Auto-BPA: An Enhanced Ball-Pivoting Algorithm with Adaptive Radius using Contextual Bandits

Houda Saffi [1] Naima Otberdout [1] Youssef Hmamouche [1]

Amal El Fallah Seghrouchni [1,2]

[1] Ai movement - International Artificial Intelligence Center of Morocco
University Mohammed VI Polytechnic, Rabat, Morocco
[2] Sorbonne Université, LIP6 - UMR 7606 CNRS, France
{houda.saffi, naima.otberdout, youssef.hmamouche, amal.elfallah-seghrouchni}@um6p.ma

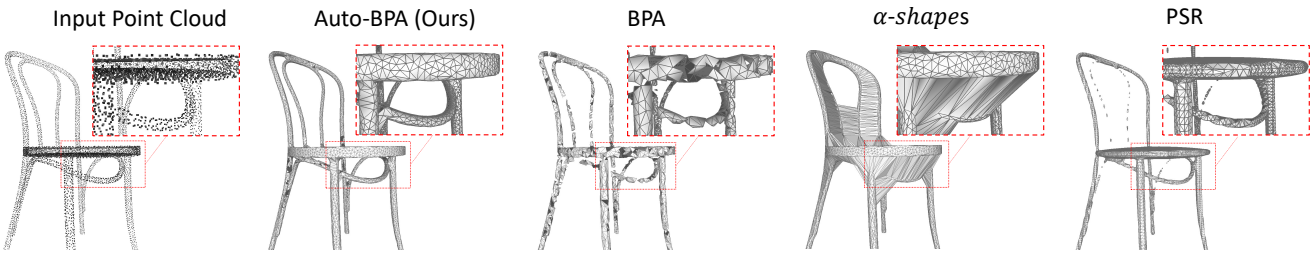

Figure 1. We introduce a method for predicting the radius parameter of the explicit reconstruction technique BPA. By utilizing contextual bandits, we predict the optimal radius for the Ball-Pivoting Algorithm. This innovative approach allows us to attain a higher-quality mesh in comparison to other state-of-the-art methods (BPA [2], α-shapes [11] and PSR [14]). Details best viewed zoomed in.

Abstract

The Ball-Pivoting Algorithm (BPA) is a notable technique for 3D surface reconstruction from point clouds, heavily reliant on the ball radius. In practical application, determining the optimal radius for BPA often necessitates iterative experimentation to achieve better reconstruction quality. BPA entails geometric computations like iterative pivoting, inherently lacking differentiability. In this paper, we tackle the dual challenges of radius selection and non-differentiability in BPA. Inspired by contextual bandits, we propose an innovative approach that learns the optimal radius based on local geometric features within point clouds. We validate our method on the ModelNet10 and ShapeNet datasets, showcasing superior surface reconstruction compared to manual tuning and other classic methods both for low and high point cloud densities. Our code is available at https://github.com/houda-pixel/Auto-BPA.

1. Introduction

Surface reconstruction is a fundamental task in computer graphics and computer vision. In recent years, the integration of learning-based methods has opened up new avenues for improving classical surface reconstruction methods, such as Poisson Surface Reconstruction [14], Dual Contouring [13], and Marching Cubes [17]. While deep learning has proven to be a transformative tool in enhancing these classical methods, it is intriguing to note that certain methods, such as the Ball-Pivoting Algorithm (BPA) [2], have yet to fully benefit from the potential offered by learning-driven approaches. This is mainly due to the non-differentiability of the BPA which presents a difficulty in effectively incorporating it into a neural network. In addition, BPA is a parameterized algorithm, its performance is significantly influenced by the value assigned to the radius parameter. Opting for a smaller radius can lead to more surface holes, while a larger radius might result in a loss of fine details. Previous works set this radius empirically either by manual intervention of the user [2] or a mathematical for-

mula that provides an approximation of the radius based on other parameters adjusted empirically [10].

To address these limitations of the BPA, we propose a novel approach that leverages the contextual bandit problem to compute the radius parameter. The idea consists of training a neural network to predict the optimal radius from the given point clouds that can lead to better reconstruction quality. The challenge here is that the value of the radius is not known beforehand, the only available information is the ground truth of the reconstructed objects, which will allow us to guide the training with the reconstruction loss, *i.e.*, the difference between the reconstructed object and its corresponding ground truth. However, this is faced because of the non-differentiability of the BPA. To overcome this limitation, we reformulated the problem in a semi-supervised manner where each selected radius for a given point cloud is considered an action. This action is determined by a reinforcement learning agent and depends on the geometry and density of the input object (observation). The agent updates its policy from the global reward defined by the reconstruction loss.

The contextual bandit framework, integral to our proposed approach, aligns with the essence of adaptive decision-making. Contextual bandit problems involve scenarios where an agent must make a sequence of decisions, each associated with an action, and is guided by contextual information that provides relevant context for each decision instance. In our context, the contextual information pertains to the features derived from our innovative integration of Fast Point Feature Histograms (FPFH) [22] and K-means clustering [12]. These contextual attributes are utilized to optimize the computation of the radius parameter. By framing radius computation as a contextual bandit problem, we unlock the abilities of reinforcement learning, enabling adaptive and contextually informed radius decisions. Thus, our aim is to elevate the capabilities of the BPA and align it with the advancements achieved by other classical methods through the integration of learning-based methods. We highlight the contributions of this paper as follows:

- We propose Auto-BPA: an improved Ball-Pivoting Algorithm. Unlike classical BPA, which sets the ball radius parameter empirically, Auto-BPA utilizes a trained RL model to automatically predict the optimal ball radius. This approach harnesses the power of intelligent decision-making to dynamically adapt radius selection based on locally extracted features from the point cloud.

- We introduce a new approach for extracting relevant features from 3D point clouds through the combination of Fast Point Feature Histograms (FPFH) and K-means clustering. This fusion enables the extraction of compact and informative feature representations, effectively capturing local geometric features present in the point cloud data.

- Through extensive experimentation, we prove the superiority of our proposed approach over other solutions for 3D surface reconstruction from point clouds. This superiority is also achieved with objects that are depicted with a restricted set of point clouds.

The remainder of this paper is organized as follows. We review the related work in Section 2. Our method is then described in Section 3. Experimental results are reported in Section 4. In Section 5, we discuss the limitations of our approach. We finally summarize our work in Section 6.

2. Related Work

Surface reconstruction from 3D point clouds is crucial in computer vision and computer graphics by creating watertight and manifold surfaces through the triangulation of provided point sets. In this section, we review methods most closely related to ours that we divided into three groups: Firstly, we review the BPA and its variant, then, we discuss the classical solutions adopted for surface reconstruction. We finish the discussion with the learning-based solutions.

BPA related methods.
The Ball-Pivoting Algorithm (BPA) [2] is a surface reconstruction method that generates a triangular mesh from point clouds. The operational framework of the algorithm involves selecting three points to serve as the vertices of a triangle if a sphere with a user-defined radius can contact these points without encompassing any additional points. The BPA is sensitive to the selected ball radius. Opting for a smaller radius can lead to more surface holes, while a larger radius might result in a loss of fine details. To mitigate this radius dependence, Bernardini et al. [2] proposed the use of multiple radii. The user specifies a sequence of radii as input parameters, initiating the process with the smallest radius. The set of boundary edges established at this initial radius serves as the basis for the rotation of the ball with progressively larger radii. However, the effectiveness of the algorithm is still reliant on manual intervention for tuning the radii values. In this context, Julie Digne [10] introduced enhancements to the algorithm, including a parallel variant that utilizes the octree data structure for faster triangulation of oriented point clouds. Julie Digne [10] also presented a formula for approximating the radius, however, this formula involves empiric coefficients. Different from these previous solutions, we propose an approach that automatically sets the radius parameters given the extracted features from the point clouds.

Classical methods for surface reconstruction.
Methods from computational geometry, *e.g.*, Delaunay triangulation [9], α-shapes [11], and ball pivoting [2] can directly output a mesh from the input point cloud. Never-

theless, explicit representations usually require meticulous parameter selection and are dependent on point sets that are uniformly and densely sampled. The Poisson Surface Reconstruction (PSR) [14] is a traditional method in implicit surface functions, however, it requires the computation of the oriented normals of the input point clouds for good performance. Marching Cubes [17] and Dual Contouring [13] use signed distance fields to extract the triangle meshes of isosurfaces.

Learning-based methods for surface reconstruction. Inspired by standard methods such as Marching Cubes (MC) [17], Dual Contouring (DC) [13], and Delaunay triangulations [9], recent learning-based approaches have been developed to leverage the properties of classical computational geometry and 3D learning. In [21], the author capitalizes on the properties of 2D Delaunay triangulations. This method involves constructing a mesh from manifold surface elements using local logmaps learned through networks. Neural Marching Cubes (NMC) [6] represents an advancement in deep learning applied to the classical MC technique. In contrast to employing coarse tessellation templates, the authors introduce innovative tessellation templates. These templates consider contextual information from neighboring cubes, achieved by learning vertex positions and mesh topologies from training meshes. Within the same framework of data-driven mesh reconstruction, Neural Dual Contouring (NDC) [5] stands out as an enhanced version of dual contouring [13]. This approach relies on a neural network for predicting vertex locations and edge crossings, avoiding the need for hand-crafted functions. While most learning approaches are based on 3D deep learning, there remains a scarcity of works that center around 3D reconstruction utilizing reinforcement learning. In [16], deep reinforcement learning is employed to acquire 3D modeling policies that mimic human modeling techniques. The approach involves a two-step neural framework, where the Prim-Agent initially estimates the shape using primitives, then, the Mesh-Agent refines the mesh to create more detailed geometry. In alignment with leveraging state-of-the-art explicit-based techniques and incorporating reinforcement learning frameworks, our research introduces an innovative approach that employs a continuous contextual bandit problem (single-step reinforcement learning) to determine the radius of the Ball-Pivoting Algorithm, a critical parameter within this classical technique.

3. Proposed Method

In this section, we describe our approach, covering the motivations behind the proposed methodology and its key stages. We start with the background related to our approach, followed by a description of the proposed solution.

3.1. Background

Fast Point Feature Histograms. Fast Point Feature Histograms (FPFH) [22] represent an advanced iteration of Point Feature Histograms (PFH) [23], primarily in terms of computational complexity. FPFH enhances the computational efficiency from $\mathcal{O}(k^2)$ to $\mathcal{O}(k)$, with k representing the number of neighbors of point p, while retaining the favorable attributes of the PFH approach. FPFH characteristics involve multidimensional features that capture the local geometric properties surrounding a point p within a given point set. These features are derived based on the geometric relationships between the point and its nearest k-neighbors. The computation of a Fast Point Feature Histogram at a specific point p necessitates the availability of 3D coordinates (x,y,z) and estimated surface normals (n_x, n_y, n_z). For every query point p_q, a set of tuples denoted as α , ϕ, and θ, describing the relationships between p_q and its neighbors, is computed in alignment with the descriptors for Point Feature Histograms (PFH) [23]. This resulting set of tuples is referred to as the Simplified Point Feature Histogram (SPFH). This process is then repeated for all points within the dataset, followed by a re-evaluation of the SPFH values of p_q, achieved by leveraging the SPFH values of its k-neighbors. This step contributes to the creation of the FPFH for p_q as mentioned in Equation 1. The entire procedure is iterated for all points in the dataset.

$$FPFH(p_q) = SPFH(p_q) + \frac{1}{k}\sum_{i=1}^{k}\frac{1}{\omega_i}.SPFH(p_i) \quad (1)$$

Contextual bandits with continuous actions. The learning policy is based on the CATS algorithm for continuous contextual bandits [18]: an innovative approach involving a smoothing method for action selection and a tree-based policy with a hierarchical structure for mapping actions to contexts. The algorithm's process involves two procedures:

1. **Tree training**: where cost-sensitive examples are processed to generate a set of tree policies. It involves training (binary) classifiers in a bottom-up fashion, routing contexts to actions with the smallest expected cost. Each tree policy is a binary tree with a depth $\mathcal{D}$. This tree comprises a total of $\mathcal{K}$ leaves each corresponding to an action, where $\mathcal{K}$ is the discretization factor defined as: $\mathcal{K} = 2^{\mathcal{D}}$. Each node $\mathcal{V}$ within the tree contains a classifier denoted as $f^{\mathcal{V}}$ and we call the set of all binary classifiers used in a tree of depth $D = log_2(K)$ as $\mathcal{F}_{\mathcal{K}}$. Eventually, we can build binary decision trees of infinite depth, which leads to an infinite set of policy classes noted $\mathcal{F}_{\infty}$. Using such structured policy classes for solving cost-sensitive multiclass classification (CSMC) problems

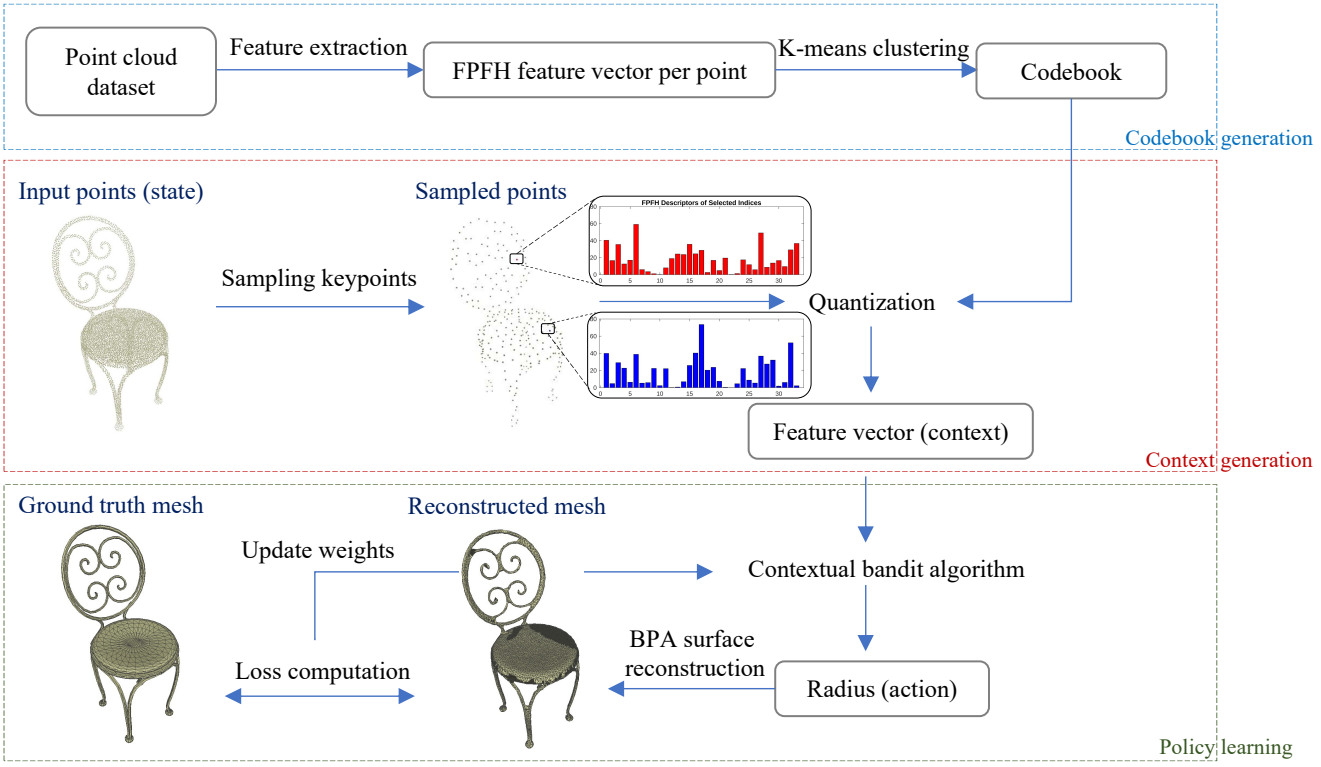

Figure 2. Pipeline of our Auto-BPA approach. It includes three stages. First, the codebook vectors are generated from an input point cloud using FPFH combined with K-means clustering. Then, the obtained features represent the input context vector of the contextual bandit algorithm which is responsible for producing the radius. In the final step, the BPA is executed with the produced radius and reconstructs the mesh. In the training phase, the policy of the contextual bandit algorithm is updated using the reconstruction loss.

can provide computational efficiency, with a running time of $O(log(K))$ per example.

2. **Policy selection**: where actions are chosen by traversing the generated policies and using supervised learners as routing functions. The ϵ-greedy strategy balances exploration and exploitation. With probability ϵ, a random action is chosen for exploration, and with probability 1 - ϵ, the action associated with the highest predicted reward is chosen for exploitation. The smoothing approach involves considering a range of actions rather than a single action. The width of this action range is determined by a parameter called the bandwidth h. The discretization strategy can still compete with policies that are not restricted to $\mathcal{A}_\mathcal{K}$, as long as the value of h is not too large, thus providing a computationally efficient approach to tackle continuous problems. Furthermore, we can prove that the error induced by the discretization can be controlled with the h parameter since the loss function is $1/h$-Lipschtiz: The concept of Lipschitz continuity refers to how much a function can change as its input changes. A function that is $1/h$-Lipschitz means that it can change by at most $1/h$ units for every unit change in its input.

These components collectively guide the algorithm's decision-making process within the contextual bandit problem with continuous actions.

3.2. Methodology

Our proposed approach combines established techniques (FPFH [22] and K-means [12]) with innovative strategies (such as the contextual bandit algorithm) to optimize the prediction of the radius parameter of the BPA. The process is outlined through the subsequent stages:

i Codebook generation: In this step, we use the Fast Point Feature Histograms (FPFH) [22] method to compute a 33-dimensional feature vector for each point in the point cloud dataset. The FPFH [22] method is effective in capturing fine details based on neighborhood, which makes it a powerful descriptor for our use case. After computing the feature vectors, we apply the K-means [12] clustering algorithm to generate cluster centers in the feature space. These cluster centers form the foundation for generating feature vectors

for each point cloud.

ii **Context generation:** Within this step, we select 100 keypoints uniformly distributed for each point cloud in the dataset. For each sampled point, we find the nearest cluster center (using Euclidean distance) and assign the cluster index to the sampled point feature vector. The combined sequence of cluster indices results in the generation of the feature vector, which is then normalized. This normalized feature vector serves as the input (context) for the subsequent phase.

iii **Policy learning:** In the final stage, the contextual bandit algorithm is employed to predict a radius value for each context. Employing the predicted radius parameter and the Ball-Pivoting Algorithm (BPA), we perform mesh reconstruction. The loss, computed via the Chamfer Distance between the reconstructed mesh and the ground truth, empowers the agent to learn the prediction of enhanced radii values, thus enhancing the quality of surface reconstruction.

The proposed approach innovatively applies the FPFH method in the context of surface reconstruction. Its advantages will be further explored in the experiments section through a comparative study with alternative surface reconstruction methodologies: The classic Ball-Pivoting Algorithm employing an empirical radius parameter, Poisson Surface Reconstruction (PSR) [14] and α-shapes [11].

3.3. Contextual bandits for BPA radius prediction

Estimating the optimal radius for the Ball-Pivoting Algorithm possesses distinctive characteristics that align seamlessly with the framework of a continuous contextual bandit problem. This framing brings to light the following specificities:

i **Contextual descriptors from point cloud:** The utilization of contextual bandits proves highly suitable due to the rich and informative features that can be derived from the point cloud data using feature extraction methods such as FPFH combined with the K-means clustering. These descriptors effectively capture the essential attributes of the point cloud, making them ideal contextual cues for making radius predictions.

ii **Non-episodic continuous actions:** Actions (the radii values) influence the reward instantly, eliminating the need for waiting until a terminal state is reached. The rewards are based on the negative Chamfer Distance between the reconstructed and ground truth meshes. Each action is independent, allowing the agent to instantly learn from its choices. This responsiveness aligns perfectly with the real-time decision-making capabilities offered by the continuous contextual bandit framework.

iii **Continuous action space:** The choice of radius for the Ball-Pivoting Algorithm falls within a continuous action space, where decisions span a range of possible values. The continuous contextual bandit setting aligns well with this aspect, as it accommodates use cases that involve making decisions within a continuous spectrum, mirroring the reality of radius selection.

iv **Learning robust policies:** The goal of framing the radius estimation as a continuous contextual bandit problem is to learn a robust policy that maps the extracted contextual features to the optimal radius. This learned policy adapts and generalizes across varying point clouds and input contexts, ensuring effective radius predictions for diverse scenarios.

We can conceptualize the task of predicting radius values as a contextual bandit problem, where the dataset of point clouds serves as the environment. In this formulation: Each single point cloud corresponds to a distinct state within the environment. The features associated with each point cloud are interpreted as contextual cues that characterize the state. Meanwhile, every radius value linked to a particular point cloud is treated as an action. Formally, we consider an environment defined by a context space X, and an action space $\mathcal{A}$. The agent engages with this environment over a sequence of time steps. In each round denoted as t, the agent observes a context x_t from the context space X, selects an action $a_t \in \mathcal{A}$, and subsequently receives as feedback from the environment a loss $\ell_t(a_t)$. The learner's objective centers on minimizing its cumulative loss $\sum_{t=1}^{T} \ell_t(a_t)$. Notably, the loss function in our case quantifies the Chamfer Distance between the sampled points from the ground truth mesh S_1, and the sampled points from the reconstructed mesh S_2, generated using the chosen action. It is given by,

$$d_{\mathrm{CD1}}(S_1, S_2) = \frac{1}{|S_1|} \sum_{x \in S_1} \min_{y \in S_2} \|x - y\|_2 + \frac{1}{|S2|} \sum_{x \in S_2} \min_{y \in S_1} \|x - y\|_2 \quad (2)$$

4. Experiments

To validate the effectiveness of our proposed approach, we conduct a comparative study involving Auto-BPA, empiric-radius-based BPA [10, 19], and classic methods such as PSR [14] and α-shapes [11]. Moreover, we compare the results obtained by the geometric-based feature extractor FPFH [22] with those of the learning-based feature extractor PointNet [20]. Additionally, we assess the ability of Auto-BPA to accurately predict radii across distributions of non-uniform point clouds.

Dataset	Number of points	Method	Surface quality							Number of missing meshes
			Overall					Sharp		
			CD1 ($\times 10^2$) ↓	CD2 ($\times 10^5$) ↓	F1↑	NC↑	NR↓	ECD1 ($\times 10^2$) ↓	EF1↑	
ModelNet10	1024	BPA1 [10]	0.8712	8.9814	0.5328	0.8612	18.8917	7.1022	0.1308	0
		BPA2 [19]	6.6658	883.2448	0.0740	0.7213	33.4799	48.3293	0.0083	898
		Auto-BPA(ours)	0.8449	8.034	0.5464	0.866	18.583	6.753	0.1382	2
		α-shapes-3% [11]	17.0603	2661.6707	0.0488	0.5391	51.3607	22.5279	0.0130	1
		α-shapes-5% [11]	1.5860	55.9103	0.4681	0.8343	22.6868	6.9556	0.1332	0
		PSR [14]	4.6530	519.1728	0.1808	0.7347	34.6196	12.8894	0.00894	0
	10000	BPA1 [10]	0.491	1.7581	0.7238	0.922	10.8112	3.6303	0.2893	0
		BPA2 [19]	0.6928	14.231	0.6903	0.9061	12.6294	4.8474	0.2771	6
		Auto-BPA(ours)	0.4818	1.7483	0.7384	0.9246	10.6315	3.6636	0.2684	0
		α-shapes-3% [11]	1.4780	71.0208	0.5725	0.8652	17.6552	4.7172	0.2228	0
		α-shapes-5% [11]	1.5449	56.3179	0.5523	0.8608	17.7502	5.8921	0.2087	0
		PSR [14]	6.6125	1649.6779	0.1041	0.7892	27.8882	37.7751	0.0022	0
ShapeNet	1024	BPA1 [10]	0.9644	15.5389	0.5060	0.8153	24.4747	6.7014	0.1165	0
		BPA2 [19]	2.6407	298.0794	0.3463	0.7816	29.4330	13.5284	0.1235	961
		Auto-BPA(ours)	0.7731	7.228	0.597	0.8408	22.001	5.536	0.1692	0
		α-shapes-3% [11]	1.9194	112.8260	0.4426	0.7587	31.6756	5.4515	0.1521	0
		α-shapes-5% [11]	1.4520	39.7176	0.4959	0.8053	25.9362	6.3746	0.1423	0
		PSR [14]	14.2130	3101.6140	0.0832	0.5530	51.6776	15.5309	0.0141	107
	10000	BPA1 [10]	0.4515	2.0824	0.7583	0.8940	14.6503	3.2356	0.3209	0
		BPA2 [19]	0.4899	3.1695	0.7750	0.8936	14.8226	3.2204	0.3835	0
		Auto-BPA(ours)	0.4209	1.385	0.7948	0.905	13.485	2.903	0.3747	0
		α-shapes-3% [11]	1.2005	39.0331	0.5920	0.8373	21.0963	4.4232	0.2173	0
		α-shapes-5% [11]	1.4309	37.6407	0.5539	0.8244	22.0679	5.7177	0.1931	0
		PSR [14]	21.5310	4424.2539	0.0220	0.4956	56.7280	28.2783	0.0055	78

Table 1. Quantitative comparison of Auto-BPA with the state-of-the-art solutions. Best results are indicated in red.

4.1. Experiment setup

Datasets. We evaluate our proposed approach using two publicly available datasets: ModelNet10 [24] and ShapeNet [4]. We employ the train/test split protocol for both datasets, aligning with methodologies employed in prior mesh reconstruction works [7, 8]. More specifically, the evaluation process for ShapeNet concentrates on eight distinct object classes, while the evaluation for ModelNet10 includes all ten classes. For validation purposes, we conducted a random sampling of 200 point clouds from the training dataset for model checkpointing. All models are normalized to the origin of the canonical frame, with a diameter of 1. This normalization process is particularly advantageous for the subsequent radius prediction, given that the action space belongs to the interval [0,1]. For each model, we employ poisson-disk sampling [3] to generate both sparse point clouds ($\sim$ 1024 points) and dense point clouds ($\sim$10,000 points) on the mesh surface. While this method ensures well-distributed point clouds, it doesn't fix the number of samples. This lack of a fixed number of samples doesn't pose a challenge to geometric feature extraction. However, when using the PointNet [20] feature extractor, the points are standardized to a size of 1024 through random replication, serving as the input point cloud. Both the FPFH and Ball-Pivoting Algorithm require normals. To calculate sample positions and normals, we utilize the face indices of each sample along with barycentric coordinates.

Evaluation metrics. Following Lie et al. [15], we assess the overall surface quality of the reconstructed meshes using Chamfer Distances (CD1, CD2), F-Score (F1), normal consistency (NC), and normal reconstruction error (NR) in degrees. Additionally, we use the Edge Chamfer Distance (ECD1) and Edge F-score (EF1) metrics to evaluate the ability of each method to preserve the sharp details on the surface. More details about these metrics can be found in the supplementary material.

Baselines. We compare our approach with classic BPA using two different parameter tuning techniques; BPA1 [10] and BPA2 [19]. We also compare our method with α-shapes [11] using $\alpha = 3\%$ and $\alpha = 5\%$ similar to previous works [15, 21]. The comparison was also performed with Poisson Surface Reconstruction (PSR) algorithm [14].

Implementation details. Concepts from contextual bandits with continuous actions can be adapted and extended to predict parameters for algorithms, such as determining the radius for the Ball-pivoting Algorithm. In the learning policy of our proposed approach, we utilize the CATX library [1]. This library provides a well-suited toolkit for addressing this challenge within the framework of continuous contextual bandits. By harnessing CATX's capability to implement custom neural network architectures and integrate them into the contextual bandit algorithm, we enhance our ability to capture direct relationships between contextual features and optimal choices for the radius. The CATX algorithm relies on several hyperparameters. We

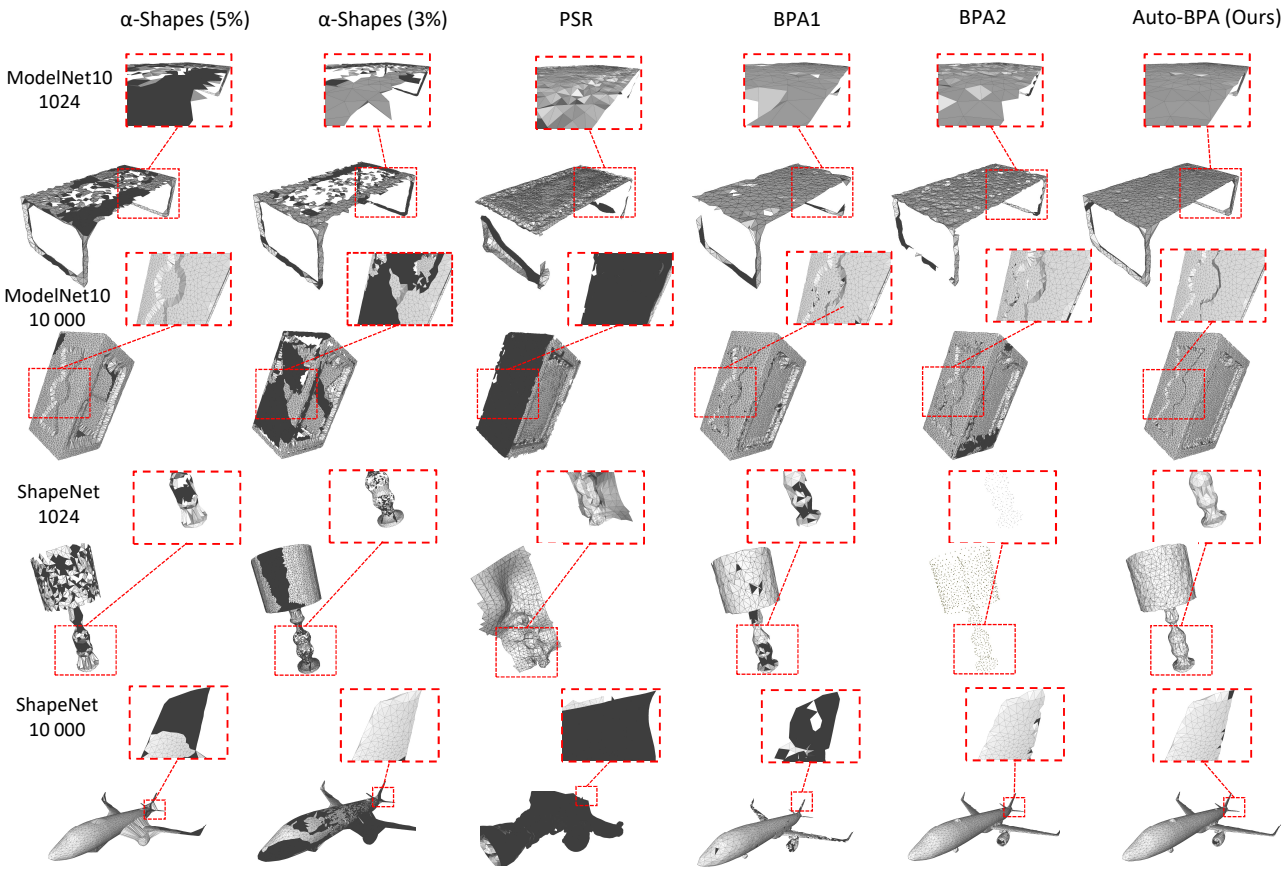

Figure 3. Qualitative results. We compare four meshes reconstructed by our method to the results of the baselines. The first two rows correspond to the ModelNet10 datasets with 1024 and 10000 points, respectively, while the last two rows correspond to the ShapeNet dataset with 1024 and 10000 points, respectively. Details best viewed zoomed in.

tune these hyperparameters on the validation set using a random search. We set the ϵ parameter in the range [0.05, 0.2] and the discretization parameter, K, to take values from $\{8, 16, 32, 64\}$. At each depth level, a neural network is employed, and the output layer dimension of each neural network is set to 2^{D+1}. The depth is set as D = 8, 16, 32, and the action space, $\mathcal{A}_{\mathcal{K}}$, is defined within the interval [0.001, 0.1]. This range is determined based on the model's predictions for radii. The Ball-Pivoting Algorithm exhibits high sensitivity to variations in radius, and thus, this interval is chosen to ensure meaningful and effective radius selections. Learning rate values are set to 0.0001, and the dropout rate varies across $\{0.1, 0.2, 0.3, 0.4, 0.5\}$. The number of steps for the ModelNet10 dataset is 1000, and for the ShapeNet dataset, it is 5000. These parameters are crucial components of our learning policy. On the other hand, for generating feature vectors, we set the codebook size as a hyperparameter. Specifically, the integer values of the codebook size are uniformly distributed within the closed interval [4, 14] for the FPFH feature extraction. In the case of the PointNet feature extractor, the feature vector size is fixed at 256.

4.2. Evaluation of mesh reconstruction

Quantitative results. Table 1 shows a quantitative comparison between our method and the baselines on ShapeNet and ModelNet10 datasets with both low (1024) and high (10000) point cloud density. The results demonstrate that our approach (Auto-BPA) significantly outperforms other baselines. On both datasets, our approach yields the lower Chamfer and squared Chamfer Distances (CD1) and (CD2) and the higher F-Score (F1) which demonstrates the high quality of the reconstructed surfaces with our approach. It also shows the high quality of the surface normals by achieving the best Normal Consistency (NC) and the lower Normal Reconstruction error (NR). The performance of our approach in preserving sharp edges is confirmed by the results obtained with ECD1 and EF1 metrics on the reconstructed meshes. While it's worth noting that a few other

approaches may marginally outperform ours in these two metrics, overall, our approach consistently show the best results.

Table 1 also indicates the number of missing meshes, which corresponds to the number of objects that the method couldn't reconstruct. Notably, our approach encountered difficulties in reconstructing two objects across both datasets, while the other methods, BPA1, PSR, and α-shapes were not able to reconstruct several objects even in the case of dense point clouds.

Qualitative results. In consistency with the quantitative results reported in Table 1, the qualitative results illustrated in Figure 3 evidence the superiority of the quality of the reconstructed meshes with our proposed approach over the other baselines. The reported examples in Figure 3 show the ability of our approach to reconstruct objects while maintaining the details of the surface. The figure demonstrates also that the other baselines, especially α-shapes and PSR suffer with surface normals in contrast to our reconstructed samples that show good normals orientations. We also observed that the reconstructed objects with the other baselines have several significant holes, whereas our approach infrequently exhibits small holes.

Robustness of Auto-BPA. We evaluate the influence of the feature extraction method and the robustness of our approach concerning various point cloud distributions. In our evaluation, we employ a set of 1024 sampled points from the ModelNet10 dataset for ablation experiments. Generally, our method is designed to excel with evenly distributed point clouds. To explore its robustness against different point cloud distributions, we subject our approach to tests using randomly sampled point clouds that are not uniformly distributed across the surface. The results presented in Table 2 demonstrate that, even with non-uniform distributions, our proposed approach continues to outperform other methods, including BPA2 and classic reconstruction techniques such as PSR, α-shapes-3%, and α-shapes-5%.

For feature extraction, we employ two distinct methods for comparison. The first method utilizes a geometric approach based on point neighborhood in the FPFH feature extraction, while the second method employs the encoder from the PointNet [20] architecture. To conduct these experiments, we train PointNet [20] on a set of 1024 sampled points from the ModelNet10 dataset. We notice under uniform distribution, the geometric method outperforms the learning-based approach. However, in cases of non-uniform distribution, the learning-based method exhibits superior performance.

5. Discussion and Limitations

The proposed approach consistently outperforms empirically-based radius selection methods, such as BPA1 and BPA2, as well as classic reconstruction approaches like PSR, α-shapes-3%, and α-shapes-5%. This performance

Points distribution	Uniform		Non-uniform	
Features extraction	PointNet	FPFH	PointNet	FPFH
CD1 ($\times 10^2$) $\downarrow$	1.0132	**0.8449**	**1.1678**	1.1720
CD2 ($\times 10^5$) $\downarrow$	13.6501	**8.034**	**16.8203**	17.1958
F1$\uparrow$	0.4899	**0.5464**	**0.4392**	0.4371
NC$\uparrow$	0.8481	**0.866**	0.8240	**0.8247**
NR$\downarrow$	20.3805	**18.583**	24.1916	**24.0598**
ECD1 ($\times 10^2$) $\downarrow$	8.2686	**6.753**	**8.1219**	8.3293
EF1$\uparrow$	0.1090	**0.1382**	**0.0611**	0.0588

Table 2. Comparative evaluation of Auto-BPA performance with regard to points distribution and feature extraction methods.

can be attributed to the detailed features that describe the geometric properties of point sets. Additionally, the smoothing approach of the CATS algorithm enhances the selection of robust parameters, accommodating variations in point cloud sampling levels. The hierarchical structure, based on tree policies, plays a pivotal role in effectively mapping diverse actions to distinct contexts. The radii predictions generated by our model underscore the sensitivity of the Ball-Pivoting Algorithm to variations in radii. Through our proposed approach, manual tuning of user-selected radii is optimized. A promising aspect of our method is its potential application beyond empirically chosen radii. Instead of relying on a predetermined list of radii choices in the case of the BPA with multiple passes, the proposed approach holds promise in setting radii values after density-based post-processing of point clouds. The model will assign the optimal radius to each single patch of points. In such cases, the model effectively will surpass the challenge of non-uniform point set distribution and will be efficient in the case of 3D scene reconstructions that are characterized by varying point cloud densities, noise, and outliers.

6. Conclusion

In conclusion, the innovative approach presented in this study signifies a notable advancement in optimizing radii value prediction for the Ball-Pivoting Algorithm. Through the fusion of geometric-based feature extraction and the capabilities of the continuous contextual bandit algorithm, we showcase a technique with significant potential. This method not only enhances radii prediction accuracy but also broadens its applicability across diverse reconstruction techniques reliant on a user-driven parameter selection process, such as the depth parameter in the Poisson Surface Reconstruction. Thus, our innovative framework represents the strength of merging geometric insights with advanced learning approaches, offering a pathway toward enhanced accuracy, and user experience in the field of 3D reconstruction.

References

[1] Wissam Bejjani and Cyprien Courtot. Catx: contextual bandits library for continuous action trees with smoothing in jax, 2022. 6

[2] Fausto Bernardini, Joshua Mittleman, Holly Rushmeier, Cláudio Silva, and Gabriel Taubin. The ball-pivoting algorithm for surface reconstruction. *IEEE transactions on visualization and computer graphics*, 5(4):349–359, 1999. 1, 2

[3] John Bowers, Rui Wang, Li-Yi Wei, and David Maletz. Parallel poisson disk sampling with spectrum analysis on surfaces. *ACM Transactions on Graphics (TOG)*, 29(6):1–10, 2010. 6

[4] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 6

[5] Zhiqin Chen, Andrea Tagliasacchi, Thomas Funkhouser, and Hao Zhang. Neural dual contouring. *ACM Transactions on Graphics (TOG)*, 41(4):1–13, 2022. 3

[6] Zhiqin Chen and Hao Zhang. Neural marching cubes. *ACM Transactions on Graphics (TOG)*, 40(6):1–15, 2021. 3

[7] Angela Dai and Matthias Nießner. Scan2mesh: From unstructured range scans to 3d meshes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5574–5583, 2019. 6

[8] Rangel Daroya, Rowel Atienza, and Rhandley Cajote. Rein: Flexible mesh generation from point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 352–353, 2020. 6

[9] Mark de Berg, Otfried Cheong, Marc van Kreveld, and Mark Overmars. Delaunay triangulations: height interpolation. *Computational geometry: algorithms and applications*, pages 191–218, 2008. 2, 3

[10] Julie Digne. An analysis and implementation of a parallel ball pivoting algorithm. *Image Processing On Line*, 4:149–168, 2014. 2, 5, 6

[11] Herbert Edelsbrunner and Ernst P Mücke. Three-dimensional alpha shapes. *ACM Transactions On Graphics (TOG)*, 13(1):43–72, 1994. 1, 2, 5, 6

[12] Greg Hamerly and Charles Elkan. Learning the k in k-means. *Advances in neural information processing systems*, 16, 2003. 2, 4

[13] Tao Ju, Frank Losasso, Scott Schaefer, and Joe Warren. Dual contouring of hermite data. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, pages 339–346, 2002. 1, 3

[14] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, volume 7, page 0, 2006. 1, 3, 5, 6

[15] Huan Lei, Ruitao Leng, Liang Zheng, and Hongdong Li. Circnet: Meshing 3d point clouds with circumcenter detection. *ICLR*, 2023. 6

[16] Cheng Lin, Tingxiang Fan, Wenping Wang, and Matthias Nießner. Modeling 3d shapes by reinforcement learning. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part X 16*, pages 545–561. Springer, 2020. 3

[17] William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. *Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, 1987. 1, 3

[18] Maryam Majzoubi, Chicheng Zhang, Rajan Chari, Akshay Krishnamurthy, John Langford, and Aleksandrs Slivkins. Efficient contextual bandits with continuous actions. *Advances in Neural Information Processing Systems*, 33:349–360, 2020. 3

[19] Alessandro Muntoni and Paolo Cignoni. Pymeshlab. *Zenodo*, 2021. 5, 6

[20] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017. 5, 6, 8

[21] Marie-Julie Rakotosaona, Paul Guerrero, Noam Aigerman, Niloy J Mitra, and Maks Ovsjanikov. Learning delaunay surface elements for mesh reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22–31, 2021. 3, 6

[22] Radu Bogdan Rusu, Nico Blodow, and Michael Beetz. Fast point feature histograms (fpfh) for 3d registration. In *2009 IEEE international conference on robotics and automation*, pages 3212–3217. IEEE, 2009. 2, 3, 4, 5

[23] Radu Bogdan Rusu, Nico Blodow, Zoltan Csaba Marton, and Michael Beetz. Aligning point cloud views using persistent feature histograms. In *2008 IEEE/RSJ international conference on intelligent robots and systems*, pages 3384–3391. IEEE, 2008. 3

[24] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015. 6