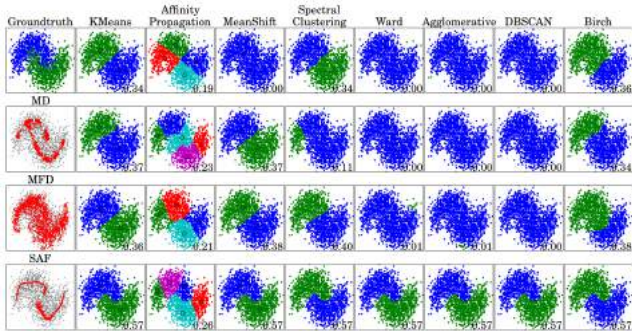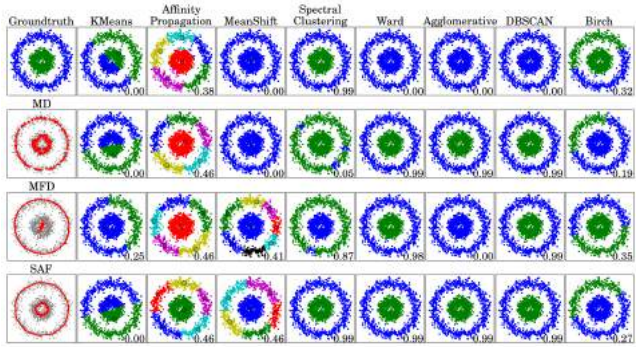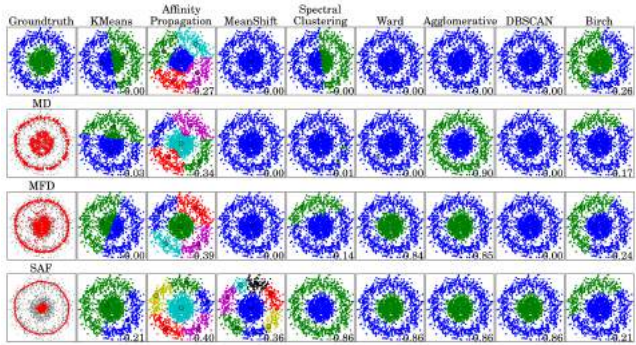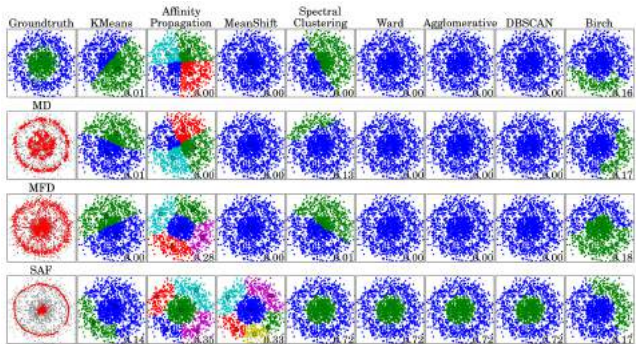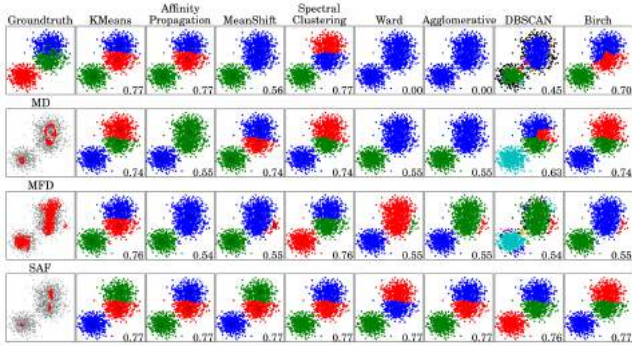(a) Noise $\sigma = 0.11$, kernel size $h = 0.046$, repulsion $\mu = 0.1$.
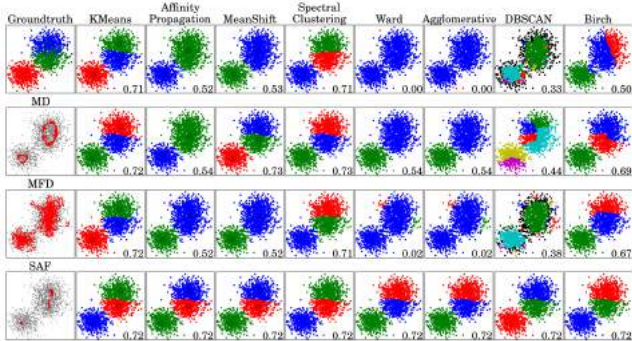
(b) Noise $\sigma = 0.13$, kernel size $h = 0.047$, repulsion $\mu = 0.05$.

Figure 4.20: Comparison of different noise levels using 2D blobs, similar to Figure 4.18.
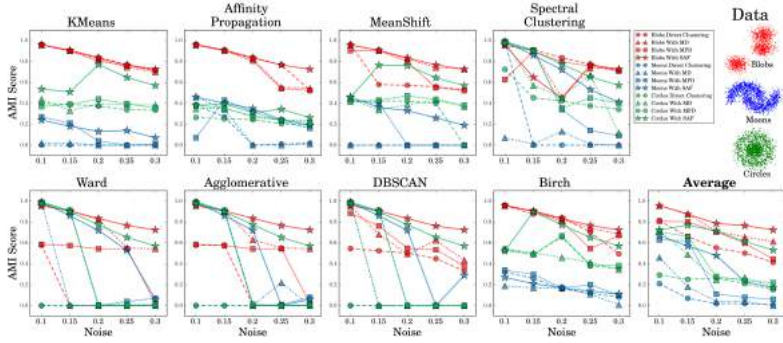
Figure 4.21: Performance under different noise levels and for different datasets, as shown in Figure 4.18, 4.19 and 4.20. We compare the clustering scores of our SAF consolidation to those of direct clustering, MD [52] and MFD [30]. SAF performs the best, especially when the noise level is high where the structures are better preserved by SAF.

and MFD [30]. All three methods work best for clustering only two groups of faces, where the underlying structure can be more easily found.

Parameter discussion There are two main parameters of our method, h and μ. We use the average sparsity r of input points to determine the h value. That is, we compute r as the average Euclidean distance to the closest neighbor. As shown in Figure 4.26 and 4.27, using a big h value can better deal with the noise and outliers, but clusters could be mistakenly merged if a too big h value is used. On the other hand, using a small h value can better preserve the shape of the structure, but less robust to the noise and may lead to less than satisfactory data distribution. The parameter μ has similar effects. Using a big μ value usually gives more regular data distribution, but makes the convergence of the optimization slower. Using a small μ value could make it a bit more resilient to noise but may cause disconnection in the same structure. In general, if the noisy level is high, a big h and
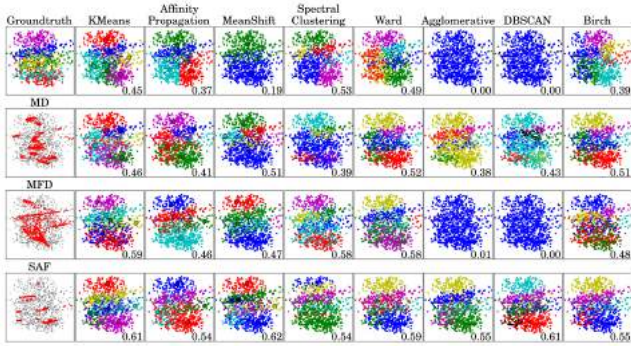
(a) PCA. $h = 0.035$, $\mu = 0.3$.
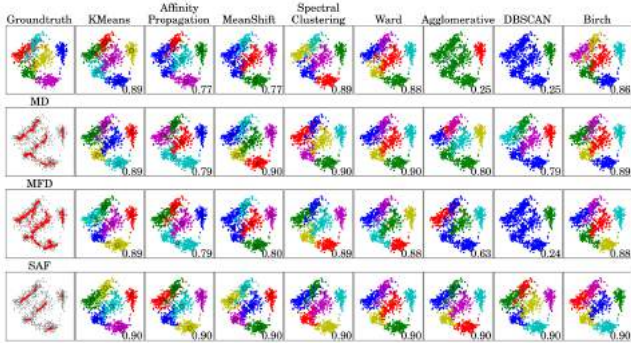
(b) ISO. $h = 0.022$, $\mu = 0.45$.
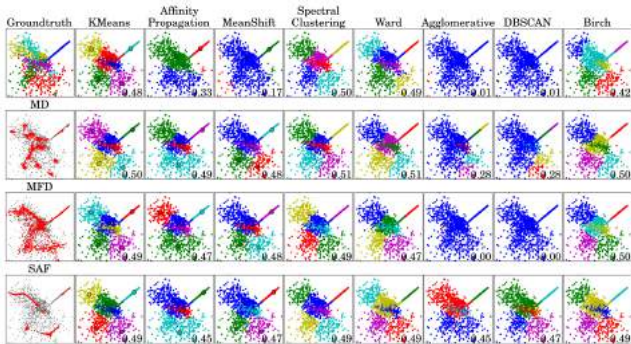
(c) LLE. $h = 0.015$, $\mu = 0.45$.
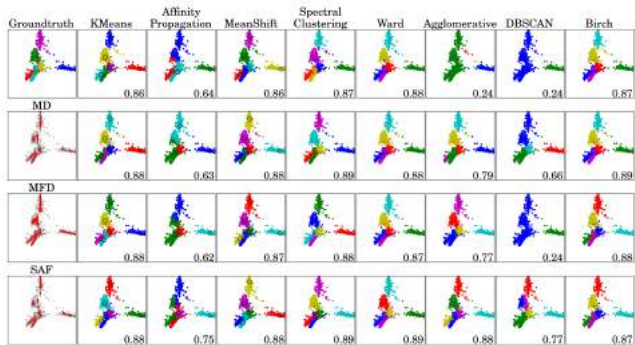
(a) Spectral. $h = 0.008$, $\mu = 0.3$.
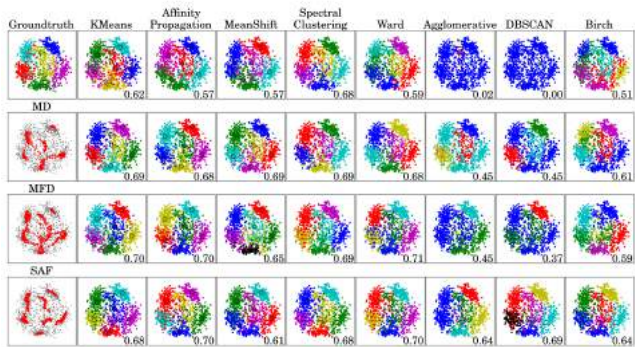
(b) MDS. $h = 0.054$, $\mu = 0.2$.

Figure 4.22: Clustering the real MINST data with different 3D embedding spaces. Top row: clustering without SAF consolidation; bottom row: clustering with SAF consolidation. All h values relative to bounding box diagonal. We perform SAF in 3D before clustering.
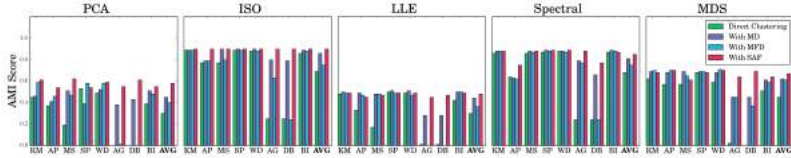
Figure 4.23: Clustering the MINST data with different 3D embedding spaces using PCA, isomap [135], LLE [114], spectral [8], and MD-S [13]. The AMI scores suggest that here agglomerative clustering (AG) and DBSCAN (DB) benefit most from consolidation, whereas spectral clustering does not.
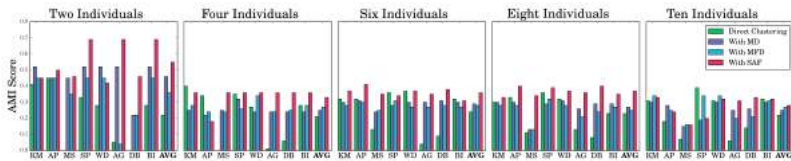
Figure 4.24: Clustering the Yale face data. The AMI scores show the benefit of SAF in almost all cases.
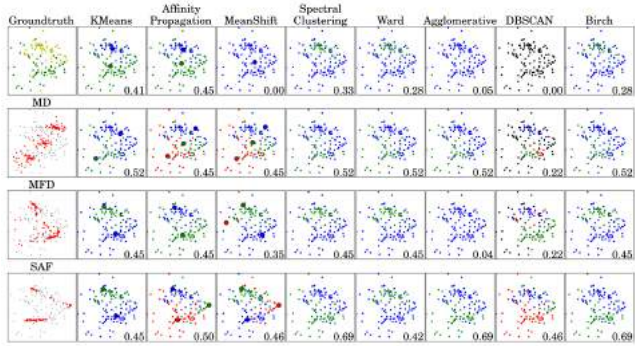
(a) Two Individuals. $h = 0.06$, $\mu = 0.25$.
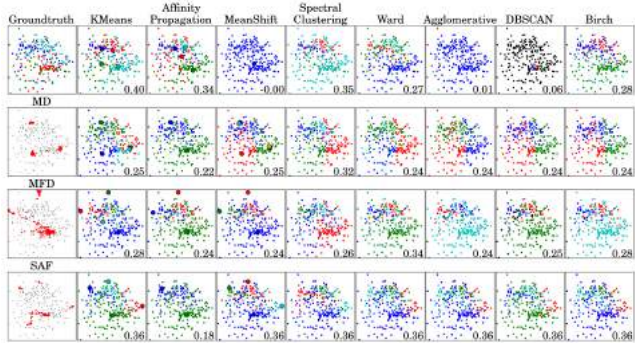
(b) Four Individuals. $h = 0.047$, $\mu = 0.15$.
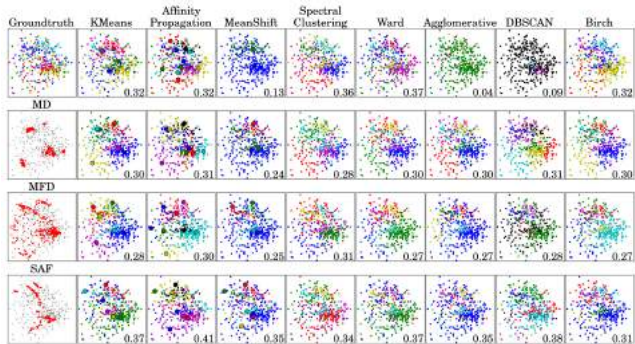
(c) Six Individuals. $h = 0.036$, $\mu = 0.15$.
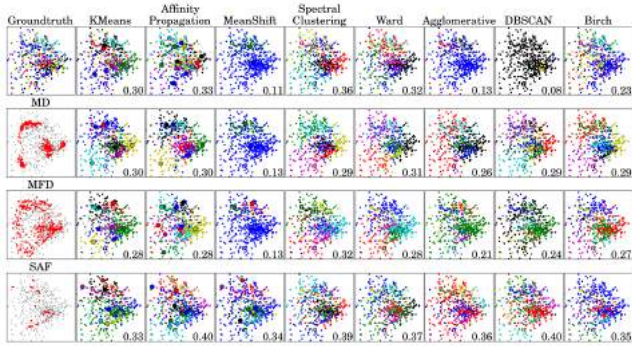
(a) Eight Individuals. $h = 0.034$, $\mu = 0.15$.
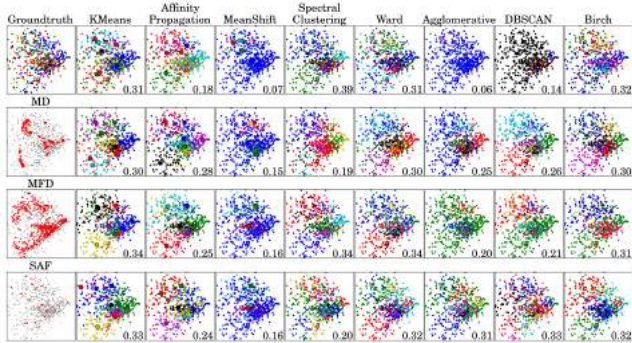
(b) Ten Individuals. $h = 0.033$, $\mu = 0.1$.

Figure 4.25: We illustrate the performance of our consolidation with increasing number of target clusters in the Yale face data. Top row: clustering without SAF consolidation; bottom row: clustering after SAF consolidation. All h values are relative to the bounding box diagonal of the data. The data is pre-processed before SAF as described in the paper (Section 4.2.5).

small μ values are recommended to use, and vice verse.

4.5 Conclusions

We present a novel structure-aware filtering (SAF) algorithm with applications in dimensionality reduction and clustering. We also provide a theoretical analysis of SAF that shows under what circumstances SAF converges to a point distribution on the underlying structures.

Through consolidating high-dimensional data, SAF can greatly facilitate existing dimensionality reduction and clustering techniques. Experiments on both synthetic and real data demonstrate that the performances of a variety of clustering algorithms are significantly boosted after SAF is applied, and SAF outperforms other state of the art techniques for manifold denoising.
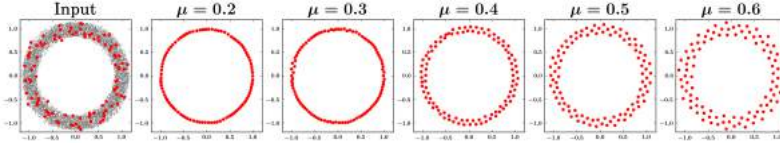
(a) $h = 0.1$. Convergence criterion: $\mu < 0.31$.
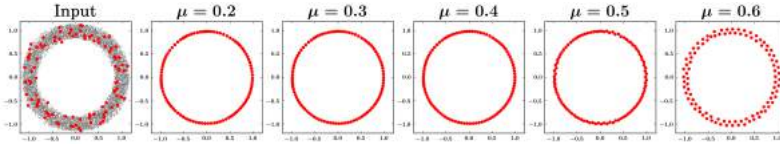
(b) $h = 0.15$. Convergence criterion: $\mu < 0.5$.
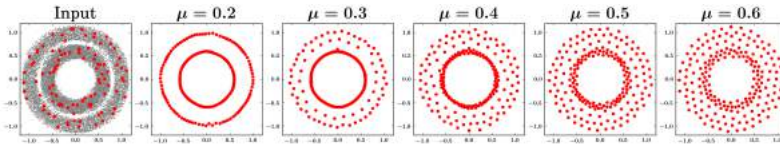
(c) $h = 0.1$. Convergence criterion: $\mu < 0.31$.
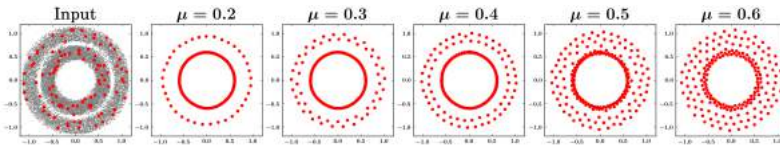
(d) $h = 0.15$. Convergence criterion: $\mu < 0.5$.

Figure 4.26: We illustrate a parameter selection strategy guided by our convergence criterion. All input data (gray) are corrupted by Gaussian noise with $\sigma = 0.15$. For simple structures, such as the ring data in the first two rows, using a larger h value (second row) provides a smoother approximation of the underlying data density (gray). This encourages a more uniform distribution of output points and allows for a wider choice ofrepulsions μ. For data with close-by structures, such as the two rings used in the last two rows, using a too large h value (second row) may be incapable of separating the nearby structures, however. In practice, one should first select an appropriate kernel size h according to the input data, and then choose μ close to the theoretical maximum to get a regular output point distribution. Note that we used denser input points (in gray color) than the consolidated output points (in red color) to better approximate our continuous formulation.

(a) $h = 0.1$. Convergence criterion: $\mu < 0.2$.

(b) $h = 0.15$. Convergence criterion: $\mu < 0.36$.

(c) $h = 0.2$. Convergence criterion: $\mu < 0.5$.

(d) $h = 0.25$. Convergence criterion: $\mu < 0.6$.
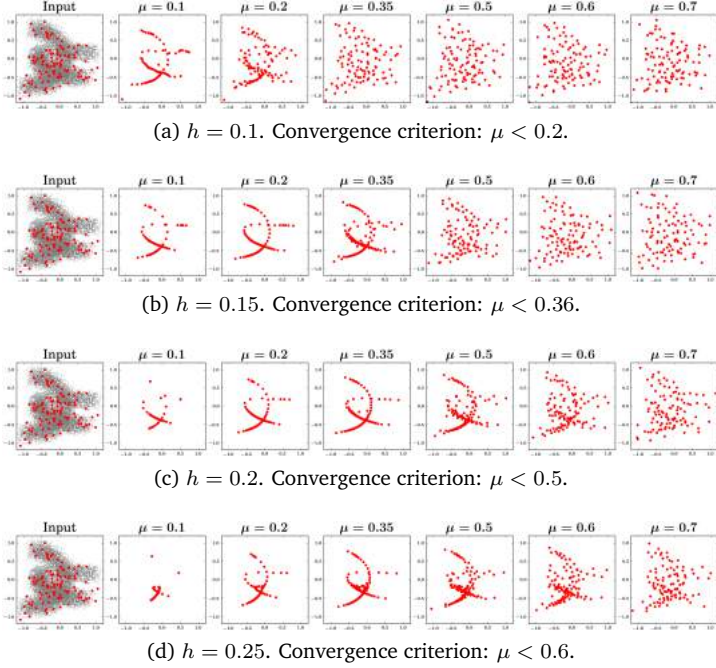
Figure 4.27: We further discuss parameter selection using the 3D data set from Figure 4.1 in the paper. All input data are corrupted by Gaussian noise with $\sigma = 0.2$ shown in gray. Different h values are used for different rows. Using a big h value can better deal with noise and outliers, but different structures could be mistakenly merged if a too big h value is used (last row). On the other hand, using a small h value can better preserve the shape of the structure, but is less robust to the noise and may lead to less than satisfactory output distributions (red). We further show different μ values in each row. Using a big μ value usually gives more regular output distribution, but may violate the convergence criterion. Using a small μ value may cause disconnection in the structure. In general, guided by our theoretical analysis and knowledge of the data (e.g. how close nearby structures could be), as large as possible h and μ values are advised. We used denser input points (gray) than consolidated output points (red) to better approximate the continuous setting.

Chapter 5

Specular-to-Diffuse Translation

5.1 Introduction

Three-dimensional reconstruction from multi-view images is a long standing problem in computer vision. State-of-the-art shape-from-shading techniques achieve impressive results [79, 90]. These techniques, however, make rather strong assumptions about the data, mainly that target objects are predominantly diffuse with almost no specular reflectance. Multi-view reconstruction of glossy surfaces is a challenging problem, which has been addressed by adding specialized hardware (e.g., coded pattern projection [133] and two-layer LCD [136]), imposing surface constraints [63, 116], or making use of additional information like silhouettes and environment maps [42], or the Blinn-Phong model [72].

In this paper, we present a generative adversarial neural network (GAN) that translates multi-view images of objects with specular reflection to diffuse ones. The network aims to generate a specular-free surface, which then can be reconstructed by a standard multi-view reconstruction technique as shown in Figure 5.1. We name our trans-

Figure 5.1: Specular-to-diffuse translation of multi-view images. We show eleven views of a glossy object (top), and the specular-free images generated by our network (bottom).

lation network, S2Dnet, for Specular-to-Diffuse. Our approach is inspired by recent GAN-based image translation methods, like pix2pix [65] or cycleGAN [169], that can transform an image from one domain to another. Such techniques, however, are not designed for multi-view image translation. Directly applying these translation techniques to individual views is prone to reconstruction artifacts due to the lack of coherence among the transformed images. Hence, instead of using single views, our network considers a triplet of nearby views as input. These triplets allow learning the mutual information of neighboring views. More specifically, we introduce a global-local discriminator and a perceptual correspondence loss that evaluate the multi-view coherency of local corresponding image patches. Experiments show that our method outperforms baseline image translation methods.

Another obstacle of applying image translation techniques to specularity removal is the lack of good training data. It is rather imprac-

tical to take enough paired or even unpaired photos to successfully train a deep network. Inspired by the recent works of simulating training data by physically-based rendering [164, 96, 125, 93] and domain adaptation [53, 9, 73, 67], we present a fine-tuned process for generating training data, then adapting it to real world data. Instead of using Shapenet [21], we develop a new training dataset that includes models with richer geometric details, which allows us to apply our method to complex real-world data. Both quantitative and qualitative evaluations demonstrate that the performance of multi-view reconstruction can be significantly improved using the images filtered by our network. We show also the performance of adapting our network on real world training and testing data with some promising results.

5.2   Related work

Specular Object Reconstruction.   Image based 3D reconstruction has been widely used for AR/VR applications, and the reconstruction speed and quality have been improved dramatically in recent years. However, most photometric stereo methods are based on the assumption that the object surface is diffuse, that is, the appearance of the object is view independent. Such assumptions, however, are not valid for glossy or specular objects in uncontrolled environments. It is well known that modeling the specularity is difficult as the specular effects are largely caused by the complicated global illumination that is usually unknown. For example, Godard et al. [42] first reconstruct a rough model by silhouette and then refine it using the specified environment map. Their method can reconstruct high quality specular surfaces from HDR images with extra information, such as silhouette and environment map.

In contrast, our method requires only the multi-view images as input. Researchers have proposed sophisticated equipment, such as a setup with two-layer LCDs to encode the directions of the emitted light field [136], taking advantages of the IR images recorded by RGB-D scanners [100, 101] or casting coded patterns onto mirror-like object-

s [133]. While such techniques can effectively handle challenging non-diffuse effects, they require additional hardware and user expertise. Another way to tackle this problem is by introducing additional assumptions, such as surface constraints [63, 116], the Blinn-Phong model [72], and shape-from-specularity [23]. These methods can also benefit from our network that outputs diffuse images, where strong specularities are removed from uncontrolled illumination. Please refer to [61] for a survey on specular object reconstruction.

GAN-based Image-to-Image Translation. We are inspired by the latest success of learning based image-to-image translation methods, such as ConditionalGAN [65], cycleGAN [169], [158] dualGAN, and discoGAN [73]. The remarkable capacity of Generative Adversarial Networks (GANs) [45] in modeling data distributions allows these methods to transform images from one domain to another with relatively small amounts of training data, while preserving the intrinsic structure of original images faithfully. With improved multi-scale training techniques, such as Progressive GAN [69] and pix2pixHD [146], image-to-image translation can be performed at mega pixel resolutions and achieve results of stunning visual quality.

Recently, modified image-to-image translation architectures have been successfully applied to ill-posed or underconstrained vision tasks, including face frontal view synthesis [60], facial geometry reconstruction [111, 119, 112, 120], raindrop removal [108], or shadow removal [145]. These applications motivate us to develop a glossiness removal method based on GANs to facilitate multi-view 3D reconstruction of non-diffuse objects.

Learning-based Multi-View 3D Reconstruction. Learning surface reconstruction from multi-view images end-to-end has been an active research direction recently [134, 24, 83, 138]. Wu et al. [152] and Gwak et al. [47] use GANs to learn the latent space of shapes and apply it to single image 3D reconstruction. 3D-R2N2 [24] designs a recurrent network for unified single and multi-view reconstruction. Image2Mesh [106] learns parameters of free-form-deformation of a base model. Nonetheless, in general, the reconstruction quality of these methods cannot really surpass that of traditional approaches

that exploit multiple-view geometry and heavily engineered photo-metric stereo pipelines. To take the local image feature coherence into account, we focus on removing the specular effect on the image level and resort to the power of multi-view reconstruction as a post-processing and also a production step.

On the other hand, there are works, closer to ours, that focus on applying deep learning on subparts of the stereo reconstruction pipeline, such as depth and pose estimation [168], feature point detection and description [157, 29], semantic segmentation [89], and bundle adjustment [170, 171]. These methods still impose the Lambertian assumption for objects or scenes, where our method can serve as a preprocessing step to deal with glossiness.

Learning-based Intrinsic Image Decomposition. Our method is also loosely related to some recent works on learning intrinsic image decomposition. These methods include training a CNN to reconstruct rendering parameters, e.g., material [86, 161], reflectance maps [110], illumination [40], or some combination of those components [125, 39, 86]. These methods are often trained on synthetic data and are usually applied to the re-rendering of single images. Our method shares certain similarity with these methods, while we focus on removing the glossy effects from multi-view images using a conditional GAN architecture.

5.3 Multi-view Specular-to-Diffuse GAN

In this section, we introduce S2Dnet, a conditional GAN that translates multi-view images of highly specular scenes into corresponding diffuse images. The input to our model is a multi-view sequence of a glossy scene without any additional input such as segmentation masks, camera parameters, or light probes. This enables our model to process real-world data, where such additional information is not readily available. The output of our model directly serves as input to state-of-the-art photometric stereo pipelines, resulting in improved 3D reconstruction without additional effort. Figure 5.2 shows a visualiza-
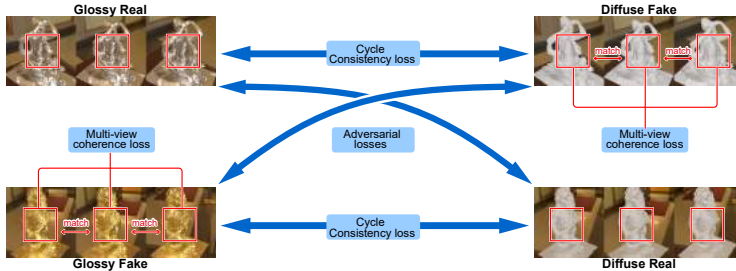
Figure 5.2: Overview of S2Dnet. Two generators and two discriminators are trained simultaneously to learn cross-domain translations between the glossy and the diffuse domain. In each training iteration, the model randomly picks and forwards a real glossy and diffuse image sequence, computes the loss functions and updates the model parameters.

Figure 5.3: Gallery of our synthetically rendered specular-to-diffuse training data.

tion of the proposed model. We discuss the training data, one of our major contributions, in Section 5.3.1. In Section 5.3.2 we introduce the concept of inter-view coherence that enables our model to process multiple views of a scene in a consistent manner, which is important in the context of multi-view reconstruction. Then, we outline in Section 5.3.3 the overall end-to-end training procedure. Implementation details are discussed in Section 5.3.4. Upon publication we will release both our data (synthetic and real) and the proposed model to foster further work.

5.3.1 Training Data

To train our model to translate multi-view glossy images to diffuse correspondents, we need appropriate data for both domains, i.e., glossy source domain images as inputs, and diffuse images as the target domain. Yi et al.[158] propose a MATERIAL dataset consisting of unlabeled data grouped in different material classes, such as plastic, fabric, metal, and leather, and they train GANs to perform material transfer. However, the MATERIAL dataset does not contain multi-view images and thus is not suited for our application. Moreover, the dataset is rather small and we expect our deep model to require a larger amount of training data. Hence, we propose a novel synthetic dataset consisting of multi-view images, which is both sufficiently large to train deep networks and complex to generalize to real-world objects. For this purpose, we collect and align 91 watertight and noise-free geometric models featuring rich geometric details from SketchFab (Figure 5.3). We exclude three models for testing and use the remaining 88 models for training. To obtain a dataset that generalizes well to real-world images, we use PBRT, a physically based renderer [105] to render these geometric models in various environments with a wide variety of glossy materials applied to form our source domain. Next, we render the target domain images by applying a Lambertian material to our geometric models.

Our experiments show that the choice of the rendering parameters has a strong impact on the translation performance. On one hand, making the two domains more similar by choosing similar materials for both domains improves the translation quality on synthetic data. Moreover, simple environments, such as a constant ground plane, also increase the quality on synthetic data. On the other hand, such simplifications cause the model to overfit and prevent generalization to real-world data. Hence, a main goal of our dataset is to provide enough complexity to allow generalization to real data. To achieve realistic illumination, we randomly sample one of 20 different HDR indoor environment maps and randomly rotate it for each scene. In addition, we orient a directional light source pointing from the camera approximately towards the center of the scene and position two

additional light sources above the scene. The intensities, positions, and directions of these additional light sources are randomly jittered. This setup guarantees a rather even, but still random illumination. To render the source domain images, we applied the various metal materials defined in PBRT, including copper, silver, and gold. Material roughness and index of refraction are randomly sampled to cover a large variety of glossy materials. We randomly sample camera positions on the upper hemisphere around the scene pointing towards the center of the scene. To obtain multi-view data, we always sample 5 close-by, consecutive camera positions in clock-wise order while keeping the scene parameters fixed to mimic the common procedure of taking photos for stereo reconstruction. Since we collect 5 images of the same scene and the input to our network consists of 3 views, we obtain 3 training samples per scene. All rendered images are of 512×512 resolution, which is the limit for our GPU memory. However, it is likely that higher resolutions would further improve the reconstruction quality. Finally, we render the exact same images again with a white, Lambertian material, i.e., the mapping from the source to the target domain is bijective. The proposed procedure results in a training dataset of more than 647k images, i.e., more than 320k images per domain. For testing, we rendered 2k sequences of images, each consisting of 50 images. All qualitative results on synthetic data shown in this paper belong to this test set.

5.3.2   Inter-view Coherence

Multi-view reconstruction algorithms leverage corresponding features in different views to accurately estimate the 3D geometry. Therefore, we cannot expect good reconstruction quality if the glossy images in a multi-view sequence are translated independently using standard image translation methods, e.g., [169, 65]. This will introduce inconsistencies along the different views, and thus cause artifacts in the subsequent reconstruction. We therefore propose a novel model that enforces inter-view coherence by processing multiple views simultaneously. Our approach consists of a global and local consistency
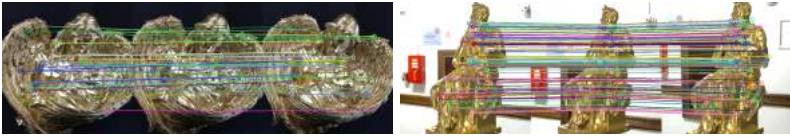
Figure 5.4: Two examples of the SIFT correspondences pre-computed for our training.

constraint: the global constraint is implemented using an appropriate network architecture, and the local consistency is enforced using a novel loss function.

Global Inter-view Coherence. A straightforward idea to incorporate multiple views is to stack them pixel-by-pixel before feeding them to the network. We found that this does not lead to strong enough constraints, since the network can still learn independent filter weights for the different views. This results in blurry translations, especially if corresponding pixels in different views are not aligned, which is typically the case. Instead, we concatenate the different views along the spatial axis before feeding them to the network. This solution, although simple, enforces the network to use the same filter weights for all views, and thus effectively avoids inconsistencies on a global scale.

Local Inter-view Coherence. Incorporating loss functions based on local image patches has been successfully applied to generative adversarial models, such as image completion [62] or texture synthesis [156]. However, comparing image patches at random locations is not meaningful in a multi-view setup for stereo reconstruction. Instead, we encourage the network to maintain feature point correspondences in the input sequence, i.e., inter-view correspondences should be invariant to the translation. Since the subsequent reconstruction pipeline relies on such correspondences, maintaining them during translation should improve reconstruction quality. To achieve

this, we first extract SIFT feature correspondences for all training images. For each training sequence consisting of three views, we compute corresponding feature points between the different views in the source domain; see Figure 5.4 for two examples. During training, we encourage the network output at the SIFT feature locations to be similar along the views using a perceptual loss in VGG feature space [37, 146, 144, 140]. The key idea is to measure both high- and low-level similarity of two images by considering the difference of feature activations in a deep CNN like VGG. We adopt this idea to keep local image patches around corresponding SIFT features perceptually similar in the translated output. The perceptual loss in VGG feature space is defined as:

$$\mathcal{L}_{VGG}(x, \hat{x}) = \sum_{i=1}^{N} \frac{1}{M_i} \|F^{(i)}(x) - F^{(i)}(\hat{x})\|_1, \qquad (5.1)$$

where $F^{(i)}$ denotes the i-th layer in the VGG network consisting of M_i elements. Now consider a glossy input sequence consisting of three images X_1, X_2, X_3, and the corresponding diffuse sequence $\tilde{X}_1, \tilde{X}_2, \tilde{X}_3$ produced by our model. A SIFT correspondence for this sequence consists of three image coordinates p_1, p_2, p_3, one in each glossy image, and all three pixels at the corresponding coordinates represent the same feature. We then extract local image patches $\tilde{x}_i$ centered at p_i from $\tilde{X}_i$, and define the perceptual correspondence loss as:

$$\mathcal{L}_{corr}(\tilde{X}_1, \tilde{X}_2, \tilde{X}_3) = \mathcal{L}_{VGG}(\tilde{x}_1, \tilde{x}_2) + \mathcal{L}_{VGG}(\tilde{x}_2, \tilde{x}_3) + \mathcal{L}_{VGG}(\tilde{x}_1, \tilde{x}_3).$$
$$(5.2)$$

5.3.3   Training Procedure

Given two sets of data samples from two domains, a source domain A and a target domain B, the goal of image translation is to find a mapping T that transforms data points $X_i \in A$ to B such that $T(X_i) = \tilde{X}_i \in B$, while the intrinsic structure of X_i should be preserved under T. Training GANs has been proven to produce astonishing results on this task, both in supervised settings where the data

of the two domains are paired [65], and in unsupervised cases using unpaired data [169]. In our experiments, we observed that both approaches (ConditionalGAN [65] and cycleGAN [169]) perform similarly well on our dataset. However, while paired training data might be readily available for synthetic data, paired real-world data is difficult to obtain. Therefore we come up with a design for unsupervised learning that can easily be fine-tuned on unpaired real-world data.

Cycle-consistency Loss. Similar to CycleGAN [169], we learn the mapping between domain A and B with two translators $G_B : A \rightarrow B$ and $G_A : B \rightarrow A$ that are trained simultaneously. The key idea is to train with cycle-consistency loss, i.e., to enforce that $G_A(G_B(X)) \approx X$ and $G_B(G_A(Y)) \approx Y$, where $X \in A$ and $Y \in B$. This cycle-consistency loss guarantees that data points preserve their intrinsic structure under the learned mapping. Formally, the cycle-consistency loss is defined as:

$$\mathcal{L}_{cyc}(X,Y) = \|G_A(G_B(X)) - X\|_1 + \|G_B(G_A(Y)) - Y\|_1. \quad (5.3)$$

Adversarial Loss. To enforce the translation networks to produce data that is indistinguishable from genuine images, we also include an adversarial loss to train our model. For both translators, in GAN context often called generators, we train two additional discriminator networks D_A and D_B that are trained to distinguish translated from genuine images. To train our model, we use the following adversarial term:

$$\mathcal{L}_{adv} = \mathcal{L}_{GAN}(G_A, D_A) + \mathcal{L}_{GAN}(G_B, D_B), \quad (5.4)$$

where $\mathcal{L}_{GAN}(G, D)$ is the LSGAN formulation [91].

Overall, we train our model using the following loss function:

$$\mathcal{L} = \lambda_{adv}\mathcal{L}_{adv} + \lambda_{cyc}\mathcal{L}_{cyc} + \lambda_{corr}\mathcal{L}_{corr}, \quad (5.5)$$

where λ_{adv}, λ_{cyc}, and λ_{corr} are user-defined hyperparameters.
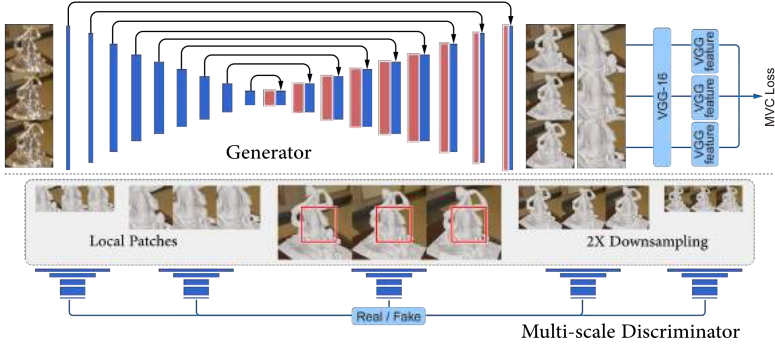
Figure 5.5: Illustration of the generator and discriminator network. The generator uses the U-net architecture and both input and output are a multi-view sequence consisting of three views. A random SIFT correspondence is sampled during training to compute the correspondence loss. The multi-scale joint discriminator examines three scales of the image sequence and two scales of corresponding local patches. The width and height of each rectangular block indicate the channel size and the spatial dimension of the output feature map, respectively.

5.3.4   Implementation Details

Our model is based on cycleGAN and implemented in Pytorch. We experimented with different architectures for the translation networks, including U-Net [113], ResNet [50], and RNN-blocks [20]. Given enough training time, we found that all networks produce similar results. Due to its memory efficiency and fast convergence, we chose U-Net for our final model. As shown in Figure 5.5, we use the multi-scale discriminator introduced in [146] that downsamples by a rate of 2, which generally works better for high resolution images. Our discriminator also considers the local correspondence patches as additional input, which helps to produce coherent translations. Followed by the training guidances proposed in [69], we use pixel-wise normalization in the generators and add a 1-strided convolutional layer after each deconvolutional layer. For computing the correspondence

loss, we use a patch size of 256×256 and sample a single SIFT correspondence per training iteration randomly. The discriminator follows the architecture as: C64-C128-C256-C512-C1. The generator's encoder architecture is: C64-C128-C256-C512-C512-C512-C512-C512. We use $\lambda_{adv} = 1, \lambda_{cyc} = 10, \lambda_{corr} = 5$ in all our experiments and train using the ADAM optimizer with a learning rate of 0.0002. Tables 5.1 and 5.2 give more detail of our network architecture. Xavier [41] is used for weights initialization. We train our models on an NVIDIA 1080Ti GPU with 11GB GPU memory, which only allows us to use a training batch size of 1.

Layer	Input $\rightarrow$ Output Shape	Layer Information
Input Layer	$(h, 3w, 3) \rightarrow (\frac{h}{2}, \frac{3w}{2}, 64)$	CONV-(N64, K4x4, S2, P1), LeReLU
Hidden Layers	$(\frac{h}{2}, \frac{3w}{2}, 64) \rightarrow (\frac{h}{4}, \frac{3w}{4}, 128)$	CONV-(N128, K4x4, S2, P1), PN, LeReLU
	$(\frac{h}{4}, \frac{3w}{4}, 128) \rightarrow (\frac{h}{8}, \frac{3w}{8}, 256)$	CONV-(N256, K4x4, S2, P1), PN, LeReLU
	$(\frac{h}{8}, \frac{3w}{8}, 256) \rightarrow (\frac{h}{16}, \frac{3w}{16}, 512)$	CONV-(N512, K4x4, S2, P1), PN, LeReLU
Output Layer	$(\frac{h}{16}, \frac{3w}{16}, 512) \rightarrow (\frac{h}{32}, \frac{3w}{32}, 1)$	CONV-(N1, K4x4, S2, P1)

Table 5.1: Discriminator network architecture. We use 5 such discriminators that have an identical network structure but operate at three scales of the image sequence and two scales of corresponding local patches using LSGAN (see Figure 5 in the paper). N: the number of output channels, K: kernel size, S: stride size, P: padding size, PN: pixel-wise Normalization, LeReLU: LeakyReLU with $\alpha = 0.2$, w, h: width and height of input images. Note that the input width is $3w$ because we spatially concatenate the three views of the input sequences.

5.4 Evaluation

In this section, we present qualitative and quantitative evaluations of our proposed S2Dnet. For this purpose, we evaluate the performance of our model on both the translation task and the subsequent 3D reconstruction, and we compare to several baseline systems. In Section 5.4.1 we report results on our synthetic test set and we also perform an evaluation on real-world data in Section 5.4.2.

Part	Input → Output Shape	Layer Information
	(h, 3w, 3) → ($\frac{h}{2}$, $\frac{3w}{2}$, 64)	CONV-(N64, K4x4, S2, P1), LeReLU
	($\frac{h}{2}$, $\frac{3w}{2}$, 64) → ($\frac{h}{4}$, $\frac{3w}{4}$, 128)	CONV-(N128, K4x4, S2, P1), PN, LeReLU
	($\frac{h}{4}$, $\frac{3w}{4}$, 128) → ($\frac{h}{8}$, $\frac{3w}{8}$, 256)	CONV-(N256, K4x4, S2, P1), PN, LeReLU
Down-sampling	($\frac{h}{8}$, $\frac{3w}{8}$, 256) → ($\frac{h}{16}$, $\frac{3w}{16}$, 512)	CONV-(N512, K4x4, S2, P1), PN, LeReLU
	($\frac{h}{16}$, $\frac{3w}{16}$, 512) → ($\frac{h}{32}$, $\frac{3w}{32}$, 512)	CONV-(N512, K4x4, S2, P1), PN, LeReLU
	($\frac{h}{32}$, $\frac{3w}{32}$, 512) → ($\frac{h}{64}$, $\frac{3w}{64}$, 512)	CONV-(N512, K4x4, S2, P1), PN, LeReLU
	($\frac{h}{64}$, $\frac{3w}{64}$, 512) → ($\frac{h}{128}$, $\frac{3w}{128}$, 512)	CONV-(N512, K4x4, S2, P1), PN, LeReLU
	($\frac{h}{128}$, $\frac{3w}{128}$, 512) → ($\frac{h}{256}$, $\frac{3w}{256}$, 512)	CONV-(N512, K4x4, S2, P1), PN, LeReLU
	($\frac{h}{256}$, $\frac{3w}{256}$, 512) → ($\frac{h}{512}$, $\frac{3w}{512}$, 512)	CONV-(N512, K4x4, S2, P1), ReLU
	($\frac{h}{512}$, $\frac{3w}{512}$, 512) → ($\frac{h}{256}$, $\frac{3w}{256}$, 512)	DECONV-(N512, K4x4, S2, P1), CONV-(N512, K3x3, S1, P1), PN, ReLU
	($\frac{h}{256}$, $\frac{3w}{256}$, 1024) → ($\frac{h}{128}$, $\frac{3w}{128}$, 512)	DECONV-(N512, K4x4, S2, P1), CONV-(N512, K3x3, S1, P1), PN, ReLU
	($\frac{h}{128}$, $\frac{3w}{128}$, 1024) → ($\frac{h}{64}$, $\frac{3w}{64}$, 512)	DECONV-(N512, K4x4, S2, P1), CONV-(N512, K3x3, S1, P1), PN, ReLU
	($\frac{h}{64}$, $\frac{3w}{64}$, 1024) → ($\frac{h}{32}$, $\frac{3w}{32}$, 512)	DECONV-(N512, K4x4, S2, P1), CONV-(N512, K3x3, S1, P1), PN, ReLU
Up-sampling	($\frac{h}{32}$, $\frac{3w}{32}$, 1024) → ($\frac{h}{16}$, $\frac{3w}{16}$, 512)	DECONV-(N512, K4x4, S2, P1), CONV-(N512, K3x3, S1, P1), PN, ReLU
	($\frac{h}{16}$, $\frac{3w}{16}$, 1024) → ($\frac{h}{8}$, $\frac{3w}{8}$, 256)	DECONV-(N256, K4x4, S2, P1), CONV-(N256, K3x3, S1, P1), PN, ReLU
	($\frac{h}{8}$, $\frac{3w}{8}$, 512) → ($\frac{h}{4}$, $\frac{3w}{2}$, 128)	DECONV-(N128, K4x4, S2, P1), CONV-(N128, K3x3, S1, P1), PN, ReLU
	($\frac{h}{4}$, $\frac{3w}{4}$, 256) → ($\frac{h}{2}$, $\frac{3w}{2}$, 64)	DECONV-(N64, K4x4, S2, P1), CONV-(N64, K3x3, S1, P1), PN, ReLU
	($\frac{h}{2}$, $\frac{3w}{2}$, 64) → (h, 3w, 3)	DECONV-(N3, K4x4, S2, P1), CONV-(N512, K3x3, S1, P1), Tanh

Table 5.2: Generator network architecture.

To evaluate the benefit of our proposed inter-view coherence, we perform a comparison to a single-view translation baseline by training a cycleGAN network [169] on glossy to diffuse translation. Since our synthetic dataset features a bijective mapping between glossy and diffuse images, we also train a pix2pix network [65] for a supervised baseline on synthetic data. In addition, we compare reconstruction quality to performing stereo reconstruction directly on the glossy multi-view sequence to demonstrate the benefit of translating the input as a preprocessing step. For 3D reconstruction, we apply a state-of-the-art multi-view surface reconstruction method [79] on input sequences consisting of 10 to 15 views. For our method, we translate each input view sequentially but we feed the two neighboring views as additional inputs to our multi-view network. For the two baseline translation methods, we translate each view independently. The 3D reconstruction pipeline then uses the entire translated multi-view sequence as input.

	Glossy	pix2pix	cycleGAN	S2Dnet
Image MSE	118.39	56.20	69.15	57.78

Table 5.3: Quantitative evaluation of the image error on our synthetic testing data.

Figure 5.6: Qualitative translation results on a synthetic input sequence consisting of 8 views. From top down: the glossy input sequence, the ground truth diffuse rendering, and the translation results for the baselines pix2pix and cycleGAN, and our S2Dnet. The output of pix2pix is generally blurry. The cycleGAN output, although sharp, lacks inter-view consistency. Our S2Dnet produces both crisp and coherent translations.

5.4.1   Synthetic Data

For a quantitative evaluation of the image translation performance, we compute MSE with respect to the ground truth diffuse renderings on our synthetic test set. Table 5.3 shows a comparison of our S2Dnet to

Model	1	2	3	4	5	6	7	8	9	10	AVG
Glossy	0.67	0.88	1.35	0.76	1.15	1.13	1.15	0.78	0.54	0.66	0.90
cycleGAN	1.18	0.72	0.89	0.59	1.35	0.72	0.99	0.62	0.51	0.42	0.80
S2Dnet	0.52	0.67	0.72	0.43	0.87	0.54	0.92	0.65	0.55	0.56	0.64

Table 5.4: Quantitative evaluation of surface reconstruction performance on 10 different scenes. The error metric is the percentage of bounding box diagonal. Our S2Dnet performs best, and the translation baseline still performs significantly better than directly reconstructing from the glossy images. The numbering of the models follows the visualization in Figure 5.7, using the same left to right order.

pix2pix and cycleGAN. Unsurprisingly, the supervised pix2pix network performs best, closely followed by our S2Dnet, which outperforms the unsupervised baseline by a significant margin. In Figure 5.6 we show qualitative translation results. Note that the output of pix2pix is generally blurry. Since MSE penalizes outliers and prefers a smooth solution, pix2pix still achieves a low MSE error. While the output of cycleGAN is sharper, the translated sequence lacks inter-view consistency, whereas our S2Dnet produces both highly detailed and coherent translations.

Next, we evaluate the quality of the surface reconstruction by feeding the translated sequences to the reconstruction pipeline. We found that the blurry output of pix2pix is not suitable for stereo reconstruction, since already the first step, estimating camera parameters based on feature correspondences, fails on this data. We therefore exclude pix2pix from the surface reconstruction evaluation but include the trivial baseline of directly reconstructing from the glossy input sequence to demonstrate the benefit of the translation step. In order to compute the geometric error of the surface reconstruction output, we register the reconstructed geometry to the ground truth mesh using a variant of ICP [115]. Next, we compute the Euclidean distance of each reconstructed surface point to its nearest neighbor in the ground truth mesh and report the per-model mean value. Table 5.4 shows the surface reconstruction error for our S2Dnet in comparison to the three baselines. The numbers show that our S2Dnet performs best, and that preprocessing the glossy input sequences clearly helps to obtain a more accurate reconstruction, even when using the cycleGAN
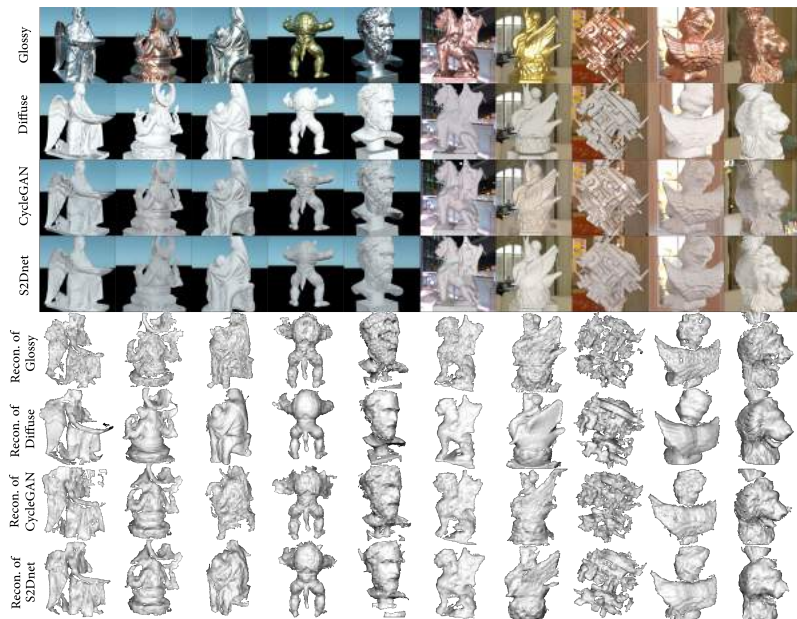
Figure 5.7: Qualitative surface reconstruction results on 10 different scenes. From top to bottom: glossy input, ground truth diffuse renderings, cycleGAN translation outputs, our S2Dnet translation outputs, reconstructions from glossy images, reconstructions from ground truth diffuse images, reconstructions from cycleGAN output, and reconstructions from our S2Dnet output. All sequences are excluded from our training set, and the objects in column 3 and 4 have not even been seen during training.

baseline. In Figure 5.7 we show qualitative surface reconstruction results for 10 different scenes in various environments.

Figure 5.8: Gallery of our glossy-to-diffuse real-world training data and the spray (leftmost column) we used to paint the objects. We first choose 5 diffuse real-world objects and take 5k pictures in total from different camera positions and under varying lighting conditions. We then use a glossy spray paint to cover our objects with a glossy coat and shoot another 5k pictures to represent the glossy domain.

5.4.2 Real-world Data

Since we do not have real-world ground truth data, we compile a real-world test set and perform a qualitative comparison on it. For all methods, we compare generalization performance when training on our synthetic dataset. Moreover, we evaluate how the different models perform when fine-tuning on real-world data, or training on real-world data from scratch. For this purpose, we compile a dataset by shooting photos of real-world objects. We choose 5 diffuse real-world objects and take 5k pictures in total from different camera positions and under varying lighting conditions. Next, we use a glossy spray paint to cover our objects with a glossy coat and shoot another 5k pictures to represent the glossy domain. The resulting dataset consists of unpaired samples of glossy and diffuse objects under real-world conditions, see Figure 5.8 for a gallery of our real-world training data.

In Figure 5.9 we show qualitative translation results on real-world data. All networks are trained on synthetic data only here, and they all manage to generalize to some extent to real-world data, thanks to our high-quality synthetic dataset. Similar to the synthetic results in Figure 5.6, pix2pix produces blurry results, while cycleGAN introduces inconsistent high-frequency artifacts. S2Dnet is able the remove most of the specular effects and preserves geometric details in a consistent

Figure 5.9: Qualitative translation results on a real-world input sequence consisting of 11 views. The first row shows the glossy input sequence and the remaining rows show the translation results of pix2pix, cycleGAN, and our S2Dnet. All networks are trained on synthetic data only. Similar to the synthetic case, cycleGAN outperforms pix2pix, but it produces high-frequency artifacts that are not consistent along the views. Our S2Dnet is able to remove most of the specular effects and preserves all the geometric details in a consistent manner.

manner. In Figure 5.10 we show qualitative surface reconstruction results for 7 different scenes. Artifacts occur mainly close to the object silhouettes in complex background environments. This could be mitigated by training with segmentation masks.

Finally, we evaluate performance when either fine-tuning or training from scratch on real-world data. We retrain or fine-tune S2Dnet and cycleGAN on our real-world dataset, but cannot retrain pix2pix for this purpose, since it relies on a supervision signal that is not present in our unpaired real-world dataset. Our experiments show that training or fine-tuning using such a small dataset leads to heavy overfitting. The translation performance for real-world objects that were not seen during training decreases significantly compared to the models trained on synthetic data only. In Figure 5.11 c) and e) we show image translation results of cycleGAN and S2Dnet when training from scratch on our real-world dataset. Since the scene in Figure 5.11 is part of the training set (although the input image itself is excluded
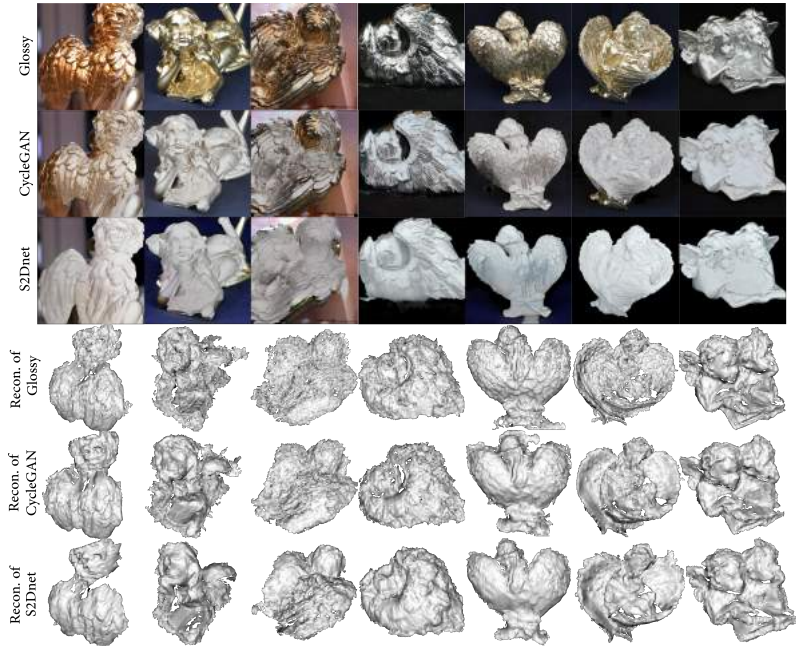
Figure 5.10: Qualitative surface reconstruction results on 7 different real-world scenes. Top to bottom: glossy input, cycleGAN translation outputs, our S2Dnet translation outputs, reconstructions from glossy images, reconstructions from cycleGAN output, and reconstructions from our S2Dnet output. All networks are trained on synthetic data only.

from the training set), our S2Dnet produces decent translation results, which is not the case for scenes not seen during training. Fine-tuning our S2Dnet produces similar results (Figure 5.11 f)).

Figure 5.12 shows more results of our S2Dnet, given input scenes with various illumination and different objects. Figures 5.13 and 5.14 demonstrate two additional examples of image translation and reconstruction, where S2Dnet outperforms both pix2pix and cycleGAN. We

also observe that when the weight for the perceptual correspondence loss is $\lambda_{corr} = 0$, i.e., removing the perceptual correspondence loss, the output of S2Dnet lacks of inter-view consistency.

5.5 Limitations and Future Work

Although the proposed framework enables reconstructing glossy and specular objects more accurately compared to state-of-the-art 3D reconstruction algorithms, a few limitations do exist. First, since the network architecture contains an encoder and a decoder with skip connections, the glossy-to-Lambertian image translation is limited to images of a fixed resolution. This resolutions might be too low for certain types of applications. Next, due to the variability of the background in real images, the translation network might treat a portion of the background as part of the reconstructed object. Similarly, the network occasionally misclassifies the foreground as part of the background, especially in very light domains on specular objects. Finally, as the simulated training data was rendered by assuming a fixed albedo, the network cannot consistently translate glossy materials with spatially varying albedo into a Lambertian surface. We predict that given a larger and more diverse training set in terms of shapes, backgrounds, albedos and materials, the accuracy of the proposed method in recovering real object would be largely enhanced. Our current training dataset includes the most common types of specular material. The proposed translation network has potential to be extended to other more challenging materials, such as transparent objects, given proper training data.

(a) Glossy

(b) Diffuse

(c) CycleGAN
on real dataset

(d) CycleGAN on
synthetic dataset

(e) S2Dnet on
real dataset

(f) S2Dnet on
synthetic dataset

(g) Fine-tune (f)
on real dataset

Recons. of (a)   Recons. of (b)   Recons. of (d)   Recons. of (f)   Recons. of (g)
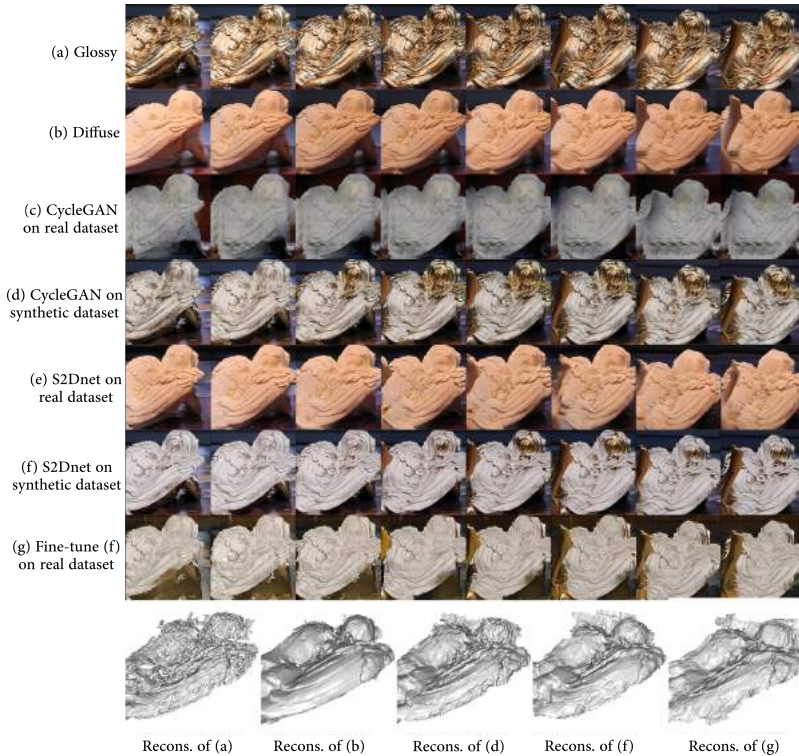
Figure 5.11: A sample of our real-world dataset is shown in (a-b). Translation results of cycleGAN when training from scratch on our real-world dataset or synthetic data only are shown in (c) and (d), respectively. S2Dnet outputs, trained from scratch on our real-world dataset or synthetic data only, are shown in (e) and (f), respectively. Another output of S2Dnet, trained on synthetic data and then fine-tuned on real-world data is presented in (g). The last row demonstrates the corresponding reconstruction results. Note that the output images are blurry when training from scratch on real-world data, i.e. (c) and (e), and thus not suitable for stereo reconstruction.
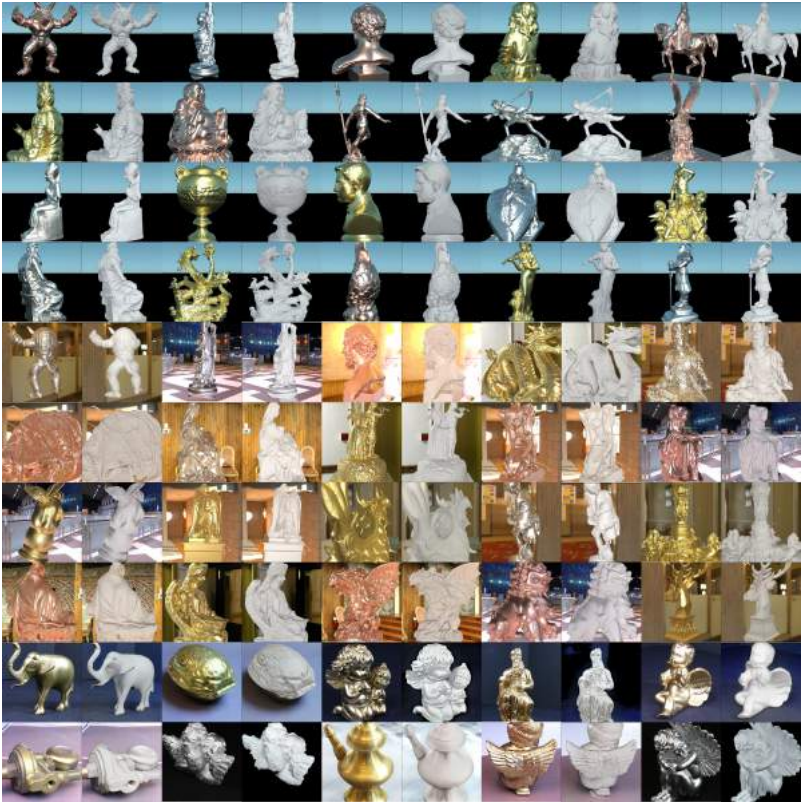
Figure 5.12: Gallery of our glossy-to-diffuse results of 40 synthetic and 10 real-world (last two rows) scenes. All sequences are excluded from our training set. Three synthetic (Armadillo, Standing Buddha, Roman Head Sculpture) and all real-world objects have not even been seen during training.
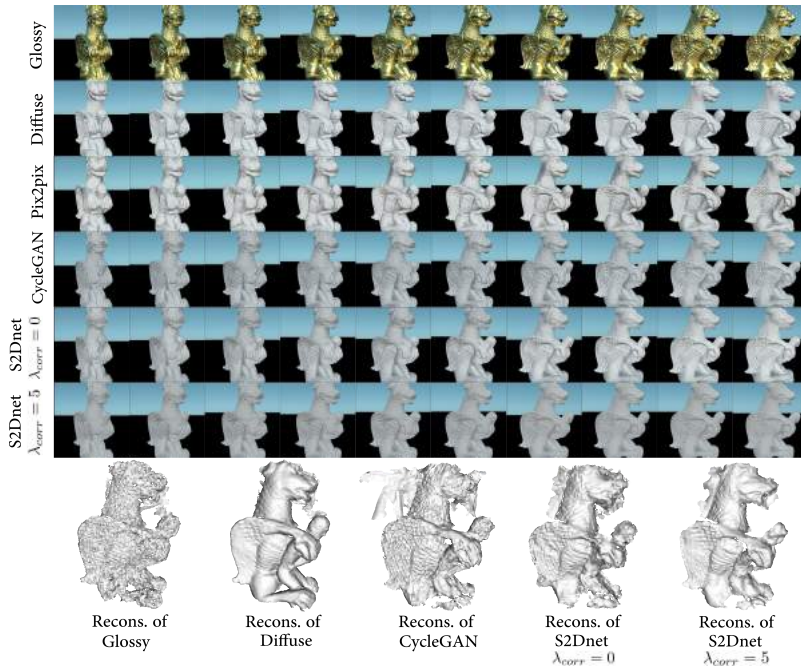
Figure 5.13: Qualitative comparison of image translation and surface reconstruction on a synthetic sequence consisting of 10 views. From top to bottom: glossy input, ground truth diffuse renderings, pix2pix translation outputs, cycleGAN translation outputs, our S2Dnet translation outputs using $\lambda_{corr} = 0$ (no perceptual correspondence loss), our S2Dnet translation outputs using $\lambda_{corr} = 5$. The last row shows the corresponding reconstruction results. All sequences are excluded from our training set. The output of pix2pix is blurry and is not suitable for multi-view reconstruction. The outputs of cycleGAN and our S2Dnet without perceptual correspondence loss, although sharp, lack of inter-view consistency. Our S2Dnet with perceptual correspondence loss ($\lambda_{corr} = 5$) produces crisp and coherent translations, resulting in a better reconstruction.
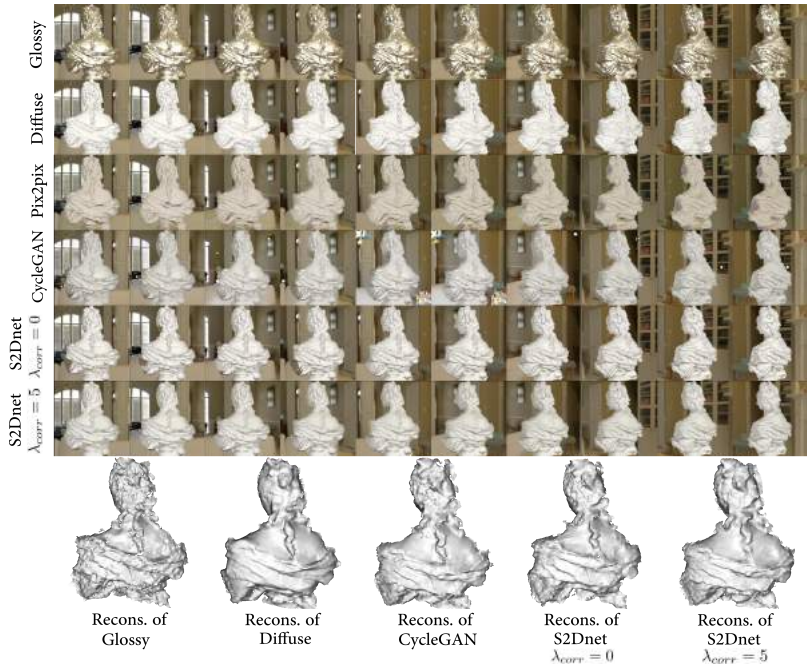
Figure 5.14: Another set of image translation and surface reconstruction comparison on a synthetic input sequence consisting of 10 views.

Chapter 6

Conclusions

In this thesis, we addressed the problem of geometric structure extraction and reconstruction from raw input data with challenging properties including incompleteness, noise, outliers, and inconsistency. This remains an open problem in computer graphics, computer vision, machine learning and etc.. Based on our research interest and expertise, we explore optimization and learning techniques to some particular types of data such as incomplete point cloud, noise contaminated high-dimensional data and inconsistent multi-view images. As a result, three related but independent works are presented: i) the novel deep points consolidation in **Chapter 3** that connects and unifies the skeletal and surface structure of a shape, leading to successful completions of models with large data missing area, ii) a structure-aware data consolidation (SAF) method in **Chapter 4** with has significant theoretical analysis, and promising experimental results and great potential in unsupervised learning, and iii) the S2Dnet for specualr-to-diffuse image translation in **Chapter 5** that improves the multi-view reconstruction of objects with specular reflection.

The limitations and possible improvements of this proposed method are discussed in the previous chapters. Lots of work still have to be done before we can robustly and intelligently extract and reconstruc-

t the underlying geometric structure from arbitrary daily input data with high fidelity. We hope the studies conducted in this thesis contribute to a better understanding of the essence of all geometric structure. As for future works in general, we would like employing our deep points representation to point-based deep learning architectures, deepening the theoretical analysis of points distribution in SAF, and augmenting S2Dnet's insensitivity to complex illumination and material.

Bibliography

[1] Anders Adamson and Marc Alexa. Point-sampled cell complexes. *ACM Trans. on Graphics (Proc. of SIGGRAPH)*, 25(3):671–680, 2006.

[2] M. Alexa, J. Behr, D. Cohen-or, S. Fleishman, D. Levin, and C. T. Silva. Point set surfaces. *Proc. IEEE Int. Conf. on Visualization*, pages 21–28, 2001.

[3] Nina Amenta, Marshall Bern, and Manolis Kamvysselis. A new voronoi-based surface reconstruction algorithm. In *Proc. of SIGGRAPH*, pages 415–421, 1998.

[4] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017.

[5] David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. *Proc. ACM-SIAM Symp. on Discrete algorithms*, pages 1027–1035, 2007.

[6] Oscar Kin-Chung Au, Chiew-Lan Tai, Hung-Kuo Chu, Daniel Cohen-Or, and Tong-Yee Lee. Skeleton extraction by mesh contraction. *ACM Trans. on Graphics (Proc. of SIGGRAPH)*, 27(3):44:1–44:10, 2008.

[7] Haim Avron, Andrei Sharf, Chen Greif, and Daniel Cohen-Or. ℓ_1-sparse reconstruction of sharp point set surfaces. *ACM Trans. on Graphics*, 29(5):135:1–135:12, 2010.

[8] Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Comput.*, 15(6):1373–1396, 2003.

[9] Sagie Benaim and Lior Wolf. One-sided unsupervised domain mapping. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 752–762. Curran Associates, Inc., 2017.

[10] Matthew Berger, Joshua A. Levine, Luis Gustavo Nonato, Gabriel Taubin, and Claudio T. Silva. A benchmark for surface reconstruction. *ACM Trans. on Graphics (Proc. of SIGGRAPH)*, 32(2):20:1–20:17, 2013.

[11] Matthew Berger, Andrea Tagliasacchi, Lee M Seversky, Pierre Alliez, Gael Guennebaud, Joshua A Levine, Andrei Sharf, and Claudio T Silva. A survey of surface reconstruction from point clouds. In *Computer Graphics Forum*, volume 36, pages 301–329. Wiley Online Library, 2017.

[12] Matthew Berger, Andrea Tagliasacchi, Lee M. Seversky, Pierre Alliez, Joshua A. Levine, Andrei Sharf, and Claudio Silva. State of the art in surface reconstruction from point clouds. *Eurographics STAR*, pages 165–185, 2014.

[13] Ingwer Borg and Patrick JF Groenen. *Modern multidimensional scaling: Theory and applications*. Springer Science & Business Media, 2005.

[14] Alexander Bucksch, Roderik Lindenbergh, and Massimo Menenti. Skeltre: Robust skeleton extraction from imperfect point clouds. *The Visual Computer*, 26(10):1283–1300, 2010.

[15] Stéphane Calderon and Tamy Boubekeur. Point morphology. *ACM Trans. on Graphics (Proc. of SIGGRAPH)*, 33(4):45:1–45:13, 2014.

[16] Emmanuel J Candès, Xiaodong Li, Yi Ma, and John Wright. Robust principal component analysis? *Journal of the ACM (JACM)*, 58(3):11, 2011.

[17] Junjie Cao, Andrea Tagliasacchi, Matt Olson, Hao Zhang, and Zhinxun Su. Point cloud skeletons via laplacian based contraction. *Proc. IEEE Int. Conf. on Shape Modeling & Applications*, pages 187–197, 2010.

[18] J. C. Carr, R. K. Beatson, J. B. Cherrie, T. J. Mitchell, W. R. Fright, B. C. McCallum, and T. R. Evans. Reconstruction and representation of 3D objects with radial basis functions. *Proc. of SIGGRAPH*, pages 67–76, 2001.

[19] Lawrence Cayton. Algorithms for manifold learning. *Univ. of California at San Diego Tech. Rep*, 12(1-17):1, 2005.

[20] Chakravarty R. Alla Chaitanya, Anton S. Kaplanyan, Christoph Schied, Marco Salvi, Aaron Lefohn, Derek Nowrouzezahrai, and Timo Aila. Interactive reconstruction of monte carlo image sequences using a recurrent denoising autoencoder. *ACM Trans. Graph.*, 36(4):98:1–98:12, July 2017.

[21] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. ShapeNet: An Information-Rich 3D Model Repository. Technical Report arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015.

[22] Moses Charikar, Sudipto Guha, É. Tardos, and David B. Shmoys. A constant-factor approximation algorithm for the k-median problem. *J. of Computer and System Sciences*, 65(1):129–149, 2002.

[23] Tongbo Chen, M. Goesele, and H. P. Seidel. Mesostructure from specularity. In *2006 IEEE Computer Society Conference on*

Computer Vision and Pattern Recognition (CVPR'06), volume 2, pages 1825–1832, 2006.

[24] Christopher B Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016.

[25] Ronald R. Coifman and Stéphane Lafon. Diffusion maps. *Applied and Computational Harmonic Analysis*, 21(1):5–30, 2006.

[26] D. Comaniciu and P. Meer. Mean shift: a robust approach toward feature space analysis. *IEEE Trans. Pattern Analysis & Machine Intelligence*, 24(5):603–619, 2002.

[27] Antonia Creswell, Tom White, Vincent Dumoulin, Kai Arulkumaran, Biswa Sengupta, and Anil A Bharath. Generative adversarial networks: An overview. *IEEE Signal Processing Magazine*, 35(1):53–65, 2018.

[28] James Davis, Stephen R Marschner, Matt Garr, and Marc Levoy. Filling holes in complex surfaces using volumetric diffusion. In *Proc. Int. Symp. on 3D Data Processing, Visualization and Transmission*, pages 428–441, 2002.

[29] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. *arXiv preprint arXiv:1712.07629*, 2017.

[30] Shay Deutsch, Antonio Ortega, and Gérard Medioni. Manifold denoising based on spectral graph wavelets. In *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, pages 4673–4677, 2016.

[31] Inderjit S. Dhillon, Yuqiang Guan, and Brian Kulis. Kernel k-means: Spectral clustering and normalized cuts. In *Proc. of ACM SIGKDD*, pages 551–556, 2004.

[32] Vincent Dumoulin and Francesco Visin. A guide to convolution arithmetic for deep learning. *arXiv preprint arXiv:1603.07285*, 2016.

[33] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proc. of ACM SIGKDD*, volume 96, pages 226–231, 1996.

[34] Pasi Franti, Olli Virmajoki, and Ville Hautamaki. Fast agglomerative clustering using a k-nearest neighbor graph. *IEEE Trans. Pattern Analysis & Machine Intelligence*, 28(11):1875–1881, 2006.

[35] Brendan J. Frey and Delbert Dueck. Clustering by passing messages between data points. *Science*, 315(5814):972–976, 2007.

[36] Yasutaka Furukawa, Carlos Hernández, et al. Multi-view stereo: A tutorial. *Foundations and Trends® in Computer Graphics and Vision*, 9(1-2):1–148, 2015.

[37] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *Computer Vision and Pattern Recognition (CVPR), 2016 IEEE Conference on*, pages 2414–2423. IEEE, 2016.

[38] Athinodoros S Georghiades, Peter N Belhumeur, and David J Kriegman. From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE Trans. Pattern Analysis & Machine Intelligence*, 23(6):643–660, 2001.

[39] Stamatios Georgoulis, Konstantinos Rematas, Tobias Ritschel, Mario Fritz, Tinne Tuytelaars, and Luc Van Gool. What is around the camera? In *Proc. IEEE Conf. on Computer Vision & Pattern Recognition*, pages 5170–5178, 2017.

[40] Stamatios Georgoulis, Konstantinos Rematas, Tobias Ritschel, Mario Fritz, Luc Van Gool, and Tinne Tuytelaars. Delight-net:

Decomposing reflectance maps into specular materials and natural illumination. *arXiv preprint arXiv:1603.08240*, 2016.

[41] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 249–256, 2010.

[42] ClÃĺment Godard, Peter Hedman, Wenbin Li, and Gabriel J. Brostow. Multi-view Reconstruction of Highly Specular Surfaces in Uncontrolled Environments. In *3DV*, 2015.

[43] Dian Gong, Fei Sha, and Gérard G Medioni. Locally linear denoising on image manifolds. In *International Conference on Artificial Intelligence and Statistics*, 2010.

[44] Ian Goodfellow. Nips 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160*, 2016.

[45] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc., 2014.

[46] G. Guennebaud and M. Gross. Algebraic point set surfaces. *ACM Trans. on Graphics (Proc. of SIGGRAPH)*, 26(3):23, 2007.

[47] JunYoung Gwak, Christopher B Choy, Manmohan Chandraker, Animesh Garg, and Silvio Savarese. Weakly supervised 3d reconstruction with adversarial constraint. In *3D Vision (3DV), 2017 Fifth International Conference on 3D Vision*, 2017.

[48] Gur Harary, Ayellet Tal, and Eitan Grinspun. Context-based coherent surface completion. *ACM Trans. on Graphics*, 33(1):5:1–5:12, 2014.

[49] Trevor Hastie, Robert Tibhsirani, and Robert Friedman. *The Elements of Statistical Learning*. Springer, 2009.

[50] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. IEEE Conf. on Computer Vision & Pattern Recognition*, June 2016.

[51] Marti A. Hearst, Susan T Dumais, Edgar Osuna, John Platt, and Bernhard Scholkopf. Support vector machines. *IEEE Intelligent Systems and their applications*, 13(4):18–28, 1998.

[52] Matthias Hein and Markus Maier. Manifold denoising. In *Advances in Neural Information Processing Systems*, pages 561–568. 2007.

[53] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei A Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. *arXiv preprint arXiv:1711.03213*, 2017.

[54] Yongjun Hong, Uiwon Hwang, Jaeyoon Yoo, and Sungroh Yoon. How generative adversarial networks and its variants work: An overview of gan.

[55] Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle. Surface reconstruction from unorganized points. *Proc. of SIGGRAPH*, pages 71–78, 1992.

[56] David W Hosmer Jr, Stanley Lemeshow, and Rodney X Sturdivant. *Applied logistic regression*, volume 398. John Wiley & Sons, 2013.

[57] Hui Huang, Dan Li, Hao Zhang, Uri Ascher, and Daniel Cohen-Or. Consolidation of unorganized point clouds for surface reconstruction. *ACM Trans. on Graphics (Proc. of SIGGRAPH Asia)*, 28(5):176:1–176:7, 2009.

[58] Hui Huang, Shihao Wu, Daniel Cohen-Or, Minglun Gong, Hao Zhang, Guiqing Li, and Baoquan Chen. l_1-medial skeleton of

point cloud. *ACM Trans. on Graphics (Proc. of SIGGRAPH)*, 32(4):65:1–65:8, 2013.

[59] Hui Huang, Shihao Wu, Minglun Gong, Daniel Cohen-Or, Uri Ascher, and Hao Richard Zhang. Edge-aware point set resampling. *ACM Trans. on Graphics*, 32(1):9:1–9:12, 2013.

[60] Rui Huang, Shu Zhang, Tianyu Li, and Ran He. Beyond face rotation: Global and local perception gan for photorealistic and identity preserving frontal view synthesis. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[61] Ivo Ihrke, Kiriakos N. Kutulakos, Hendrik P. A. Lensch, Marcus Magnor, and Wolfgang Heidrich. Transparent and Specular Object Reconstruction. *Computer Graphics Forum*, 2010.

[62] Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa. Globally and locally consistent image completion. *ACM Transactions on Graphics (TOG)*, 36(4):107, 2017.

[63] K. Ikeuchi. Determining surface orientations of specular surfaces by using the photometric stereo method. *IEEE Trans. Pattern Analysis & Machine Intelligence*, (6):661–669, 1981.

[64] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[65] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[66] Anil K. Jain. Data clustering: 50 years beyond k-means. *Pattern Recognition Letters*, 31(8):651–666, 2010.

[67] Guoliang Kang, Liang Zheng, Yan Yan, and Yi Yang. Deep adversarial attention alignment for unsupervised domain adaptation:

the benefit of target expectation maximization. *arXiv preprint arXiv:1801.10068*, 2018.

[68] Ravi Kannan, Santosh Vempala, and Adrian Vetta. On clusterings: Good, bad and spectral. *J. of the ACM*, 51(3):497–515, 2004.

[69] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *International Conference on Learning Representations*, 2018.

[70] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. *Proc. Eurographics Symp. on Geometry Processing*, pages 61–70, 2006.

[71] Michael Kazhdan and Hugues Hoppe. Screened poisson surface reconstruction. *ACM Trans. on Graphics*, 32(1):29:1–29:13, 2013.

[72] Maryam Khanian, Ali Sharifi Boroujerdi, and Michael Breuß. Photometric stereo for strong specular highlights. *Computational Visual Media*, Feb 2018.

[73] Taeksoo Kim, Moonsu Cha, Hyunsoo Kim, Jung Kwon Lee, and Jiwon Kim. Learning to discover cross-domain relations with generative adversarial networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1857–1865, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.

[74] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[75] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[76] Hans-Peter Kriegel, Peer Kröger, and Arthur Zimek. Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering. *ACM Trans. Knowl. Discov. Data*, 3(1):1:1–1:58, 2009.

[77] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[78] Carsten Lange and Konrad Polthier. Anisotropic smoothing of point sets. *Computer Aided Geometric Design*, 22(7):680–692, 2005.

[79] F. Langguth, K. Sunkavalli, S. Hadap, and M. Goesele. Shading-aware multi-view stereo. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016.

[80] Yann LeCun, Bernhard E Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne E Hubbard, and Lawrence D Jackel. Handwritten digit recognition with a back-propagation network. In *Advances in neural information processing systems*, pages 396–404, 1990.

[81] Guo Li, Ligang Liu, Hanlin Zheng, and Niloy J. Mitra. Analysis, reconstruction and manipulation using arterial snakes. *ACM Trans. on Graphics (Proc. of SIGGRAPH Asia)*, 29(5):152:1–152:10, 2010.

[82] Yangyan Li, Xiaokun Wu, Yiorgos Chrysathou, Andrei Sharf, Daniel Cohen-Or, and Niloy J Mitra. Globfit: Consistently fitting primitives by discovering global relations. *ACM Trans. on Graphics (Proc. of SIGGRAPH Asia)*, 30(4):52, 2011.

[83] Chen-Hsuan Lin, Chen Kong, and Simon Lucey. Learning efficient point cloud generation for dense 3d object reconstruction. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2018.

[84] Bruce G. Lindsay. Mixture models: theory, geometry, and applications. *NSF-CBMS regional conference series in probability and statistics*, 5:i–163, 1995.

[85] Yaron Lipman, Daniel Cohen-Or, David Levin, and Hillel Tal-Ezer. Parameterization-free projection for geometry reconstruction. *ACM Trans. on Graphics (Proc. of SIGGRAPH)*, 26(3):22:1–22:6, 2007.

[86] Guilin Liu, Duygu Ceylan, Ersin Yumer, Jimei Yang, and Jyh-Ming Lien. Material editing using a physically based rendering network. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[87] Yotam Livny, Feilong Yan, Matt Olson, Baoquan Chen, Hao Zhang, and Jihad El-sana. Automatic reconstruction of tree skeletal structures from point clouds. *ACM Trans. on Graphics (Proc. of SIGGRAPH Asia)*, 29(6):151:1–151:8, 2010.

[88] Xuequan Lu, Shihao Wu, Honghua Chen, Sai-Kit Yeung, Wenzhi Chen, and Matthias Zwicker. Gpf: Gmm-inspired feature-preserving point set filtering. *IEEE Trans. Visualization & Computer Graphics*, 2017.

[89] L. Ma, J. Stueckler, C. Kerl, and D. Cremers. Multi-view deep learning for consistent semantic mapping with rgb-d cameras. In *iros*, Vancouver, Canada, Sep 2017.

[90] Robert Maier, Kihwan Kim, Daniel Cremers, Jan Kautz, and Matthias Niessner. Intrinsic3d: High-quality 3d reconstruction by joint appearance and geometry optimization with spatially-varying lighting. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 3133–3141, 2017.

[91] Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. In *ICCV*, pages 2813–2821. IEEE, 2017.

[92] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.

[93] Abhimitra Meka, Maxim Maximov, Michael Zollhoefer, Avishek Chatterjee, Christian Richardt, and Christian Theobalt. Live intrinsic material estimation. *arXiv preprint arXiv:1801.01075*, 2018.

[94] Balint Miklos, Joachim Giesen, and Mark Pauly. Discrete scale axis representations for 3D geometry. *ACM Trans. on Graphics (Proc. of SIGGRAPH)*, 29(4):101:1–101:10, July 2010.

[95] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018.

[96] Yair Movshovitz-Attias, Takeo Kanade, and Yaser Sheikh. How useful is photo-realistic rendering for visual learning? In *European Conference on Computer Vision*, pages 202–217. Springer, 2016.

[97] Fionn Murtagh and Pierre Legendre. Ward's hierarchical clustering method: clustering criterion and agglomerative algorithm. *J. of Classification*, 31(3):274–295, 2014.

[98] Mattia Natali, Silvia Biasotti, Giuseppe Patan, and Bianca Falcidieno. Graph-based representations of point clouds. *Graphical Models*, 73:151–164, 2011.

[99] NIST. Range scans based 3D shape retrieval dataset. *http://www.itl.nist.gov/iad/vug/sharp/contest/2015/Range/*, 2015.

[100] Roy Or-El, Rom Hershkovitz, Aaron Wetzler, Guy Rosman, Alfred M Bruckstein, and Ron Kimmel. Real-time depth refinement for specular objects. In *Proc. IEEE Conf. on Computer Vision & Pattern Recognition*, pages 4378–4386, 2016.

[101] Roy Or-El, Guy Rosman, Aaron Wetzler, Ron Kimmel, and Alfred M Bruckstein. Rgbd-fusion: Real-time high precision depth recovery. In *Proc. IEEE Conf. on Computer Vision & Pattern Recognition*, pages 5407–5416, 2015.

[102] C. Öztireli, G. Guennebaud, and M. Gross. Feature preserving point set surfaces based on non-linear kernel regression. *Computer Graphics Forum (Proc. of Eurographics)*, 28(2):493–501, 2009.

[103] Mark Pauly, Niloy J Mitra, Johannes Wallner, Helmut Pottmann, and Leonidas J Guibas. Discovering structural regularity in 3D geometry. In *ACM Trans. on Graphics (Proc. of SIGGRAPH)*, volume 27, pages 43:1–43:11, 2008.

[104] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *J. of Machine Learning Research*, 12:2825–2830, 2011.

[105] Matt Pharr and Greg Humphreys. Physically based rendering, second edition: From theory to implementation. 2010.

[106] Jhony K Pontes, Chen Kong, Sridha Sridharan, Simon Lucey, Anders Eriksson, and Clinton Fookes. Image2mesh: A learning framework for single image 3d reconstruction. *arXiv preprint arXiv:1711.10669*, 2017.

[107] Reinhold Preiner, Oliver Mattausch, Murat Arikan, Renato Pajarola, and Michael Wimmer. Continuous projection for fast l_1 reconstruction. *ACM Trans. on Graphics (Proc. of SIGGRAPH)*, 33(4):47:1–47:13, 2014.

[108] Rui Qian, Robby T Tan, Wenhan Yang, Jiajun Su, and Jiaying Liu. Attentive generative adversarial network for raindrop removal from a single image. *arXiv preprint arXiv:1711.10098*, 2017.

[109] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[110] Konstantinos Rematas, Tobias Ritschel, Mario Fritz, Efstratios Gavves, and Tinne Tuytelaars. Deep reflectance maps. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[111] Elad Richardson, Matan Sela, and Ron Kimmel. 3d face reconstruction by learning from synthetic data. In *3D Vision (3DV), 2016 Fourth International Conference on*, pages 460–469. IEEE, 2016.

[112] Elad Richardson, Matan Sela, Roy Or-El, and Ron Kimmel. Learning detailed face reconstruction from a single image. In *Proc. IEEE Conf. on Computer Vision & Pattern Recognition*, pages 5553–5562. IEEE, 2017.

[113] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[114] Sam T. Roweis and Lawrence K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

[115] Szymon Rusinkiewicz and Marc Levoy. Efficient variants of the icp algorithm. In *3-D Digital Imaging and Modeling, 2001. Proceedings. Third International Conference on*, pages 145–152. IEEE, 2001.

[116] S. Savarese and P. Perona. Local analysis for 3d reconstruction of specular surfaces. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 2, pages II–738–II–745 vol.2, 2001.

[117] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Comput.*, 10(5):1299–1319, 1998.

[118] Steven M Seitz, Brian Curless, James Diebel, Daniel Scharstein, and Richard Szeliski. A comparison and evaluation of multiview stereo reconstruction algorithms. In *Computer vision and pattern recognition, 2006 IEEE Computer Society Conference on*, volume 1, pages 519–528. IEEE, 2006.

[119] Matan Sela, Elad Richardson, and Ron Kimmel. Unrestricted facial geometry reconstruction using image-to-image translation. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[120] Soumyadip Sengupta, Angjoo Kanazawa, Carlos D Castillo, and David Jacobs. Sfsnet: Learning shape, reflectance and illuminance of faces in the wild. *arXiv preprint arXiv:1712.01261*, 2017.

[121] Lee M Seversky and Lijun Yin. A global parity measure for incomplete point cloud data. In *Computer Graphics Forum (Proc. Pacific Conf. on Computer Graphics & Applications)*, volume 31, pages 2097–2106, 2012.

[122] Andrei Sharf, Marc Alexa, and Daniel Cohen-Or. Context-based surface completion. *ACM Trans. on Graphics (Proc. of SIGGRAPH)*, 23(3):878–887, 2004.

[123] Andrei Sharf, Thomas Lewiner, Ariel Shamir, and Leif Kobbelt. On-the-fly curve-skeleton computation for 3D shapes. *Computer Graphics Forum*, 26:323–328, 2007.

[124] Andrei Sharf, Thomas Lewiner, Gil Shklarski, Sivan Toledo, and Daniel Cohen-Or. Interactive topology-aware surface reconstruction. *ACM Trans. on Graphics (Proc. of SIGGRAPH)*, 26(3):43:1–43:9, 2007.

[125] Jian Shi, Yue Dong, Hao Su, and Stella X. Yu. Learning non-lambertian object intrinsics across shapenet categories. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[126] Jianbo Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. Pattern Analysis & Machine Intelligence*, 22(8):888–905, 2000.

[127] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[128] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, Andrew Rabinovich, et al. Going deeper with convolutions. Cvpr, 2015.

[129] Andrea Tagliasacchi, Ibraheem Alhashim, Matt Olson, and Hao Zhang. Mean curvature skeletons. *Computer Graphics Forum (Proc. Eurographics Symp. on Geometry Processing)*, 31(5):1735–1744, 2012.

[130] Andrea Tagliasacchi, Thomas Delame, Michela Spagnuolo, Nina Amenta, and Alexandru Telea. 3d skeletons: A state-of-the-art report. In *Computer Graphics Forum*, volume 35, pages 573–597. Wiley Online Library, 2016.

[131] Andrea Tagliasacchi, Matt Olson, Hao Zhang, Ghassan Hamarneh, and Daniel Cohen-Or. Vase: Volume-aware surface evolution for surface reconstruction from incomplete point clouds. In *Computer Graphics Forum (Proc. Eurographics Symp. on Geometry Processing)*, volume 30, pages 1563–1571, 2011.

[132] Andrea Tagliasacchi, Hao Zhang, and Daniel Cohen-Or. Curve skeleton extraction from incomplete point cloud. *ACM Trans. on Graphics (Proc. of SIGGRAPH)*, 28(3):79:1–79:9, 2009.

[133] Marco Tarini, Hendrik P. A. Lensch, Michael Goesele, and Hans-Peter Seidel. 3d acquisition of mirroring objects using striped patterns. *Graph. Models*, 67(4):233–259, July 2005.

[134] M. Tatarchenko, A. Dosovitskiy, and T. Brox. Multi-view 3d models from single images with a convolutional network. In *European Conference on Computer Vision (ECCV)*, 2016.

[135] Joshua B Tenenbaum, Vin De Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

[136] S. K. Tin, J. Ye, M. Nezamabadi, and C. Chen. 3d reconstruction of mirror-type objects using efficient ray coding. In *2016 IEEE International Conference on Computational Photography (ICCP)*, pages 1–11, May 2016.

[137] Michael E Tipping and Christopher M Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 61(3):611–622, 1999.

[138] Shubham Tulsiani, Tinghui Zhou, Alexei A. Efros, and Jitendra Malik. Multi-view supervision for single-view reconstruction via differentiable ray consistency. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[139] Greg Turk and Marc Levoy. Zippered polygon meshes from range images. In *Proc. of SIGGRAPH*, pages 311–318, 1994.

[140] Elias Vansteenkiste and Patrick Kern. Taming adversarial domain transfer with structural constraints for image enhancement. *arXiv preprint arXiv:1712.00598*, 2017.

[141] Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416, 2007.

[142] Michael Wand, Philipp Jenke, Qixing Huang, Martin Bokeloh, Leonidas Guibas, and Andreas Schilling. Reconstruction of deforming geometry from time-varying point clouds. 2007.

[143] B. Wang and Z. Tu. Sparse subspace denoising for image mani-
folds. In *Proc. IEEE Conf. on Computer Vision & Pattern Recogni-
tion*, pages 468–475, 2013.

[144] Chaoyue Wang, Chang Xu, Chaohui Wang, and Dacheng Tao.
Perceptual adversarial networks for image-to-image transfor-
mation. *arXiv preprint arXiv:1706.09138*, 2017.

[145] Jifeng Wang, Xiang Li, Le Hui, and Jian Yang. Stacked con-
ditional generative adversarial networks for jointly learning
shadow detection and shadow removal. *arXiv preprint arX-
iv:1712.02478*, 2017.

[146] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan
Kautz, and Bryan Catanzaro. High-resolution image synthe-
sis and semantic manipulation with conditional gans. *arXiv
preprint arXiv:1711.11585*, 2017.

[147] W. Wang and M. A. Carreira-Perpinan. Manifold blurring mean
shift algorithms for manifold denoising. In *Proc. IEEE Conf.
on Computer Vision & Pattern Recognition*, pages 1759–1766,
2010.

[148] Yining Wang, Yu-Xiang Wang, and Aarti Singh. A deterministic
analysis of noisy sparse subspace clustering for dimensionality-
reduced data. In *Proc. IEEE Int. Conf. on Machine Learning*,
pages 1422–1431, 2015.

[149] Yong Wang, Yuan Jiang, Yi Wu, and Zhi-Hua Zhou. Spectral
clustering on multiple manifolds. *IEEE Transactions on Neural
Networks*, 22(7):1149–1161, 2011.

[150] Alfred Weber. *Über den Standort der Industrie*. Tübingen, J.C.B.
Mohr (Paul Siebeck), 1909. English translation by C. J. Frei-
drich (1929): Alfred WeberâĂŹsTheory of the Location of In-
dustries.

[151] Katja Wolff, Changil Kim, Henning Zimmer, Christopher Schroers, Mario Botsch, Olga Sorkine-Hornung, and Alexander Sorkine-Hornung. Point cloud noise and outlier removal for image-based 3d reconstruction. In *3D Vision (3DV), 2016 Fourth International Conference on*, pages 118–127. IEEE, 2016.

[152] Jiajun Wu, Chengkai Zhang, Tianfan Xue, Bill Freeman, and Josh Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In *In Advances in Neural Information Processing Systems (NIPS)*, pages 82–90, 2016.

[153] Shihao Wu, Peter Bertholet, Hui Huang, Daniel Cohen-Or, Minglun Gong, and Matthias Zwicker. Structure-aware data consolidation. *IEEE transactions on pattern analysis and machine intelligence*, 2017.

[154] Shihao Wu, Hui Huang, Minglun Gong, Matthias Zwicker, and Daniel Cohen-Or. Deep points consolidation. *ACM Trans. on Graphics (Proc. of SIGGRAPH Asia)*, 34(6):176:1–176:13, 2015.

[155] Shihao Wu, Wei Sun, Pinxin Long, Hui Huang, Daniel Cohen-Or, Minglun Gong, Oliver Deussen, and Baoquan Chen. Quality-driven poisson-guided autoscanning. *ACM Trans. on Graphics (Proc. of SIGGRAPH ASIA)*, 2014.

[156] Wenqi Xian, Patsorn Sangkloy, Jingwan Lu, Chen Fang, Fisher Yu, and James Hays. Texturegan: Controlling deep image synthesis with texture patches. *arXiv preprint arXiv:1706.02823*, 2017.

[157] Kwang Moo Yi, Eduard Trulls, Vincent Lepetit, and Pascal Fua. Lift: Learned invariant feature transform. In *European Conference on Computer Vision*, pages 467–483. Springer, 2016.

[158] Zili Yi, Hao Zhang, Ping Tan, and Minglun Gong. Dualgan: Unsupervised dual learning for image-to-image translation. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[159] K. Yin, H. Huang, M. Gong, D. Lischinski, D. Cohen-Or, U. As-
cher, and B. Chen. Morfit: Interactive surface reconstruction
from incomplete point clouds with curve-driven topology and
geometry control. *ACM Trans. on Graphics (Proc. of SIGGRAPH
Asia)*, 33(6):202:1–202:12, 2014.

[160] Stella X Yu and Jianbo Shi. Multiclass spectral clustering. In
Proc. Int. Conf. on Computer Vision.

[161] Ye Yu and William AP Smith. Pvnn: A neural network library
for photometric vision. In *Proceedings of the IEEE Conference
on Computer Vision and Pattern Recognition*, pages 526–535,
2017.

[162] Matthew D Zeiler and Rob Fergus. Visualizing and understand-
ing convolutional networks. In *European conference on comput-
er vision*, pages 818–833. Springer, 2014.

[163] Tian Zhang, Raghu Ramakrishnan, and Miron Livny. Birch: an
efficient data clustering method for very large databases. In
Proc. of ACM SIGMOD, volume 25, pages 103–114, 1996.

[164] Yinda Zhang, Shuran Song, Ersin Yumer, Manolis Savva, Joon-
Young Lee, Hailin Jin, and Thomas Funkhouser. Physically-
based rendering for indoor scene understanding using convo-
lutional neural networks. In *The IEEE Conference on Computer
Vision and Pattern Recognition (CVPR)*, July 2017.

[165] Qian Zheng, Andrei Sharf, Guowei Wan, Yangyan Li, Niloy J.
Mitra, Baoquan Chen, and Daniel Cohen-Or. Non-local scan
consolidation for 3D urban scene. *ACM Trans. on Graphics
(Proc. of SIGGRAPH)*, 29:94:1–94:10, 2010.

[166] Yinglong Zheng, Guiqing Li, Shihao Wu, Yuxin Liu, and Yue-
fang Gao. Guided point cloud denoising via sharp feature skele-
tons. *The Visual Computer*, 2017.

[167] Yinglong Zheng, Guiqing Li, Xuemiao Xu, Shihao Wu, and Y-ongwei Nie. Rolling normal filtering for point clouds. *Computer Aided Geometric Design*, 62:16–28, 2018.

[168] Tinghui Zhou, Matthew Brown, Noah Snavely, and David G. Lowe. Unsupervised learning of depth and ego-motion from video. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[169] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[170] Rui Zhu, Chaoyang Wang, Chen-Hsuan Lin, Ziyan Wang, and Simon Lucey. Object-centric photometric bundle adjustment with deep shape prior. *arXiv preprint arXiv:1711.01470*, 2017.

[171] Rui Zhu, Chaoyang Wang, Chen-Hsuan Lin, Ziyan Wang, and Simon Lucey. Semantic photometric bundle adjustment on natural sequences. *arXiv preprint arXiv:1712.00110*, 2017.