Point Cloud Filtering via Generalized Gaussian Mixture Model

Chunzhi Gu, Kota Ushiwaka, Shogo Tokai, Chao Zhang*

University of Fukui, 3-9-1, Bunkyo, Fukui, 910-8507, Japan

*Corresponding Author: zhang@u-fukui.ac.jp

Abstract

3D point cloud is extensively utilized to reconstruct the surface of the object. However, in many real-world situations, the acquired point cloud is corrupted by noise due to the insufficient precision of the measuring instruments. To filter the noise, a typical conventional method showing promising results attempts to capture the probabilistic distribution of the data with Gaussian mixture model (GMM). However, the assumption that the data only complies with Gaussian distribution restricts its performance if the data is non-Gaussian. In this paper we propose to filter the noise by generalized Gaussian mixture model (GGMM). With the shape parameter in GGMM determining the type of probabilistic model of each component, GGMM is able to capture the underlying distribution of the point cloud more accurately regardless of the non-Gaussianity of the data, which leads to a better performance in preserving geometric features while filtering the point cloud. Experiment with synthetic noisy data demonstrates the effectiveness of the proposed method both qualitatively and quantitatively.

Keywords: GMM, GGMM, Point Cloud Filtering.

1. Introduction

The reconstruction of the surface of the object usually depends on the 3D point cloud. However, the quality of the acquired point cloud deteriorates due to the noise or outliers caused by the insufficient precision of the measuring instruments. As a result, it is impossible to successfully reconstruct the surface under such a noisy input data simply because the geometric features are completely altered, as illustrated in Fig. 1.

The core problem of point cloud filtering is to determine the projection of noisy inputs to the underlying surface. In an effort to generate the desired surface, many methods\(^{1,2,3}\) for point cloud filtering has been studied, which aim to hold the geometric features and simultaneously the smoothness. LOP\(^1\) (locally optimal projection) operator is a highly efficient approach due to its property of parameterization-free. Nevertheless, it fails to handle the extremely non-uniform data. EAR\(^2\) (edge aware resampling) extends the traditional LOP method by putting emphasis on the geometric features. However, the large neighborhood size it uses may exclude points from the edge. Among those methods GPF\(^2\) shows the best result. GPF\(^2\) assumes that the data follows Gaussian mixture distributions, and it utilizes Expectation Maximization (EM) algorithm\(^4\) to estimate the parameters of each Gaussian component. The noise or outliers are considered less dominant to each Gaussian component, and they will be smoothed out during the iterations of EM algorithm. Finally, the centroids of the Gaussian components are the filtered points. Also, a novel repulsion term is added to the objective function in the M-step to alleviate the over-aggregation issue when updating parameters. Despite the outstanding performance of GPF\(^2\), it still suffers from two major issues: (1) the input noisy data may not well follow the Gaussian distribution, and under this condition the result by GMM can deviate greatly from the true underlying distribution of the data; (2) geometric features may lost when large noise occurs.

In this paper we propose to use a more flexible probabilistic model - generalized Gaussian mixture model

Fig. 1. The reconstructed surface (right) using noisy input (left).

DOI: 10.12792/icisp2019.018 © 2019 The Institute of Industrial Applications Engineers, Japan.
(GGMM) \(^{(8,9,10)}\) as an alternative. The additional shape parameter in GGMM contributes to its capability and flexibility to approximate a variety of probabilistic models, which can better handle the non-Gaussian data by capturing the true underlying distribution comparing with GMM.

The remaining part of this paper is summarized as follows: First we review the existing approach GPF for better readability of our proposed method. Secondly, we introduce our improvement by utilizing GGMM to the point set filtering task. Eventually, we show the superiority of our method by comparing it against GMM based method.

2. Review of GPF

2.1 Bilateral Smoothing with 3D Point Normals

In the process of filtering noise, the distance between two points is the point-to-plane distance, where the plane is given by the normal of each point. Therefore, to accurately calculate the distance, it is necessary to apply smoothing to the normal of the input data, which is realized by bilateral normal smoothing\(^{(1)}\).

Let the set of the input be \(S = \{s_i\}_{i \in I} = \{(x_i, n_i) \in \mathbb{R}^6\}\), where \(x_i\) and \(n_i\) represent the coordinate and normal vector of \(s_i\) respectively. \(I\) denotes the set of point indices. The bilateral smoothing of point normals is defined as:

\[
\begin{align*}
n_i^\text{new} &= \frac{\sum_{j \in N_{\delta_{x_i}}} \phi(n_i, x_i - x_j) \psi(n_j, n_j) n_j}{\sum_{j \in N_{\delta_{x_i}}} \phi(n_i, x_i - x_j) \psi(n_j, n_j)}, \\
\phi(n_i, q_i - q_j) &= e^{-\frac{n_i^\top [p_i - p_j] n_j}{\sigma_p^2}} \phi(n_i, n_j) = e^{-\frac{n_i^\top [p_i - q_j] n_j}{\sigma_n^2}},
\end{align*}
\]

where \(N_{\delta_{x_i}} = \{s_j \mid s_j \in S \land ||x_i - x_j|| < \sigma_p\}\) is the set of neighbor points under the neighborhood size \(\sigma_p\). \(\phi(\cdot)\) and \(\psi(\cdot)\) are two weight functions controlling smoothing in terms of spatial correspondence and dependency on normals. \(\phi(\cdot)\) is defined as the height difference of point \(q_j\) over the tangent plane of the point \(p_i\), and \(\psi(\cdot)\) takes the intersection angle between normals of two points as input. \(\sigma_p\) and \(\sigma_n\) control the degree of smoothness of two weight functions.

Fig. 2 illustrates the smoothed normals by iteratively applying Eq. (1).

2.2 GPF

The key idea of GPF is to model the underlying distribution of unorganized noisy input point cloud \(X, X = \{x_1, x_2, ..., x_N\}\), under the GMM framework.

A GMM can be formulated as

\[
p(x_n) = \frac{1 - \omega}{M} \sum_{m=1}^{M} p_m(x_n | \mu_m, \sigma) + \frac{\omega}{N},
\]

where \(p_m(x | \mu_m, \sigma) = \frac{1}{(2\pi \sigma^2)^{d/2}} e^{-\frac{||x - \mu_m||^2}{2\sigma^2}}\) is the \(m\)-th Gaussian component, and \(P = \{\mu_1, \mu_2, ..., \mu_M\}\) is drawn sampled from \(X\) to initialize GMM as centroids. \(D\) is the dimensionality of the input data \((D = 3\) in this work). Note that each Gaussian component is revised similar to Myronenko et al.\(^{(5)}\) with a uniform distribution, i.e., \((1/N,\) for noise and outliers, which is weighted with \(\omega \in [0,1]\), and the equal isotropic covariances \(\sigma^2\) are introduced rather than the usual covariance matrix.

The centroids and covariances of the GMM can be estimated by maximizing log-likelihood function\(^{(4)}\)

\[
E(P, \sigma^2) = \sum_{n=1}^{N} \log \left( \frac{1 - \omega}{M} \sum_{m=1}^{M} p_m(x_n | \mu_m, \sigma) + \frac{\omega}{N} \right),
\]

and the expectation maximization (EM) algorithm is used to maximize it. The EM algorithm consists of two steps: E-step and M-step. They are alternately performed for multiple iterations to achieve decent estimations.

**E-step.** The posterior probability \(p_{\text{old}}(\mu_m | x_n)\) is calculated based on Bayes’ theorem and the parameters in the previous iteration. \(p_{\text{old}}^{mn}\) represents \(p_{\text{old}}(\mu_m | x_n)\) for simplicity,

\[
p_{mn}^{\text{old}} = \frac{e^{-\frac{||x_n - \mu_m||^2}{2\sigma^2}}}{\sum_{m'=1}^{M} e^{-\frac{||x_n - \mu_{m'}||^2}{2\sigma^2}} + (2\pi \sigma^2)^{d/2} M} \frac{(2\pi \sigma^2)^{d/2} \omega^2 M}{(1 - \omega)N}.
\]

**M-step.** The M-step is to update the involved parameters \((P\) and \(\sigma^2\)) based on the computed posteriors.

\[
E(P, \sigma^2) = \frac{1}{2\sigma^2} \sum_{n=1}^{N} \sum_{m=1}^{M} p_{mn} ||x_n - \mu_m||^2 + \frac{DN}{2} \log \sigma^2,
\]
This is equivalent to maximizing the upper bound of Eq. (3), where $N_p = \sum_{n=1}^{N} \sum_{m=1}^{M} p_{mn}$.

In GPF, $\|x_n - \mu_m\|^2$ is replaced with $||(x_n - \mu_m) b_n^T||^2$, in which $b_n$ is the filtered normal to each point to preserve geometric features. Let $X'$, $X' = \{b_1, b_2, \ldots, b_N\}$ be the smoothed normal, and this modified term is defined as the data term $D(P, \sigma^2)$, shown as follows:

$$D(P, \sigma^2) = \frac{1}{2\sigma^2} \sum_{n=1}^{N} \sum_{m=1}^{M} P_{mn} ||(x_n - \mu_m) b_n^T||^2.$$  \hspace{1cm} (6)

Besides the aforementioned improvement, an additional repulsion term $R(P)$ is added to the objective function Eq. (5) in M step to generate repulsive force, similar to GMM:

$$R(P) = \sum_{m=1}^{M} \sum_{m'=1, m' \neq m}^{M} \eta(|r_{mm'}|) \theta(r_{mm'}),$$  \hspace{1cm} (7)

where $r_{mm'} = ||(\mu_m - \mu_{m'}) - (\mu_m - \mu_{m'}) b_m^T b_{m'}||$, and $b_m$ is the normal vector of $\mu_m$. $R(P)$ consists of the repulsion function $\theta(r) = -r$ and the smoothly decaying function $\eta(r) = e^{-r^2/(h^2)}$, in which $h \in [3h_0, 12h_0]$ denotes the support radius and $h_0 = \frac{d_{bb}}{\sqrt{c}}$. $d_{bb}$ is the diagonal length of the bounding box of the input point set.

With the introduction the repulsion term, the issue that down-sampling is biased or the points get over-aggregated in the process of EM optimization can be avoided, and thus the final objective function can be written as

$$E(P, \sigma^2) = D(P, \sigma^2) + R(P) + \frac{DN_p}{2} \log \sigma^2.$$  \hspace{1cm} (8)

Finally, the $\mu_{new}$ and $(\sigma^2)_{new}$ can be updated as follows:

$$\mu_{new}^m = \mu_{old}^m + \frac{\sum_{n=1}^{N} P_{mn} / \rho_n (x_n - \mu_{old}^m) b_n^T} {\sum_{n=1}^{N} P_{mn} / \rho_n}$$  \hspace{1cm} (9)

$$(\sigma^2)_{new}^m = \frac{1}{DN_p} \sum_{n=1}^{N} \sum_{m=1}^{M} P_{mn} ||(x_n - \mu_{new}^m) b_n||^2,$$  \hspace{1cm} (10)

where $C_{mm'} = 1 - b_m^T b_{m'}$, $\beta_{mm'} = \frac{\theta(r_{mm'})}{r_{mm'}}$, $\rho_n = 1 + \sum_{n'=1, n' \neq m}^{N} \theta(||x_n - x_{n'}||)$ and $w_{mn} = 1 + \sum_{m'=1, m' \neq m}^{M} \theta(||x_m - x_{m'}||)$ are the density weights of the input point and the projected point respectively. $\alpha$ is a constant determined by the user. Interested readers can refer to GPM for more details for derivation.

**3. Proposed Method**

In this section we describe our proposed method: point cloud filtering based on GGMM. We first formulate the filtering problem using generalized Gaussian mixture model (GGMM), then we show the derivation of parameters involved using gradient descent and the Newton’s method.

**3.1 Generalized Gaussian**

The generalized Gaussian is a probabilistic distribution that includes an additional shape parameter $c$ comparing with the Gaussian distribution.

$$p_m(x|\mu_m, \sigma, c_m) = \frac{K}{\sigma^D |\beta|^{1/2}} \exp \left(-\sum_{d=1}^{D} \frac{||a_d||^c}{d!}\right),$$  \hspace{1cm} (11)

$$K = \frac{\Gamma(1/c_d)}{\Gamma(1)} \beta,$$  \hspace{1cm} (11)

$$a_d = \sigma^{-1} \frac{\Gamma(3/c_d)}{\Gamma(1/c_d)} (x_d - \mu_d),$$  \hspace{1cm} (11)

where $\mu_m$, $\sigma$, and $c_m$ are the mean, the scale and the shape parameters of the $m$-th class-conditioned distribution respectively. $\Gamma(x)$ denotes the Gamma function. The adjustment to the shape parameter only can directly contribute to a wide range of probabilistic distributions, as plotted in Fig. 3.

**3.2 Generalized Gaussian Mixture Model**
The shape parameter provides huge flexibility and robustness that mere Gaussian distribution can hardly achieve. This major advantage covers the drawback in the assumption by GPF that the input only follows Gaussian distributions. We thus formulate the problem under GGMM framework with input $X$,

$$p(x_n) = \frac{1 - \omega}{M} \sum_{m=1}^{M} p_m(x_n|x_m, \mu, \sigma, c_m) + \frac{\omega}{N}.$$  \hfill (12)

Similar to GPF, the objective function is improved by adding the repulsive term:

$$E(P, \sigma, C) =$$

$$\sum_{n=1}^{N} \log \left(\frac{1 - \omega}{M} \sum_{m=1}^{M} p_m(x_n|x_m, \mu, \sigma, c_m) + \frac{\omega}{N} \right) + R(P, X'),$$

and we utilize EM algorithm to estimate the parameters (i.e., $P, \sigma, C = \{c_1, c_2, \ldots, c_M\}$) in GGMM.

In E-step, we calculate the posterior probability $p_{old}(\mu_m|x_n)$, and represent it as $p_{mn}$ for simplicity,

$$p_{mn} = \frac{p_m(x|x_m, \mu, \sigma, c_m)}{\sum_{m'=1}^{M} p_{m'}(x|x_{m'}, \mu, \sigma, c_{m'}) + \frac{M\omega}{N(1 - \omega)}}.$$ \hfill (14)

And in M-step, the calculated $p_{mn}$ is used to update parameters. We take the partial derivative of Eq. (13) with respect to $x, y, z$ axes of $\mu_m$ respectively, take $x$ as an example:

$$D_{m_x} = \sum_{n=1}^{N} \frac{p_{mn}}{p_m} \text{sgn}(v_{n_x}) c_{m_x} (\Gamma(3/c_{m_x})/\Gamma(1/c_{m_x}))^{c_{m_x}/2} ||v_{n_x}||^{c_{m_x}-1}.$$ \hfill (15)

$$v_{n_x} = (x_n - \mu_{n_x})b_{n_x}b_{n_x},$$

where $\text{sgn}(\cdot)$ denotes the sign function that returns -1 or 1 regarding the input being negative or positive. Given the computational burden imposed by the calculation of Hessian matrix from the repulsive term when using Newton’s method, we choose to use gradient descent to lower the processing time. The $\mu_m$ is updated as follows:

$$\mu_{m'}^{\text{new}} = \mu_{m'}^{\text{old}} + \frac{1}{\sum_{n=1}^{N} p_{mn}/p_m} \left(\frac{D_{m_x}}{D_{m_y}}\right)$$

$$+ \alpha \sum_{m'=1,m'\neq m}^{M} \beta^{k}_{mm'} \gamma^{k}_{m'} (\mu_{m'} - \mu_{m}) C^{k}_{mm'}.$$ \hfill (16)

The same partial derivation operation is taken with respect to $\sigma$ of Eq. (13). Note that we organize the updating of $\sigma$ to be linear similar to GPF for less computational cost, and achieve

$$\sigma_{n}^{\text{new}} = \frac{1}{DN} \sum_{n=1}^{N} \sum_{m=1}^{M} p_{mn} \left(\sum_{d=1}^{D} |\alpha_{d} |^{c_{d}/2} \right).$$ \hfill (17)

$$\alpha_{d} = \frac{\Gamma(3/c_{d})}{\Gamma(1/c_{d})} (x_d - \mu_{d})b_{d}$$

The update of the last parameter $\sigma$ is via the Newton’s method. The Newton’s method requires both first and second derivative,

$$c_{m_x} = \sum_{n=1}^{N} p_{mn} \left(\frac{1}{c_{m_x}} + \frac{1}{c_{m_x}^2} \psi \left(\frac{1}{c_{m_x}}\right) - (\sigma^{-1} a'_{m_x})^2 \log(\sigma^{-1} a_{m_x}') \right)$$ \hfill (18)

and

$$c_{m'}^{\text{new}} = \sum_{n=1}^{N} p_{mn} \left(\frac{1}{c_{m'}^{\text{old}}} - \frac{2c_{m'}^{3}}{\psi' (1/c_{m'})} - \frac{1}{c_{m_x}} \frac{1}{c_{m_x}} \psi' (1/c_{m_x}) \right) - (\sigma^{-1} a'_{m_x})^2 \log(\sigma^{-1} a_{m_x}')^2, \quad \text{for } m' \neq m.$$ \hfill (19)

where $\psi(\cdot)$ and $\psi'(\cdot)$ are digamma function and polygamma function respectively.

Thus the update of $C$ by Newton’s method is finally presented as

$$c_{m}^{\text{new}} = c_{m}^{\text{old}} - c_{m'}^{\text{old}}.$$ \hfill (20)

Our proposed point cloud filtering is the summarized in Alg. 1

**Algorithm 1 Point cloud filtering via GGMM**

**Input:** Noisy point cloud $X$, Smoothed normal $X'$

**Output:** Centroids $P$

1. Parameter settings: $\omega = 0.02, \sigma = 0.0004, h \in [5h_0, 12h_0], \alpha \in [0.05, 0.30], C = 2.0$ for all component, downsample $X$ to $P$.
2. **repeat**
3. **E-step:** update $p_{mn}$ by Eq. 14.
4. **M-step:** update $P$, $\sigma$ and $C$ by Eq. (15), Eq. (16) and Eq. (17).
5. **until** convergence or the maximum iterations are reached.
6. **return** $P$

**4. Experiment**

In this section we report the result on point cloud filtering. We compare our proposed method against the conventional method (i.e., GPF) on synthetic data both visually and quantitatively.
4.1 Experiment Setup

The point cloud is manually generated by random vertex displacement implemented by MeshLab*, and is added with noise by the ratio of 0.5, 1.0, 1.5%. We first apply bilateral smoothing to the normal via Eq. (1) to compute the distance, shown in Fig. 4(e)-(g). Note that all the normal are

* http://www.meshlab.net/
smoother and variances of the noisy input.

For the purpose of fair comparison with GPF, we adopt the following principles. (1) The downsampled points $P$ to initialize both methods are the same, and the number of downsampled points is set to be $M = N/2$ for both methods. (2) All parameters involved in both methods are carefully tuned to ensure the best performance. (3) Termination criteria for parameter estimation is the number of iteration reaches 100 for both methods.

4.2 Experimental result

The visual experimental results are illustrated in Fig. 5 and Fig. 7. The error bar exhibits visualization of distance error between each point and their nearest ground truth point. It is worth pointing out that in the case of noise ratio 0.5 and 1.0, the projected points in Fig. 5(d) and Fig. 6(d) show more promising results by GGMM than GPF. As we can observe, the points adhere tighter to the edges and there exists less outliers in the case of GGMM. This also leads to the better reconstruction accuracy in Fig. 5(f) and Fig. 6(f). Also, the geometric features around the edges are better preserved by GGMM since the color mapping presents more “blue-like” and “less-red” areas by GGMM than GPF, which also indicates that the filtering results around discontinuities (i.e., sharp edges) by GGMM is slightly more sensitive.

The quantitative evaluation is by calculating the average distance between points from the ground truth and their nearest points in the reconstructed point set. As summarized in Tab. 1, the reconstructed surface based on GGMM outperforms GPF, which further shows the superiority of the shape parameter in capturing the true underlying distribution of the noisy input.

Nevertheless, the proposed GGMM also suffers from limitations. For example, in the case of large noise (Fig. 7), both methods fail because large noise leads to an unreliable normal estimation when applying bilateral smoothing. Also, severe noise can negatively affect the proper estimation of parameters of some probabilistic components.

5. Conclusions

In this paper we proposed to use generalized Gaussian mixture model to capture the probabilistic distribution of noisy 3D point cloud to accomplish the filtering task. We improve the existing method by substituting GGMM for the conventional Gaussian mixture model to overcome the limitation of the non-Gaussianity of the data. Experimental results on synthetic data demonstrate the better performance of our method especially in preserving the geometric features around the edges. However, both methods do not perform well with the existence of large noise. As the future work, we would like to replace the current isotropic covariance with the anisotropic covariance matrix for dealing with the more complex data distribution.

References


(3) Lipman, Yaron and Cohen-Or, Daniel and Levin, David