IEEE/ EG Symposium on Volume and Point-Based Graphics (2008)
H.- C. Hege, D. Laidlaw, R. Pajarola, O. Staadt (Editors)

A Survey of Methods for Moving Least Squares Surfaces

Z.-Q. Cheng†, Y.-Z. Wang, B. Li, K. Xu, G. Dang, and S.-Y. Jin

PDL laboratory, National University of Defense Technology, P.R. China

Abstract

Moving least squares (MLS) surfaces representation directly defines smooth surfaces from point cloud data, on which the differential geometric properties of point set can be conveniently estimated. Nowadays, the MLS surfaces have been widely applied in the processing and rendering of point-sampled models and increasingly adopted as the standard definition of point set surfaces. We classify the MLS surface algorithms into two types: projection MLS surfaces *and* implicit MLS surfaces*, according to employing a stationary projection or a scalar field in their definitions. Then, the properties and constrains of the MLS surfaces are analyzed. After presenting its applications, we summarize the MLS surfaces definitions in a generic form and give the outlook of the future work at last.*

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Curve, surface, solid, and object representation

1. Introduction

The problem of defining surfaces out of point samples has been actively researched for many years. With the development of Point-Based Graphics [KB04][GP07], it is natural to consider the more general approach of defining surfaces directly from point set. Point set surfaces, proposed by Alexa et al. [ABCO*01], is the seminal paper in this area. It presents how a simple and effective representation could be achieved by the use of moving least squares (MLS) [MaL76][Lev03] technique. And now, the MLS surface is being increasingly adopted as the standard definition of point set surfaces. In addition to the principal smoothing representing function, the advantages of the MLS surfaces have many others, such as the intrinsic ability to handle noisy input, the simplicity to compute the differential geometric properties of the surface (e.g., normal, curvature).

Until now, there are many variations and extensions to the original MLS surface approach [ABCO*01]. According to whether the definition is treated as a fixed projection operator or a scalar field, existing algorithms can be mainly classified into two categories (Section 3): *projection MLS surfaces* and *implicit MLS surfaces*. A *projection MLS surface* is defined as a set of stationary points of a projecting operator, while an *implicit MLS surface* is defined by the zero isosurface of a level set function.

However, original approaches sometimes exhibit undesirable behaviors for sharp features and complex output meshes. So, Amenta and Kil [AK04b] raise the analysis of the properties and constraints of the MLS surfaces. The basic practical and theoretical issues, e.g., the domain of the MLS surfaces, sampling condition, and the limitations in handling sharp features, are further discussed in Section 4.

The applications of the MLS surfaces, such as other reconstructions, processing (e.g., deformation, animation, similarity marching, editing, and simplification), and the high-quality and efficient display of the MLS surfaces, are presented in Section 5. Then, we summarize the MLS surfaces definitions in a generci form (Section 6), and give the outlook of the further work at last.

2. Preliminaries

Problem Formulation. Suppose that the discrete point set $\mathcal{P} = \{p_i \in \mathbf{R}^D, D = 3\}$, $i \in \{1, 2, \ldots, n\}$, p_i is the positional information, is sampled from an unknown surface $\mathcal{S}$. The goal of the MLS surfaces is to find a computational method, which directly defines a reconstructed surface from $\mathcal{P}$. Assuming that the surface function $f(x)$ is defined in arbitrary parameter domain Ω, which approximates the given scalar values f_i for a moving point $x \in \mathbf{R}^D$ in the MLS sense

† cheng.zhiquan@gmail.com

[MaL76][Nea04]. f is taken from $\prod_r^D$, the space of polynomials with total degree r in D spatial dimensions. The idea is to start with a weighted least squares (WLS) formulation $f'(x)$ for an arbitrary fixed point in $\mathbf{R}^D$, and then move this point over the entire parameter domain, where a WLS fit is evaluated for each point individually. Then, the fitting function $f(x)$ is obtained from a set of local approximation functions $f'(x)$.

$$f(x) = argmin_{f' \in \prod_r^p} \sum_{i=1}^{n} w_i(\|x-p_i\|)\|f'(p_i) - f_i\| \quad (1)$$

$$f'(x) = \mathbf{g}^T(x)\mathbf{c}(x) = \sum_{j\in[1,m]} g_j(x)c_j(x) \quad (2)$$

then,$f(x)$can be expressed as

$$f(x) = min \sum_i w_i(\|x-p_i\|)\|\mathbf{g}^T(p_i)\mathbf{c}(x) - f_i\|^2 \quad (3)$$

where, $\mathbf{g}(x) = [g_1(x), g_2(x), ..., g_m(x)]^T$ is the polynomial basis vector and $\mathbf{c}(x) = [c_1(x), c_2(x), ..., c_m(x)]^T$ is the vector of unknown coefficients, which we wish to resolve to satisfy Equation 3. The number m of elements in $g(x)$ and $c(x)$ is $m = (D+r)!/(D!r!)$. $w_i(\|x-p_i\|)$ is the weighting function by distance to x, and has the following characteristics: compact support, non-negative, and monotone decreasing. Many choices for the weighting function have been proposed, such as the Gaussian

$$w_i(d) = e^{-d^2/h^2} \quad (4)$$

where h is a spacing scalar parameter, which can be used to smooth out small features in the data, and can be also called as the radius of supporting region or bandwidth.

Equation 3 can be equivalently expressed as a linear system of equations (LSE) in matrix and vector form.

$$\begin{bmatrix} w_1^{\frac{1}{2}}(\|x-p_1\|) & & \\ & \ddots & \\ & & w_n^{\frac{1}{2}}(\|x-p_n\|) \end{bmatrix} \mathbf{g}(x)\mathbf{c}(x) = \begin{bmatrix} w_1^{\frac{1}{2}}(\|x-p_1\|) & & \\ & \ddots & \\ & & w_n^{\frac{1}{2}}(\|x-p_n\|) \end{bmatrix} \begin{bmatrix} f_1 \\ \vdots \\ f_n \end{bmatrix} \quad (5)$$

Solution 1. By setting the partial derivatives of $f(x)$ to zero, i.e., $\nabla f(x) = 0$ where $\nabla = [\partial/\partial c_1, ..., \partial/\partial c_m]^T$, we obtain

$$\frac{\partial f(x)}{\partial c_j(x)} = 2\sum_i w_i(||x-p_i||)g_j(p_i)[\mathbf{g}^T(p_i)\mathbf{c}(x) - f_i] = 0 \quad (6)$$

where $j = 1, ..., m$, and we can get

$$\sum_i w_i(\|x-p_i\|)\mathbf{g}(p_i)\mathbf{g}^T(p_i)\mathbf{c}(x) = \sum_i w_i(\|x-p_i\|)\mathbf{g}(p_i)f_i \quad (7)$$

If the matrix $Mg = \sum_i w_i(||x-p_i||)\mathbf{g}(p_i)\mathbf{g}^T(p_i)$ is not singular, i.e., its determinant is not zero, $\mathbf{c}(x)$ is solved as

$$\mathbf{c}(x) = Mg^{-1}\left(\sum_i w_i(||x-p_i||)\mathbf{g}(p_i)f_i\right) \quad (8)$$

As a result, $f(x)$ can be expressed as

$$f(x) = \mathbf{g}^T(x)\, Mg^{-1}\left(\sum_i w_i(||x-p_i||)\mathbf{g}(p_i)f_i\right) \quad (9)$$

The method of Normal Equations also computes the identical solution for $\mathbf{c}(x)$ and $f(x)$ in the MLS sense.

Solution 2. The MLS function can also be given in terms of **polynomial reproduction property** [Lev98]:

$$f(x) = \sum_i f_i L_i^{P,r}(x) \quad (10)$$

where shape function $L_i^{P,r}(x)$ minimizes the quadratic form,

$$Q = \sum_i \left|L_i^{P,r}(x)\right|^2 w(\|x-p_i\|) \quad (11)$$

subject to the linear constraints of polynomial reproduction.

$$\sum_i L_i^{P,r}(x)g_j(p_i) = g_j(x),\ j = 1, \ldots, m \quad (12)$$

The minimization is computed by solving a linear system derived by Lagrange multipliers.

3. MLS Surfaces Definition

For existing MLS surface algorithms, we classify them into two categories. The first type is referred to as the *projection MLS surfaces* definition, where the MLS method is used to define a surface as the stationary points of an iterative parametric fit procedure. The second type, namely the *implicit MLS surfaces* definition, employs MLS technique to solve an implicit function under some constraints. These constraints include the positional constraint, which enforces exactly interpolation/approximation at the sample points, and even derivative constraint, which enforces the matching between the gradients of the implicit function and the normals of corresponding sample points.

3.1. Projection MLS Surfaces

Levin [Lev03] generalizes his previous work [Lev98] in function approximation theory to adapt for manifold by involving a non-linear optimization for each point projection. The MLS **stationary projection** operator f: $\Omega \rightarrow \mathbf{R}^3$ projects points from a vicinity Ω of the MLS surface onto the surface itself.

$$\mathcal{S} = \{x \in \Omega : f(x) = x\} = range(f) \quad (13)$$

Besides the intact projection procedures described in Section 3.1.1, there is another popular reduced version (Section 3.1.2), which simply defines the point projection in terms of a combination of the weighted centroid and a normal field. This later function can be evaluated very efficiently, and its simplicity makes it more suitable for analysis that gives strong theoretical guarantees.

3.1.1. Point Set Surfaces

The MLS surface of Levin [Lev98][Lev03] is first introduced to 3D computer graphics by Alexa et al. [ABCO*01], and named as **point set surfaces**. The domain Ω is the neighborhood of the input points. For the irregular sampling point set, the domain is generated from k-nearest neighboring points, while for regular samples, it can be defined by a

union of balls centered at each point p_i :

$$\Omega = \bigcup_i \{x \in \mathrm{R}^3 | \ \|x - p_i\| < r_\Omega\} \tag{14}$$

The computation of the projection $x \mapsto f(x)$, detailed in [ABCO*03], is carried out by two steps:

(1) Local reference domain $H_x = \{z \in \mathbf{R}^3 : n^T z = n^T q\}$is computed by minimizing the non-linear energy function,

$$e_{MLS}(q,n) = \sum_i w(\|p_i - q\|)(n^T p_i - n^T q)^2 \tag{15}$$

for any n,$q \in \mathbf{R}^3$,$||n||$=1,with $n = n(q) = (x-q)/||x-q||$, where w is the Gaussian weighting function (Equation 4). The scalar parameter h should be set to reflect the anticipated spacing between neighboring points, since the surfaces can be tuned to smooth out features of size $< h$ in $\mathcal{S}$.

Setting $q = x+tn$ for $t \in \mathbf{R}$, Equation 15 becomes:

$$e_{MLS}(x,n) = \sum_i w(\|p_i - x - tn\|) \langle n, p_i - x - tn \rangle^2 \tag{16}$$

By iterative non-linear minimization, Alexa et. al. find the local minimum of Equation 16 with smallest t and the local tangent plane H_x near x accordingly (Figure 1.a). The local reference domain is then given by an orthonormal coordinate system on H_x so that q is the origin of this system.

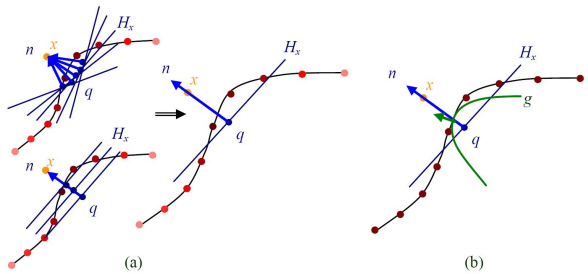

Figure 1: *The two step projection (Taken from [GP07]). (a) The reference plane for the first step of the MLS projection is found by optimizing over all normal direction n and offsets t. (b) The reference plane is used to compute a polynomial least square approximation. The value at the origin of the reference frame is used as q.*

(2) Find a local bivariate polynomial approximation g:$H_x \rightarrow \mathbf{R}^3$ by a standar least squares fit to the points p_i in the neighborhood of q. Once the plane H_x is computed, the weights $w(||p_i - q||)$ are known. Polynomials of degree 3 to 4 have proved to be appropriate, as they produce good fits of the neighborhood without oscillation and can be computed efficiently. Let q_i be the orthogonal projection of p_i onto H_x , (x_i, y_i) be its local 2D coordinates, and $g_i = n^T(q_i - p_i) = ||q_i - p_i||$ be its height over H_x. Then the error

$$\sum_i w(\|p_i - q\|)(\ g(x_i,y_i) - g_i\ )^2 \tag{17}$$

is to be minimized (Figure 1.b). And the projection of x is finally defined by

$$f(x) = q + g(0,0)n = x + (t + g(0,0))n \tag{18}$$

In particular, The normal of point x might be given or could be computed by refining a point set surface (e.g., from close points in the already processed point set). In the later case, the normal of x is obtained by a weighted averaging of input normals n_j, when point p_j of k nearest neighbors carries normal n_j.

$$n(x) = \frac{\sum_{j\in[1,k]} w(\|x - p_j\|)n_j}{\left\|\sum_{j\in[1,k]} w(\|x - p_j\|)n_j\right\|} \tag{19}$$

If no normal is available, it is computed using the Jacobian,

$$J_{MLS}(x,n) = \sum_i w(\|p_i - x\|) \langle n, p_i - x \rangle^2 \tag{20}$$

and the normal is the eigenvector (corresponding to the smallest eigenvalue λ_0) of the covariance matrix.

$$\sum_i w(\|p_i - x\|)(p_i - x)(p_i - x)^T = E diag(\lambda_0, \lambda_1, \lambda_2) I_3^T \tag{21}$$

Fitting a tangent plane will only yield a normal direction, not an orientation. The normal orientation is always estimated by the minimum spanning tree (MST) algorithm [HDD*92].

Based on the polynomial reproduction property [Lev98] of the MLS approach and local Taylor expansion of f, Lipman et al. [LCOL06] evaluate **the approximation quality of the MLS surfaces**.

$$\left|\partial^r f(x) - \partial^r f'(x)\right| \leq C B_{r,\varepsilon}(x, \mathcal{P}_h) \tag{22}$$

where, C is the local upper bound to the $(r+1)$th order derivatives $|\partial^{r+1} f|$. Π_h denotes the subset $\Pi \cap BALL_h(x)$, $BALL_h(x)$ stands for a ball of radius h with center x. $B_{r,\varepsilon}(x,\mathcal{P}_h)$ is the error bounding function defined to the given points Π_h with ε noise. For $B_{r,\varepsilon}(x,\mathcal{P}_h)$, they develop two error bounds: tight data-independent and data-dependent bound. To minimize the two bounds, they look for two different optimal support h with the weight function $w(d)$.

$$w(d) = e^{-d^2/(h-d)^2}, \ d \in [0,h) \tag{23}$$

Their solution method can be well integrated with the MLS projection operator as follow: After find the local reference plane, the second step of local bivariate polynomial is performed by using the computed optimal h.

By using the weight (Equation 23), Wang et al. [WSS08] also discuss the implications of Levin's second-step MLS projection for **optimal bandwidth**, i.e., the Gaussian width h. They use kernel regression weights to determine the optimal bandwidth, but actually use regular MLS weighting function to perform the second polynomial fitting step. By plugging the circular-form bandwidth (expressed as a matrix) into an approximated error criteria evaluating the kernel regression performance, the optimal bandwidth can be found. The fitting results for typical functional shape (similar to [LCOL06]) and example point clouds show that the optimal bandwidth outperforms the heuristic bandwidth where h is a constant or determined by the k nearest neighbors.

3.1.2. Derived From Normal and Weighted Average

One slightly different but considerably **simple projection** is proposed by Adamson and Alexa [AA03a]. Their projection procedure iteratively projects a given point x onto local reference planes that are defined by a weighted average position

$$a(x) = \frac{\sum_i w(\|x - p_i\|) p_i}{\sum_i w(\|x - p_i\|)} \tag{24}$$

The local normal approximation $n(x)$ is computed (this approximation allows drawing a connection to Levin's MLS surfaces) similar to [ABCO*01][ABCO*03]: If each point p_i carries a normal n_i, Equation 19 is used, otherwise, Equation 21 is employed. And then, Repeat the computation (see Figure 2.a for pseudocode) until convergence of the projection operator.

$$x' = x - n(x)^T (x - a(x)) n(x) \tag{25}$$

Convergence means that a point is stationary under the repeated projection onto the subspace orthogonal to $n(x)$. This constructive definition also gives rise to an alternative interpretation of the surface defined by an implicit function. The implicit function $f : \mathbf{R}^3 \rightarrow \mathbf{R}$ describes the distance of a point x to the weighted average $a(x)$ projected along the normal direction $n(x)$:

$$f(x) = n(x)^T (x - a(x)) \tag{26}$$

In the following work, Alexa and Adamson [AA04b] point out that "since the normal $n(x)$ of the approximating tangent plane is generally not the normal to Σ, this basic projection procedure does not result in an orthogonal projection". The implicit description allows the exact evaluation of surface normals using the gradient ∇f. Based on the property of projection operator, the projection can be further classified into **three type procedures**: basic projection (Figure 2.a), almost orthogonal projection (Figure 2.b), and orthogonal projection (Figure 2.c). Figure 2.d shows the different results of the three projections for one point. Obviously, the sophisticated orthogonal projection produces the best result and the basic projection has the simplicity with less accuracy. The almost orthogonal projection trades off the simplicity and accuracy and it has been widely adopted as the implementation standard.

In contrast to the algorithmic construction of MLS surfaces using the projection operator, Amenta and Kil [AA04a] give an **explicit definition of MLS surfaces** in terms of critical points of the energy function e along lines determined by a vector field $n(x)$. The energy function takes a position x and an un-oriented direction b.

$$e_{MLS}(x,b) = \sum_i w(\|p_i - x\|)(\langle b, p_i \rangle - \langle b, x \rangle)^2 \tag{27}$$

The vector function takes a position x.

$$n_{MLS}(x) = \text{argmin}_b \, e_{MLS}(x,b) \;, \|n\| = 1 \tag{28}$$

(a) Basic Orthogonal Projection(x,σ)
1) x= weightedAverage(x);
2) do
3) n=localNormal(x);
4) a=weightedAverage(x);
5) f=n^T(x–a);
6) x=x–fn;
7) while (|f|>σ);
8) return x;

(b) Almost Orthogonal Projection(x,σ)
1) x_0=x;
2) do
3) n=localNormal(x);
4) a=weightedAverage(x);
5) f=n^T(x_0–a);
6) x=x_0–fn;
7) while (|f|>σ);
8) return x;

(c) Orthogonal Projection(x,σ)
1) x_0=x;
2) do
3) n=localNormal(x);
4) a=weightedAverage(x);
5) explicitly compute gradient $\nabla f(x)$;
6) $g = \frac{n^T(x_0 - a)}{n^T \nabla f}$
7) x=x_0–$g\nabla f$;
8) while (|f|>σ);
9) return x;

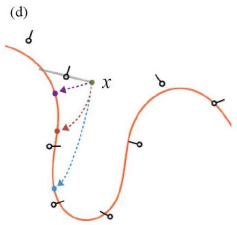

Figure 2: *Three projection procedures. (a) Pseudocode for the basic projection operator. (b) Pseudocode for the almost orthogonal projection operator. (c) Pseudocode for the orthogonal projection operator. (d) Comparisons for some point extremities (Taken from [GGG08]). Blue: basic. Red: almost orthogonal. Magenta: orthogonal.*

The MLS surfaces (Figure 3 and 4) is defined as

$$\mathcal{S} = \{x \,|\, x \in \text{arglocalmin}_{y \in l_{x,n(x)}} e_{MLS}(y, n(x))\} \tag{29}$$

where $l_{x,n(x)}$ is the line through $x$with direction $n(x)$, $\text{arglocalmin}_{y \in l_{x,n(x)}}$ denotes the set of inputs y producing local minima of a function of variable y. The process of projecting a point onto the extremal surface Σ implied by n and e is illustrated in Figure 3. At each iteration, find $n(x_j)$ and consider the line $l_{x_j,n(x_j)}$, then search for a nearby local minimum of $e(y,n(x_j))$ over the set $y \in l_{x_j,n(x_j)}$. The nearest local minimum becomes x_{j+1}, until $n(x_n)$ at final point does not increase e. The energy does indeed decrease for the MLS function at every step and also for any function $e(x,a)$ which does not depend on the direction parameter. And, once the above procedure converges, it produces a point of Σ.

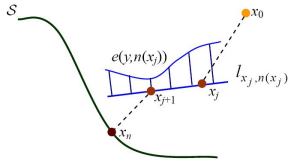

Figure 3: *Illustration of the definition and the projection process of a projection MLS.*

Since a point of the extremal surface is a critical point of x on the line $l_{x,n(x)}$, the directional derivative of $e(y,n(x))$ in the direction $n(x)$ has to be zero at x. In other words, at a

point x of the extremal surface, n(x) is perpendicular to the gradient of $e(y,n(x))$, and tangent to the iso-surface $e(y,n(x))$ for y in $\mathbf{R}^3$. This is illustrated in Figure 4. Based on the form of implicit definition (Equation 26), Amenta and Kil define more general formulation of extremal surfaces,

$$f(x) = n(x)^T \left(\frac{\partial e_{MLS}(y,n(x))}{\partial y}|_x\right) = 0 \qquad (30)$$

where $\frac{\partial e_{MLS}(y,n(x))}{\partial y}|_x$ is the gradient of e as a function of y, keeping $n(x)$ fixed, and then evaluated at x. When the weighting function is a Gaussian (Equation 4), the implicit function f takes the following form:

$$f(x)=2\sum_i n(x)^T(x-p_i)\left[1-\left(\frac{n(x)^T(x-p_i)}{h}\right)^2\right]w_i(\|x-p_i\|) \qquad (31)$$

Notice that a point x on one of the iso-surfaces might be either a maximum on $l_{x,n(x)}$ or a minimum (Figure 4).

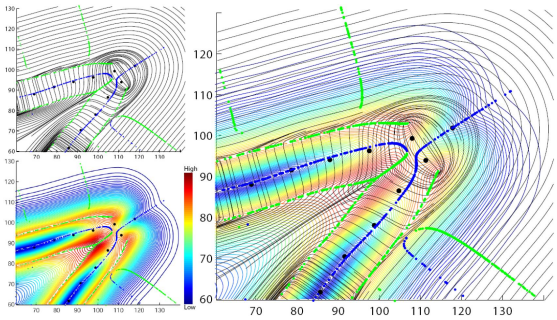

Figure 4: *The reconstruction instance near a feature corner (Taken from [AK04b]). (a) Top left, the stream lines of the vector field generated by n. (b) Bottom left, the iso-contours of the energy function e. (c) Right, both together. The scale factor on the Gaussian weights in the MLS energy function h = 10. The extremal surface $f(x) = n^T \nabla e = 0$ (Equation 30) includes both minima (heavy blue curve, Equation 29) and maxima (green curve). The endpoints and junctions in the union of the blue and green curves are singularities of n.*

The energy function can also be reduced as

$$e_{MLS}(y,n(x)) = \sum_i w(\|p_i - x\|)\left(n(x)^T(y-p_i)\right)^2 \qquad (32)$$

where the weighting function w varies with x instead of y. This is essentially the MLS surface implemented in Pauly et al.'s methods [PGK02][PKKG03] and the popular PointShop3D platform [ZPK*02], and it has a very **straightforward implicit form**.

$$f(x) = \sum_i w(\|x-p_i\|)\left(n_i^T(x-p_i)\right)^2 \qquad (33)$$

Dey et al. [DGS05] prove **theoretical guarantees for the projection MLS surfaces** [AK04a]. The extremal surface based projection procedure converges and the target extremal surface is isotopic to the original sampled surface. The projection procedure indeed converges to extremal surface when the point set $\mathcal{P}$ is sufficiently dense and the initial point x is chosen sufficiently close to $\mathcal{S}$. The sampling condition implies two facts: uniform (ε,α) samples and that two nearby sample points have similar normals.

3.2. Implicit MLS Surfaces

Implicit MLS (IMLS) surfaces is a zero level-set surface of a signed distance field based on the MLS methods. General reconstruction via implicit surfaces includes the following steps: estimate normals, approximate signed distance function, apply the mesh creation algorithm (e.g., marching cube [LE87]), build final mesh. And if necessary, an optimization to the resultant mesh will be performed.

3.2.1. IMLS Surfaces Based on Euclidean Distance

To provide the **interpolation and approximation simultaneously**, Shen et al. [SO'S04] use the weight function

$$w(d) = \left(\frac{1}{d^2+\varepsilon^2}\right)^2 \qquad (34)$$

Setting the parameter ε to zero results in a singularity at d=0, which enforces the MLS fitting function to interpolate the data. Shen et al. solve Equation 5 with the positional constraints f_i and normal constrains $n_i^T(x-p_i)$ at the same time.

$$\begin{bmatrix} w_1^{\frac{1}{2}}(\|x-p_1\|) & & \\ & \ddots & \\ & & w_n^{\frac{1}{2}}(\|x-p_n\|) \end{bmatrix}\begin{bmatrix} g^T(p_1) \\ \vdots \\ g^T(p_n) \end{bmatrix}\mathbf{c}(x)=\begin{bmatrix} w_1^{\frac{1}{2}}(\|x-p_1\|) & & \\ & \ddots & \\ & & w_n^{\frac{1}{2}}(\|x-p_n\|) \end{bmatrix}\begin{bmatrix} \tilde{f}_1 \\ \vdots \\ \tilde{f}_n \end{bmatrix} \qquad (35)$$

where $\tilde{f}_i(x) = f_i + n_i^T(x-p_i)$, it's a signed distance function and can be used to express iso-surfaces. For simplicity, choose g=[1], i.e., f(x) is a level set value (f(x)=c(x)). Notice that, since the weights vary over x, $f(x)$ becomes a function of x instead of a constant. Then, Equation 35 becomes

$$\left(\sum_i w^{\frac{1}{2}}(\|x-p_i\|)\right) f(x) = \sum_i w^{\frac{1}{2}}(\|x-p_i\|)\tilde{f}_i(x) \qquad (36)$$

In the case, the fit form equation can be intuitively interpreted as the value of the interpolating function at x, which is simply the weighted average of the values at x predicted by each of the $\tilde{f}_i$. Since the implicit surface definition is designed to reconstruct mesh from polygon soup, positional constrains should be changed to polygonal constrains. The parenthesized term of Equation 36 and the term on the right are replaced by integrals over the polygons.

$$f(x) = \frac{\sum_i \int_{Poly_i} w(\|x-p\|)(f_i + n_i^T(x-p_i))dp}{\sum_i \int_{Poly_i} w(\|x-p\|)dp} \qquad (37)$$

The quadrature are approximated numerically using an importance sampling approach. Note that,when extracting final meshes, they use the polygonizing algorithms described in [Blo94]. Besides the mesh reconstruction, the MLS definition has been applied to level-of-detail, bounding volume creation, and deformable object simulation.

Shen et al.'s approach [SO'B04] is close to the previous implicit algorithm [OBA*03], which uses the **partition-of-unity**(POU) method with a fast hierarchical evaluation scheme to compute surfaces from data sets. POU performs local least squares approximation at fixed position and blend them using weighted average while MLS performs a weighted local least squares approximation for every point. In practice it turns out that when using a constant basis, the two approaches are equivalent (assuming the POU is performed for each point using the same weighting scheme).

In the simplest case, by taking f_i=0, $\tilde{f}_i = n_i^T(x-p_i)$, Kolluri [Kol05] defines **uniform implicit MLS surfaces** f as a weighted average of the tangent planes.

$$f(x) = \frac{\sum_i w(\|x-p_i\|)\left(n_i^T(x-p_i)\right)^2}{\sum_i w(\|x-p_i\|)} \tag{38}$$

with the weighting function

$$w_i(d) = e^{-d^2/(\rho\cdot slfs)^2}/ns_i \tag{39}$$

where, ns_i is the number of samples inside a ball of radius r_i centered at p_i, including p_i itself. The local feature size (*lfs*) at point $x \in \mathcal{S}$ is defined as the distance from x to the nearest point of the medial axis of $\mathcal{S}$. The width of the Gaussian functions is ρ (ρ<0.01) times of the smallest local feature size (*slfs*), normalized to unit 1 in the algorithm. At the uniform (ε,α) sampling condition (Figure 5.a), which is a shell with uniform thickness 2ε·*slfs*, Kolluri proves that the implicit function is a good approximation of the signed distance function of the original surface, and that the reconstructed surface is geometrically and topologically correct.

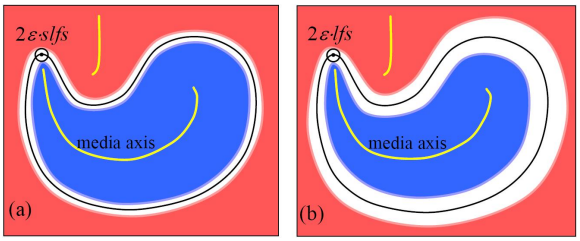

Figure 5: *The (ε,α) sampling condition. Implicit MLS surfaces (black) located in the sampling space (white regions) is defined based on the local feature size relating to the media axes (yellow). (a) Uniform (ε,α) sampling space. (b) Adaptive (ε,α) sampling space.*

In a larger adaptive sampling space (Figure 5.b), Dey and Sun [DS05] can reconstruct provably good **adaptive MLS surfaces**, at the aid of one modified Gaussian function to be a fraction of the local feature size.

$$w_i(\|x-p_i\|) = e^{-\sqrt{2}\|x-p_i\|^2/(\rho^2 lfs(\hat{x})lfs(\hat{p}_i))} \tag{40}$$

where ρ is a user-specified parameter that is less 1. For sample x, $\hat{x}$ is the closest point on $\mathcal{S}$. Since the local feature size should be pre-computed, Dey et al. use another surface reconstruction procedure via adaptive MLS function. First, they estimate each point's normal by Delaunay ball. Then, they compute local feature size of each sample. After that, they project each sample point by iteratively using $x' = x - f(x)\nabla f(x)/\|\nabla f(x)\|^2$. Finally, they build final meshes based on the Cocone algorithm [ACDL02].

3.2.2. IMLS Surfaces Derived From Algebraic Distance

By directly fitting (withoud iterative solving procedure) higher order algebraic spheres [Pra87] rather than planes, Guennebaud and Gross [GG07] propose an algebraic MLS surface, called **algebraic point set surfaces** (APSS). An algebraic sphere is the 0-isosurface of the scalar field $s_{\mathbf{u}}(x) = [1, x^T, x^T x]\mathbf{u}$, where $\mathbf{u} = [u_0, \ldots, u_{D+1}]^T \in \mathbf{R}^{D+2}$ (D is the dimension, here D=3) is the vector of scalar coefficients describing the sphere. The APSS $\mathcal{S}_P$ is defined as the zero set of the implicit scalar field $f(x)$ representing the algebraic distance between the point x and the fitted sphere $u(x)$.

$$f(x) = s_{u(x)}(x) = [1, x^T, x^T x]u(x) = 0 \tag{41}$$

The weighting function used in the paper is a compact supporting polynomial

$$w_i(x) = \begin{cases} \left(1 - \left(\frac{\|x-p_i\|}{h_i(x)}\right)^2\right)^4, & \|x-p_i\|<1 \\ 0, & \text{otherwise} \end{cases} \tag{42}$$

where $h_i(x)$ describes the local feature size.

The solution $u(x)$ of the algebraic sphere fit at position x, only used to estimate the missing normals, is expressed as

$$u(x) = \arg\min_{u,u\neq 0}\left(u^T \mathbf{W}P(x)u\right) \tag{43}$$

where, $\mathbf{W}P(x)$ is

$$\mathbf{W}P(x) = \begin{bmatrix} 1 & p_1^T & p_1^T p_1 \\ \vdots & \vdots & \vdots \\ 1 & p_n^T & p_n^T p_n \end{bmatrix}^T \begin{bmatrix} w_1(\|x-p_1\|) & & \\ & \ddots & \\ & & w_n(\|x-p_n\|) \end{bmatrix} \begin{bmatrix} 1 & p_1^T & p_1^T p_1 \\ \vdots & \vdots & \vdots \\ 1 & p_n^T & p_n^T p_n \end{bmatrix} \tag{44}$$

By using of the normals obtained by the consistent orientation propagation method or given as part of the input, Guennebaud and Gross add the following derivative constraints to the minimization problem (Equation 43).

$$n_i = \nabla s_u(p_i),\ \|n_i\| = 1 \tag{45}$$

The normal constraint leads to a standard linear system of equation that can be solved efficiently.

In the following work, Guennebaud et al. [GGG08] show that the former algorithm actually minimizes Equation 41 under the positional constraints $s_u(p_i)$=0 and the derivative constraints (Equation 45) simultaneously:

$$u(x) = \arg\min_u \sum_i w_i(x)\left(s_u(p_i)^2 + \|\nabla s_u(p_i) - n_i\|^2\right) \tag{46}$$

And they observe that the derivative constraints provide sufficient information to determine the four coefficients u_1 to u_4 that define the gradient of the sphere, by explicitly solving the below normal equation.

$$\begin{bmatrix} \sum_i w_i(x) I_3 & 2\sum_i w_i(x) p_i \\ (2\sum_i w_i(x) p_i)^T & 4\sum_i w_i(x) p_i^T p_i \end{bmatrix} \cdot \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} \sum_i w_i(x) n_i \\ 2\sum_i w_i(x) p_i^T n_i \end{bmatrix} \quad (47)$$

Next, they use the algebraic distance constraints to determine the value of u_0. By multiplying u_4 by an introduced parameter β, Guennebaud et al. [GGG08] can continuously tweak their algebraic spherical fit between a pure planar fit ($\beta = 0$) and a pure spherical fit ($\beta = 1$). Moreover, the parameter can control the curvature of the fitted sphere (inverting the curvature with a negative value), or enhance the surface microstructures.

3.3. Summary

For MLS surfaces, the surface definition is the most important issue, since it determine which form of projection procedures is used or which formulation of implict functions is taken. Although the two definitions are formulated from different point of view and put their emphasis on their own aspects, it dose not mean they can't be converted between each other. Generally, the projection MLS surfaces may be expressed by the implicit functions [AA03a][AA04b][AK04a], while the pure implicit MLS surfaces can also include projection procedure [GG07]. In particular, it can be found that Equation 33 (one projection MLS surface) and 38 (one implicit MLS surface) define scalar fields by similar signed distance of local tangent planes, and have exactly the same zero level set. Equation 38 has a significant computational advantage, since Newton iteration for Equation 38 has a much larger convergent domain than the one for Equation 33. However, Equation 33 facilitates theory analysis.

4. Properties and Conditions

Amenta and Kil [AK04b] find two characteristics (Figure 4) of the extremal surface defined by the energy function [AK04a]. (1) The domain of MLS surfaces (i.e., the converging neighborhood of surface $\mathcal{S}$) is a narrow region around the ideal $\mathcal{S}$. This means that, when the noise level exceeds expectations, it may not even include the entire point cloud by causing some of the points in the cloud to project away from where the surface ought to be. (2)When the input points come from surfaces with sharp corners, in the vicinity of which two sheets of the MLS surfaces seem to collapse into one. The MLS surfaces extend through the corners with discontinuity, and some unexpected surfaces exist not near the input points. The two un-satisfying facts can be generalized into two types of constrains: (1) sampling space properties (Section 4.1) including spatial scope, sampling rule and sampling density, (2) desired outputs (Section 4.2), e.g., smooth surfaces with sharp features, and complex manifolds.

4.1. Sampling Space

4.1.1. MLS Surfaces Domain

To solve the previous two undesirable phenomena, Amenta and Kil [AK04b] address two MLS variants which can be defined on two wider domains without producing spurious surface components. Similar to the simple MLS surfaces method [AA03a], they first define a **weighted center-of-mass (COM) energy** function by evaluating position x.

$$e_{COM}(c,b) = \sum_i w(\|p_i - x\|)(\langle b, p_i\rangle - \langle b, c\rangle)^2 \quad (48)$$

where $c = \sum_i p_i \bar{w}(\|x - p_i\|)$ is calculated by the normalized Gaussian weight of the sample p_i $\bar{w}(\|x-p_i\|) = w(\|x-p_i\|)/\sum_j w(\|x-p_j\|)$. The COM method is efficient and works well except near sharp corners. So another **line integral** approach, computationally expensive, is defined to work well everywhere.

$$e_I(x,a) = \int_{y \in \ell_{x,a}} e^2_{DIST}(y) w(\|y-x\|) \quad (49)$$

where $e_{DIST}(y) = \sum_i \bar{w}(\|y-p_i\|)\,\|y-p_i\|^2$ estimates the weighted square distance of x from the input point set.

In Section 3.2.1, domains that allow provable MLS surface construction with uniform sampling [DGS05][Kol05] and adaptive sampling [DS05] have been discussed.

Besides these, Bremer and Hart [BH05] analyze the mild sampling conditions of MLS surfaces domain by defining normals within an adaptive tubular neighborhood [AA04a] of the sampling points. They impose the upper and lower bounds on sampling density by requiring an (u,l)-sampling of the surface, which simultaneously satisfies two conditions: (1)the collection of balls centered at p_i of radius $u \cdot lfs(\hat{p}_i)$ covers $\mathcal{S}$, (2) balls center at p_i of radius $l \cdot lfs(\hat{p}_i)$ do not intersect. In the (u,l)-sampling domain, they prove the constructed MLS surface with an adaptive Gaussian weighting function can be well defined.

$$w_i(\|x-p_i\|) = e^{-\frac{(25\|x-p_i\|)^2}{lfs^2(\hat{p}_i)}} \quad (50)$$

The well defined normal of position x is the vector corresponding to only one smallest eigenvalue of the weighted covariance matrix (Equation 21).

4.1.2. Adaptive Sampling

Sampling condition should guarantee that a surface is appropriately sampled, which at least intuitively means that the local sampling density should depend on the local geometric complexity, e.g., some indication of curvature.

For the projection operator defined by Alexa et al. [ABCO*01], using a **global constant scale factor** h in Gaussian function can be problematic for non-uniformly

sampled surfaces. If h is too large, areas of high sampling density will be smoothed excessively. On the contrary, numerical instabilities occur in sparsely sampled regions, if h is too small.

To deal with this problem, Pauly et al. [PGK02] use an **extension of the static MLS approach**. Instead of considering samples within a fixed radius proportional to h, they collect the k-nearest neighbors and adapt according to the radius r of the enclosing sphere. The weighting function is defined as

$$w_i(\|x-p_i\|) = e^{-\|x-p_i\|^2/h_r^2} \tag{51}$$

By dynamically choosing $h_r = r/3$, they ensure that only points within the k-nearest neighbors contribute noticeably to the least squares optimization of Equation 15.

Subsequently, Pauly et al. [PKKG03] propose an **anisotropic weighting MLS method** by redefining a new weighting function, where $h_r = h/\rho(r)$ and $\rho(r)$: $\mathbf{R}^3 \rightarrow \mathbf{R}$ is a continuous and smooth function approximating the local sampling density. To compute ρ, they first estimate the local sampling density ρ_i for each p_i by finding the sphere with minimum radius r_i centered at p_i that contains the k-nearest neighbors of p_i. Then ρ_i is defined as $\rho_i = k/r_i^2$ where their experiments showed that k should be greater than 6 to ensure stable computations, but less than 20 to avoid excessive smoothing of the density estimation. In the second step, ρ can be interpolated using standard scattered data approximation techniques, e.g., radial basis functions.

Adamson and Alexa [AA06a] propose an **anistropic point set surfaces**, attaching individual weight function to each sample rather than to the location in space. The weight $w_i(||x–p_i||)$ is performed by transforming the vector $x–p_i$ with ellipsoidal weight H_i derived from each sample.

$$w_i(\|x-p_i\|) = e^{-\|x-p_i\|^2/\|H_i\|^2} \tag{52}$$

The ellipsoid H_i has a good orientation that one of its axis points into normal direction, and the other two align with the local minimum and maximum principal curvature direction. H_i is determined by a covariance matrix $cov(p_i)$ constructed by the k-nearest neighbors $p_{i(1)}, \ldots, p_{i(k)}$ of point p_i.

$$cov(p_i) = \frac{1}{k}\sum_{j\in[1,k]} (p_{i(j)} - p_i)(p_{i(j)} - p_i)^T \tag{53}$$

The covariance matrix can be further diagonalized as $cov(p_i) = R_i \Lambda_i R_i^T$, then H_i is defined as $H_i = \left(\frac{2+\sqrt{k}}{3\sqrt{k}}\right)^{-1/2} R_i \Lambda_i^{1/2}$. Then, $w_i(||x-p_i||)$ is used to compute the weighted average location and the weighted principal normal direction of the simple MLS surfaces [AA03a].

By using **Mahalanobis distance** which is a distance measure with ellipsoidal support, Fiorin et al. [FCS07] define the following distance function, used for surfels by [AK04a].

$$dist_M(x,p_i) = \left(n_i^T(x-p_i)\right)^2 + \beta \left\| (x-p_i) - \left(n_i^T(x-p_i)\right) n_i \right\|^2 \tag{54}$$

where n_i is the vector direction at point p_i, β is a scalar factor which affects the ellipsoid shape. In particular, when β=1, the Mahalanoibis distance is quivalent to the Euclidian distance between the point x and the samples p_i, whereas when β=0, it corresponds to the distance from x to the plane through p_i with normal n_i. The enegy function is changed to

$$e_{MLS}(x,b) = \sum_i \bar{w}(\|p_i - x\|) dist_M(x,p_i) \tag{55}$$

where $\bar{w}(||x-p_i||)$ is the normalized Gaussian function.

4.2. Outputs

The desired output is another important factor that should be carefully considered to develop reasonable algorithm. The following two kinds of outputs have been thoroughly investigated: complex surfaces (Section 4.2.1) and models with sharp features (Section 4.2.2).

4.2.1. Complex MLS Surfaces

By computing the weighted average and the local tangent frame as [AA03a], Alexa and Adamson [AA04a] define the **MLS surfaces with boundaries** in a natural way. For points far away from the point set, the distance $c(x) = ||x - a(x)||$ (called as off-center value of x) increases. The main idea for defining a boundary is to require $c(x)$ to be less than a user-specified threshold δ_c. More precisely, the surface $\mathcal{S}_P$ is implicitly defined as

$$f(x) = n(x)^T (x - a(x)) = 0 \quad \wedge \quad c(x) < \delta_c \tag{56}$$

δ_c and $c(x)$ are both dependent on sampling conditions. Alexa and Adamson [AA03a][AA04a] assume that each point of $\mathcal{P}$ is near $\mathcal{S}_P$, and the union of the balls B_{p_i}, centered at p_i with radius r_B, contain the surface $\mathcal{S}_P$. Determined by experiments, $r_B \approx 1.5 D_{knn}$, where D_{knn} is computed as the average Euclidean 6-nearest neighbor distance. Obviously, $c(x)$ behaves roughly like the distance to the point set, and δ_c is relative to r_B, $\frac{2}{3} r_B < \delta_c < \frac{1+4\sqrt{3}}{9} r_B$.

Beside the bounded surfaces, handling **non-orientable surfaces** [AA04a] is straightforward, as the surface definition (Equation 56) is not altered when inverting $n(x)$. With sufficient and small radii, the normals can be oriented consistently so that $n(x)$ is indeed a smooth vector-valued function of x inside the ball.

Alexa and Adamson [AA08] find that old definition (Equation 26) has certain deficiencies in shape (e.g., given input points in convex position, the resulting shape is not convex). By using the singular weight function

$$w(d) = d^{-k},\ k \geq 2 \tag{57}$$

, they define **Hermite point set surfaces** with an extended weighted average:

$$\tilde{a}_i(x) = \frac{\sum_i w(||x-p_i||)\tilde{p}_i(x)}{\sum_i w(||x-p_i||)} \tag{58}$$

where $\tilde{p}_i(x) = x - (n_i^T(x-p_i))n_i$ is the projection of x onto the tangent space of p_i. The surface is then defined as

$$f(x) = n(x)^T(x - \tilde{a}(x)) = 0 \tag{59}$$

and can be equivalently expressed as the locally weighted combination of implicit plane equations at points.

$$f(x) = \sum_i \left(n_i^T(x-p_i) \left(\sum_{j\in[1,n]} n_i^T n_j w(\|x-p_i\|) w(\|x-p_j\|) \right) \right) \tag{60}$$

Therefore, for the given input samples in convex position, the resulting shape is convex, and can also be flexibly controlled by a shape parameter between $\tilde{p}_i(x)$ and p_i.

This Hermite MLS work has been initiated in the context of **Point-Sampled Cell Complexes** [AA06b]. The cell complexes consist of the manifold cells M_i^D, where D={0,1,2} denotes the dimension of the cell with index i. Under uniform sampling condition, for point x in the support of cell M, Adamson and Alexa approximate a tangent space T_xM with D dimension, where T is represented by matrix $\mathbf{T} = (\mathbf{t}_0, \ldots, \mathbf{t}_{D-1})$. The tangent frame at x is obtained by computing the larger eigenvalues and corresponding eigenvectors of a matrix of weighted covariances, with the common Wendland's radial function $w(d)$.

$$w(d) = \begin{cases} (1-\frac{d}{h})^4(\frac{4d}{h}+1), & 0 \le d \le h \\ 0, & d > h \end{cases} \tag{61}$$

The surface comprised by the cell complex is defined by the projection sequence $Q_M^i(x)$(let $Q_M^0(x) = x$): project the vector from the current weighted average $a_M(x)$ to x onto the basis of the current tangent frame space in each step, until $||Q_M^{i+1}(x) - Q_M^i(x)||$ is less that a given threshold. The computation of $a_M(x)$ should consider the $q_j^{D-1}(x)$, the results of projecting onto any connected lower-dimensional cell.

$$a_M(x) = \frac{\sum_i w(||x-p_i||)p_i + \sum_j \omega(||x-q_j^{D-1}(x)||)q_j^{D-1}(x)}{\sum_i w(||x-p_i||) + \sum_j \omega(||x-q_j^{D-1}(x)||)} \tag{62}$$

with

$$\omega(d) = \begin{cases} \left(\ln\frac{d}{h}\right)^2 - \left(\frac{d}{h}\right)^2 + 2\left(\frac{d}{h}\right) - 1, & 0 \le d \le h \\ 0, & d > h \end{cases} \tag{63}$$

Through the projection onto its boundary M_i^{D-1}, the projection onto cell M is restricted to lie within a half space through $q_j^{D-1}(x)$. The normal of the binary space is computed as

$$b_j = T_{q_j^{D-1}(x)} M_i^D - T_{q_j^{D-1}(x)} M_i^{D-1} \tag{64}$$

The result of the stationary projection is finally defined as

$$Q_M(x) = \begin{cases} Q_M^\infty(x), & b_j^T(Q_M^\infty(x) - q_j^{D-1}(x)) \ge 0 \\ q_j^{D-1}, & b_j^T(Q_M^\infty(x) - q_j^{D-1}(x)) < 0 \end{cases} \tag{65}$$

4.2.2. Sharp Features

In order to detect and reconstruct sharp creases and corners in a possibly noisy point cloud, Fleishman et al. [FCOS05] propose the **robust MLS** (RMLS) surfaces, which handle samples in a more natural way by fitting a piecewise smooth surface (Equation 20) to a sample set of piecewise smooth object rather than fitting a smooth surface to whole data. In their work, locally fitting multiple surfaces to points in the area of discontinuity is regarded as a statistical problem of fitting an estimator to data with outliers, where outliers refer to the points across the discontinuities and the statistical problem is solved by forward-search paradigm.

To overcome jittering artifacts of [FCOS05], Daniels et al. [DHOS07] leverage the robust statistical method of RMLS to project points to all possible features such that **smooth and complete curves are extracted to approximate the features well**. The process of feature extraction is carried out in five steps. Firstly, they identify potential feature points by tolerance tuning, which is measured by an automatic threshold rather than a manual value in RMLS. Secondly, they use RMLS procedure to fit multiple surfaces to the neighborhood of identified points and project each point to its nearest intersection between the surfaces. Thirdly, they remove the projected point noise using principal component analysis (PCA) of adaptively grown neighborhood. Fourthly, they create a set of lines by feature polyline propagation method, after reversely order points by distance to nearest corner point. Finally, they complete feature curves by analyzing the end points of feature polylines.

To deal with singularity problem [FCOS05] in sparse or noisy point clouds, Lipman et al. [LCOL07] realize **data-dependent MLS surfaces** in a spline approximation space. The optional singularity for each local approximation space is modeled via a singularity indicator field (SIF). For each point x, a preprocessing is performed to find a local reference plane and calculate its SIF value. The reference plane is defined using a local weighted PCA with weight

$$w(d) = e^{-\frac{d^2}{(h/4)^2}} \tag{66}$$

Based on the error expression of the MLS approximation [LCOL06], the singularity indicator Λ_i at point p_i is defined as the lower bound of the derivative

$$\Lambda_i = \frac{|f_i - f(p_i)|}{\sum_{|v|=r+1} \sum_{j\in J_h(x)} \frac{|x_j - x_i|^v}{v!} |L_j^{\mathcal{P},r}(p_i)|} \tag{67}$$

where $J_h(x) = \{j | p_j \in \cap B(x,h)\}$, B($x$,$h$) is the ball with radius h centered at x. The shape functions of MLS $L_i^{\mathcal{P},r}(x)$ are obtained via Equation 12. f_i is known and $f(p_i)$ is obtained from standard local polynomial approximation in [ABCO*01]. Then, they create the local approximation spline space S_r of total degree $\le r$ (mostly r=3) by applying the MLS framework on the SIF data.

$$f(x) = \arg\min_{f\in S_r} \sum_i w_i(||x-p_i||)||f(p_i) - f_i||^2 \tag{68}$$

Reuter et al. [RJT*05] present an alternative approach that offers the possibility to represent sharp features, by manually tagging sharp features in the point cloud as an input to

their algorithm. The key idea is to switch the second step of projection operator [ABCO*01][Lev03] to a similar one (an **enriched Reproducing Kernel Particle Approximation** (RKPA)) with the Gaussian weight (Equation 4).

By incorporateing the idea of the bilateral denoising method, Miao et al. [MHL*05] propose a **modified MLS approach** [Lev03] with the weighting function

$$w(\|x-p_i\|) = e^{-\frac{\|x-p_i\|^2}{h^2}} \cdot e^{-\frac{\|x-\varphi_{p_i}(x)\|^2}{(h')^2}} \tag{69}$$

where h' is another scale constant parameter. $\varphi_{p_i}(x)$ is the linear perdiction for x given the information at point p_i, and is defined as its projection on the tangent plane of p_i

$$\varphi_{p_i}(x) = x + (n_i^T(p_i - x))n_i \tag{70}$$

Experiments show that the modified MLS smoother can also preserve sharp edges in a degree.

Ochotta et al. [OSS*07] observe that the failures at corners and edges are unrelated to sampling conditions and they are in fact intrinsic to the geometry of the surface from which the points are sampled. Instead of using two steps [Lev03], they define MLS surfaces by directly fitting a approximating function in a **unified minimization scheme**.

$$\arg\min_{q,f} \sum_i |C_{H,f}(x) - p_i|^2 w(|C_{H,f}(x) - p_i|) \tag{71}$$

where $C_{H,f}(x)$ is the closest point projection of x, whose projection onto H is q, onto the function f in the function space $\mathcal{F}$ over reference plane H. The encapsulation of H into the closest point projection function C effectively removes any major dependence on the local reference frame. H can now simply be thought of as a means of parameterizing all possible functions in $\mathcal{F}$, over all orientations and translations. Especially, when $\mathcal{F}$ includes functions with sharp features, sharp features would be accurately reconstructed.

Guennebaud and Gross [GG07] propose another approach to **handle sharp features in the context of APSS**. Before the actual projection of a point onto the surface, they first group the selected neighbors by tagging values. Then, the APSS is executed for each group and the actual point is projected onto each algebraic sphere. Finally, they detect for each pair of groups whether the sharp crease form originates from a Boolean intersection or from a union and apply corresponding constructive solid geometry (CSG) rule.

5. Applications

5.1. Other Reconstructions

For dense samples without normals, Scheidegger et al. [SFS05] directly construct a high-quality triangulation with bounded error, by using MLS surfaces [ABCO*01][Lev03] as the underlying representation. Extending the work of Scheidegger et al. [SFS05] mainly by a more principled way to compute the guidance field, Schreiner et al. [SSFS06] propose a direct (re)meshing algorithm with higher geometric fidelity, lower triangle count and better triangle shape.

For the scans even with sparse, noisy or large missing parts, Sharf et al. [SLS*06] present a deformable model to reconstruct a watertight surface. The deformable model recovers the target shape with multiple competing evolving fronts. Once the deformable model has completed its evolution, a simplified MLS projection [ABCO*01] is used to reconstruct the final shape, where the reference plane is simply taken as the tangent to the target mesh and the order of the MLS interpolating polynomial is always reduced to 0. Areas, where the data is missing, are interpolated (hole-filled) with a least-square-mesh using the MLS projected vertices as constraints. Hole-filling tool is also provided by a cleaning toolkit [WPH*04] for the post-processing of raw scanner data. Besides the automatic hole-filling tool, Weyrich et al. [WPH*04] propose other interactive tools to overcome typical scanning artifacts such as noise, outliers, holes, or ghost geometry. And the MLS projection [ABCO*01] is used to define the final reconstructed surfaces.

Yang and Qian [YQ07b] directly process point cloud without normal to construct layered manufacturing model by intersecting a slicing plane with the underlying defined MLS surfaces [AK04a]. The two fundamental components of their algorithm are MLS projection and its differential geometrical analysis that provides the normal and curvature information for MLS surface. The closed formulas for curvature computing [YQ07a][YQ07b] is directly and exactly derived by the expressions of gradient and Hessian of explicit function f in Equation 31.

5.2. Processing

The algorithm of Alexa et al. [ABCO*01][ABCO*03] creates a simplified point cloud that is a true subset of the original point set, by ordering iterative point removal operations according to a surface error metric. Based on a numerical and visual error metric computed by the MLS projection operator [ABCO*03][Lev03], Pauly et al. [PGK02] propose four types of simplification methods for point-sampled geometry, including a particle simulation simplification using MLS projection operator. Mederos et al. [MVD03] create surface approximation by computing a representative point of each cluster on the MLS surface [ABCO*03] of its k-nearest neighbors, and define final refined triangular meshes with points on the MLS surface of the original point cloud. Based on the MLS surface definition (Equation 33) with the weighing function (Equation 51), Pauly et al. [PKG06] propose a discrete multi-scale surface representation by combining the smoothing and decomposition operators into a single projection, which allows efficient construction of the hierarchical multi-resolution. By using the MLS definition (Equation 26) with the adaptive weighting function (Equation 51), Duranleau et al. [DBP08] address a multi-resolution representation for point set surfaces. The multi-resolution method [DBP08] is close to progressive point set surfaces proposed by Fleishman et al. [FACOS03], except

that The basic building of [FACOS03] is the projection operator [ABCO*01] depending on implicit k-neighborhoods.

For MLS surfaces [ABCO*01], Ruggeri et al. [RDSK06] present a technique to computie piecewise linear approximations of geodesics on point-based surfaces. In the following work, Ruggeri et al. [RS08] use the approximate geodesics between anchors to form a geodesic distance matrix, which yields descriptors of the target shape. Consequently, an effective matching method is proposed to recognize point-based models deformed with isometric transformations.

Since MLS surfaces could be described by implicit functions, operations such as shape blending, offsets, deformations and CSG are simple to perform [OBA*03]. By combining unstructured point cloud with the implicit surface definition of the MLS approximation, Pauly et al. [PKKG03] present a hybrid geometry representation that exploits the advantages of implicit and parametric surface models. Based on the combined representation, Pauly et al. propose a free-form shape modeling framework for point-sampled geometry. Müller et al. [MKN*04] present a mesh-free animation algorithm derived from continuum mechanics to simulate elastic, plastic, melting and solidifying physical objects. And the simple MLS surfaces projection [AA03a] is used for dynamic surface reconstruction.

By extending the idea of the local reference domain in the MLS surfaces projection [Lev03] to the construction of a local and global surface distance field [GQ03], Guo et al. [GHQ04a][GHQ04b] build an editing framework on point set surface with several editing tools, such as force tool, free-form deformation, sketching-based editing, embossing/engraving, and smoothing. Guo et al. [GHQ04c] then enhance the editing operations with interactive haptics, so users can have realistic force feedback.

When extracting a watermark from point-sampled geometry, Cotting et al. [CWPG04] use the MLS approximation [Lev03] to establish a one-to-one correspondence between samples in the given point cloud and the original data set. While Li and Guskov [LG05] introduce a multi-scale salient feature extraction algorithm and analyze its performance in the context of approximate alignment of point-based surfaces, based on explicit MLS surfaces definition [AK04a].

5.3. Rendering

5.3.1. Splatting

The early MLS surfaces is visualized by a simple up-sampling and splatting rendering scheme [ABCO*01] [ABCO*03]. The main procedure of their interactive rendering method is to first select the visible samples, then up-sample the tangent plane of each representation point and yield a dense set of splats, project splats onto the pre-computed polynomial approximating the underlying surface, and finally perform the splatting of the generated splats.

Guennebaud et al. [GGG08] trade off the rendering quality and performance by dynamically adjusting the target density. The core of their approach is an adaptive up-sampling scheme based on a view-dependent geometric error metric. The pre-processing of the rendering is to construct an octree data structure for finding all the points. Their per-frame rendering is similar to the procedure of [ABCO*03] except for the projection of the splats onto the algebraic point set surfaces [GG07] with hardware acceleration.

5.3.2. Ray Tracing

Ray-tracing rendering of MLS surface [AA03a][AA03b] amounts to finding points on the ray where the function as defined in Equation 28 evaluates to zero. The ray-surface intersection proceeds in two steps. First, a starting point x_0 close to the reconstructed surface $\mathcal{S}_P$ is computed in the bounding sphere of the projection sample p_i, and an average plane $n(x_0)^T(x-a(x_0))$=0 is constructed. Next, starting from this point x_0, the ray is intersected with a local bivariate approximating polynomial computed by the plane in previous step, and yields a new point x_1. This procedure is repeated until convergence. Since computing the ray-surface intersection is rather time consuming, Adamson and Alexa [AA03b] use octree to accelerate the computation. They try to first find all ray-sphere intersections, and then sort the spheres front to back and perform the ray-surface intersection test for the nearest sphere.

Wald and Seidel [WS05] introduce various optimizations of [AA03b] to achieve interactive frame rates. The performance is increased due to the combination of an efficient surface-intersection algorithm (called ray marching with linear intersection interpolation) and a highly optimized kd-tree acceleration structure. Until now, various extensions and optimizations of the basic algorithms [AA03a][AA03b] have been used to visualize various models, including the animation of elastic and plastic solids [AKP*05], fracturing materials [PKA*05], viscous fluids [KAG*05], point-based shells [WSG05], geometry textures synthesis [DP06], and hardware acceleration of ray tracing on the GPU [TGN*06]. The sequences show high-quality rendering results of point-sampled surfaces with complex shading effects such as shadows, reflections, and refractions, which can't provided by the splatting rendering.

6. Discussion

Table 1 summarizes the typical MLS surface algorithms in terms of type, name, weighting equation, defining formulation, input and output, and applications. It is clear that there is no certain connection between any specific definition and the application, since one application can be realized by different MLS surfaces definitions.

The primary issue of the MLS surface method is the defining form, which is the mathematic and practical fundamental

of its design, implementation and analysis. Although, there are a great deal of methods so far, they can be mainly represented by six typical definitions. (1) The original work [ABCO*01], implemented the two-step (local tangent frame computation and bivariate polynomial approximation) stationary projection theory [Lev03], directly defines point set surfaces from the input point set. Its main extension handles sharp features [FCOS05][DHOS07][LCOL07], and the two step implementation can be transformed to one step by a unified formulation [OSS*07]. (2) The simple intersecting point set surfaces [AA03a][AA03b] is defined by the product of normal and distance vector from the evaluation position to its local weighted centroid. For the point set without normals, when the resolving normal of Equation 21 is well-defined (unique), i.e., the samples locate in the restricted (u,l) sampling domain [BH05], the reconstructed surface is also well-defined. Note that, the normal to the local tangent frame is not the surface normal, and the iterative projection procedure of Equation 26 can be classified into three kinds: basic, almost orthogonal, and orthogonal projector [AA04b]. Currently, the ray-tracing rendering of point set surfaces [AA03a][AA03b] has become one of the two rendering standards in the point-based graphics. The simple MLS definition can be extended to deal with the scatting points of complex models. Special treatments have been performed to obtain bounded, non-orientable MLS surfaces [AA04a], anisotropic [AA06a] and Hermite [AA08] point set surfaces, even point-sampled cell complex [AA06b]. (3) The general explicit MLS surfaces definition is given by minimal extremal surface, where its normal is perpendicular to the gradient of its energy field [AK04a]. The extremal surface has been proven [DGS05] that its projection is converging and isotopic to the original sampled surface, if the uniform (ε,α) sampling condition is satisfied and two nearby sample points have similar assigned normal vectors. The reduced version of the extremal surfaces has been used in the Poinshop3D tools [ZPKG02] and some other work [PKG02][PKKG03] [MHL*05]. (4) Amenta and Kil [AK04] discuss the domain of the point set surfaces, which originates the researches on the properties and conditions of the MLS surfaces, such as the sampling condition [PKG02][PKKG03][AA06a][BH05][DS05][DGS05] [Kol05], sharp features [FCOS05][DHOS07][LCOL07], and complex surface reconstruction [AA04a][AA06a][AA06b] [AA08]. (5) The pure implicit MLS surfaces is defined by the signed Euclidean distance function, which is computed from the position to the tangent planes attached to the local neighboring samples in the MLS sense [SO'S04]. The theoretical guarantees for the implicit MLS surfaces has been provided in the uniform [Kul05] and adaptive [DS05] (ε,α) sampling conditions. (6) The recent algebraic point set surfaces [GG07] is an implicit MLS surfaces by directly fitting algebraic spheres rather than the previous Euclidean planes. From the definition form column in the table 1, it is easy to find the elements used in the formulations: the normal obtained by equation 19 or 21, the signed scalar field that is a locally weighted combination of implicit plane equations $n(x)^T(x-a(x))$ or $n_i^T(x-p_i)$.

The other key issue in the MLS surfaces definition is the weighting function, which is a smooth, non-negative, and monotonically decreasing function. Obviously, the Gaussian function and its variants have dominated the weighting field, just as shown in table 1. The Gaussian width h is one important factor that affects the final results, since it determines the shape of compactly spherical supporting region of each sample. In the pioneering work [ABCO*01][ABCO*03], h is a pre-defined constant value that reflects the anticipated spacing between neighboring points. The optimal h is respectively investigated by [LCOL06] and [WSS08], and used in the second polynomial approximation step of [Lev03] to achieve better polynomial approximation. For the irregular samples, the variant function $h(x)$ should account for the anisotropic settings. The anisotropic $h(x)$, related to the local space, is first intuitively defined by [PGK02] and [PKKG03], then is sensibly computed to represent the anisotropic support regions by the Mahalanoibis distance [AK04a][FCS07] and ellipsoidal weight [AA06a]. Besides these, the variant function $h(x)$ can be carefully defined, so that the sampling space [Kul05][DS05][DGS05][BH05] is proportional to the local feature size, hence, the reconstructed surface in the sampling space is geometrically and topologically correct.

7. Conclusion

The paper has surveyed the methods used in literature for MLS surfaces, by classifying the algorithms, describing main ideas behind each typical approach, and comparing their strongpoints and weaknesses. Especially, we have identified the distinct difference between the *projection MLS surfaces* and the *implicit MLS surfaces*. This difference originates from a different point of view on whether an MLS surface is defined as the stationary projection set of input points or it is created by an implicit scalar field. In general, an MLS surface is determined by the definition form and the weighting function, respecting the sampling condition (such as noise level, sampling density among others) and the specific requirement of the output.

Although there are already numerous techniques for MLS surfaces, it seems that directions to address this problem are only beginning. Just as a measure, one can compare the large number of 2D MLS curve publications to the small number of 3D MLS surface publications. It seems that more advanced issues are still largely open problems and would require further research. Firstly, Future MLS surface schemes will be innovate by new mathematic defining forms, e.g., the algebraic ellipsoidal surfaces representation for anisotropic objects. Secondly, it is important to theoretically analyze both the increasing sensitivity with respect to unwanted stationary points far away form the surface and the sampling requirements with desired blue noise property. Thirdly, further research should be taken on how to effectively use the MLS

surfaces in different applications, e.g., non-manifold reconstruction, reliable quadric surface extraction, partial similarity analysis, interactive editing, animation, and real-time rendering with hardware acceleration.

References

[AA03a] Adamson A., Alexa M.: Approximating and inter-secting surfaces from points. In *Proc. SGP* (2003), 230–239.

[AA03b] Adamson A., Alexa M.: Ray tracing point set surfaces. In *Proc. SMI* (2003), 272–279.

[AA04a] Adamson A., Alexa M.: Approximating bounded, non-orientable surfaces from points. In *Proc. SMI* (2004), 243–252.

[AA04b] Alexa M., Adamson A.: On normals and projection operators for surfaces defined by point sets. In *Proc. PBG* (2004), 149–156.

[AA06a] Adamson A., Alexa M.: Anisotropic point set surfaces. In *Proc. AfriGraphic* (2006), 7–13.

[AA06b] Adamson A., Alexa M.: Point-sampled cell complexes. *ACM TOG* 25, 3 (2006), 671–680.

[AA08] Alexa M., Adamson A.: Interpolatory point set surfaces - convexity and hermite data. *ACM TOG* (2008).

[ABCO*01] Alexa M., Behr J., Cohen-Or D., Fleishman S., Levin D., Silva C. T.: Point set surfaces. In *Proc. IEEE Visualization* (2001), 21–28.

[ABCO*03] Alexa M., Behr J., Cohen-Or D., Fleishman S., Levin D., Silva C. T.: Computing and rendering point set surfaces. *IEEE CGV* 9, 1 (2003), 3–15.

[ACDL02] Amenta N., Choi S., Dey T. K., Leekha N.: A simple algorithm for homeomorphic surface reconstruction. *IEEE CGA* 12 (2002), 125–141.

[AK04a] Amenta N., Kil Y.: Defining point-set surfaces. *ACM TOG* 23, 3(2004), 264–270.

[AK04b] Amenta N., Kil Y.: The domain of a point set surface. In *Proc. PBG* (2004), 139–147.

[AKP*05] Adams B., Keiser R., Pauly M., Guibas L. J., Gross M., Dutre P.: Efficient ray tracing of deforming point-sampled surfaces. *Computer Graphics Forum* 24, 3 (2005), 677–684.

[Blo94] Bloomenthal J.: An implicit surface polygonizer. *Graphics Gems IV*. Acamedic Press Professional, 1994, 324–349.

[BH05] Bremer P.-T., Hart J. C.: A sampling theorem for MLS surfaces. In *Proc. PBG* (2005), 47–54.

[CWPG04] Cotting D., Weyrich T., Pauly M., Gross M.: Robust Watermarking of Point-Sampled Geometry. In: *Proc. SMI* (2004), 232–242.

[DBP08] Duranleau F., Beaudoin P., Poulin P.: Multiresolution Point-set Surfaces. In *Proc. GI* (2008).

[DGS05] Dey T. K., Goswami S., Sun J.: Extremal surface based projections converge and reconstruct with isotopy. *Tech. Rep.*, Ohio State University, 2005.

[DHOS07] Daniels J., Ha L., Ochotta T., Silva C.: Robust Smooth Feature Extraction from Point Clouds. In *Proc. SMI* (2007), 123–136.

[DP06] Duranleau F., Poulin P.: Patch-based Synthesis of Geometry Textures with Point-set Surfaces. In *Proc. VMV* (2006).

[DS05] Dey T. K., Sun J.: An adaptive MLS surface for reconstruction with guarantees. In *Proc. SGP* (2005), 43–52.

[FACOS03] Fleishman S., Alexa M., Cohen-Or D., Silva C. T.: Progressive point set surfaces. *ACM TOG* 22, 4 (2003), 997–1011.

[FCOS05] Fleishman S., Cohen-Or D., Silva C. T: Robust moving least-squares fitting with sharp features. *ACM TOG* 24, 3 (2005), 544–552.

[FCS07] Fiorin V., Cignoni P., Scopigno R.: Out-of-core mls reconstruction. In *Proc. IASTED CGI* (2007), 27–34.

[GG07] Guennebaud, G., Gross, M.: Algebraic point set surfaces. *ACM TOG* 26, 3 (2007).

[GGG08] Guennebaud, G., Germann M., Gross, M.: Dynamic sampling and rendering of algebraic point set surfaces. In *Proc. Eurogrphaics* (2008).

[GHQ04a] Guo X., Hua J., Qin H.: Scalar-Function-Driven Local and Global Editing on Point Set Surfaces. *IEEE CGA* 24, 4 (2004), 43–52.

[GHQ04b] Guo X., Hua J., Qin H.: Point Set Surface Editing Techniques based on Level-Sets. In *Proc. CGI* (2004).

[GHQ04c] Guo X., Hua J., Qin H.: Touch-based Haptics for Interactive Editing on Point Set Surfaces. *IEEE CGA* 24, 6 (2004), 31–39.

[GP07] Gross M., Pfister H.: *Point-based graphics*. MORGAN KAUFMANN publishers, 2007.

[GQ03] Guo X., Qin H.: Free-Form Deformations via Sketching and Manipulating Scalar Fields. In *Proc.* Solid Modeling and Applications (2003), 328–333.

[HDD*92] Hoppe H., DeRose T., Duchamp T., McDonald J., Stuetzle W.: Surface reconstruction from unorganized points. *Computer & Graphics* 26, (1992), 71–78.

[LE87] Lorensen W., ECline H. E.: Marching cubes:a high resolution 3D surface construction algorithm. *Computers & Graphics* 21, 4 (1987), 163–169.

[KAG*05] Keiser R., Adams B., Gasser D., Bazzi P., Dutre P., Gross M.: A unified lagrangian approach to solid-fluid animation. In *Proc. PBG* (2005), 125–148.

[KB04] Kobbelt L., Botsch M.: A Survey of Point-based techniques in computer graphics. *Computers & Graphics* 28, 6(2004), 801–814.

[Kol05] Kolluri R.: Provably good moving least squares. In *Proc. SIAM SDA* (2005), 1008–1018.

[LCOL06] Lipman Y., Cohen-Or D., Levin D.: Error bounds and optimal neighborhoods for mls approximation. In *Proc. SGP* (2006), 71–80.

[LCOL07] Lipman Y., Cohen-Or D., Levin D.: Data-dependent MLS for faithful surface approximation. In *Proc. SGP* (2007), 59–67.

[Lev98] Levin D.: The approximation power of moving least-squares. *Mathematics Computing* 67, 224 (1998), 1517–1523.

[Lev03] Levin D.: Mesh-independent surface interpolation. *Geometric Modeling for Scientific Visualization* (2003), 37–49.

[LG05] Li X., Guskov I.: Multiscale Features for Approximate Alignment of Point-based Surfaces. In *Proc. SGP* (2005), 217–226.

[MaL76] McLain D. H.: Two dimensional interpolation from random data. *The Computer Journal* 19, (1976), 178–181.

[MHL*05] Miao L., Huang J., Liu X., Bao H., Peng Q., Guo B.: Computing variation modes for point set surfaces. In *Proc. PBG* (2005), 63–69.

[MKN*04] Müller M., Keiser R., Nealen A., Pauly M., Gross M., Alexa M.: Point Based Animation of Elastic, Plastic and Melting Objects. In *Proc.* ACM *SIGGRAPH/Eurographics SCA* (2004), 141–151.

[MVD03] Mederos B., Velho L., De Figueiredo L.H.: Moving Least Squares Multiresolution Surface Approximation. In *Proc. Brazilian CGIP* (2003), 19–24.

[Nea04] Nealen A.: An As-Short-As-Possible Introduction to the Least Squares, Weighted Least Squares and Moving Least Squares Methods for Scattered Data Approximation and Interpolation. *Tech. Rep.*, TU Darmstadt, 2004.

[OBA*03] Ohtake Y., Belyaev A., Alexa M., Turk G., Seidel H.-P.: Multi-level partition of unity implicits. *ACM TOG* 22, 3(2003), 463–470.

[OSS*07] Ochotta T., Scheidegger C., Schreiner J., Lima Y., Kirby R. M., Silva C.: A Unified Projection Operator for MLS Surfaces. *Tech. Rep.*, Utah University, 2007.

[PGK02] Pauly M., Gross M., Kobbelt L. P.: Efficient simplification of point-sampled surfaces. In *Proc. IEEE Visualization* (2002), 163–170.

[PKA*05] Pauly M., Keiser R., Adams B., Dutre P., Gross M., Guibas L. J.: Meshless animation of fracturing solids. *ACM TOG* 24, 3(2005), 957–964.

[PKG06] Pauly M., Kobbelt L. P., Gross M.: Point-based multi-scale surface representaition. *ACM TOG* 25, 2(2006), 177–193.

[PKKG03] Pauly M., Keiser R., Kobbelt L. P., Gross M.: Shape modeling with point-sampled geometry. *ACM TOG* 22, 3(2003), 641–650.

[Pra87] Pratt V.: Direct least-squares fitting of algebraic surfaces. In *Proc. SIGGRAPH* (1987), 145–152.

[RDS*06] Ruggeri M. R., Darom T., Saupe D., Kiryati, N.: Approximating geodesics on point set surfaces. In *Proc. PBG* (2006), 85–93.

[RJT*05] Reuter P., Joyot P., Trunzler J., Boubekeur T., Schlick C.: Surface reconstruction with enriched reproducing kernel particle approximation. In *Proc. PBG* (2005), 79–87.

[RS08] Ruggeri M. R., Saupe D.: Isometry-invariant matching of point set surfaces. In *Proc. Eurographics Workshop on 3D Object Retrieval* (2008).

[SFS05] Scheidegger C., Fleishman S., Silva C.: Triangulating Point-Set Surfaces With Bounded Error. In *Proc. SGP* (2005), 63–72.

[SLS*06] Sharf A, Lewiner T, Shamir A, Kobbelt L, Cohen-Or D.: Competing fronts for coarse-to-fine surface reconstruction. *Computer Graphics Forum* 25, 3 (2006).

[SO'S04] Shen C., O'Brien J. F., Shewchuk J. R.: Interpolating and approximating implicit surfaces from polygon soup. *ACM TOG*, 23, 3(2004), 896–904.

[SSFS06] Schreiner J., Scheidegger C. E., Fleishman S., Silva C. T.: Direct (re)meshing for efficient surface processing. *Computer Graphics Forum* 25, 3(2006).

[TGN*06] Tejada E., Gois J. P., Nonato L. G., Castelo A., Ertl T.: Hardware-accelerated Extraction and Rendering of Point Set surfaces. In *Proc. Eurographics/IEEE Symposium on Visualization* (2006), 21–28.

[WPH*04] Weyrich T., Pauly M., Heinzle S., Keiser R., Scandella S., Gross M.: Post-processing of scanned 3D surface data. In *Proc. PBG* (2004), 85–94.

[WS05] Wald I., Seidel H.-P.: Interactive ray tracing of point-based models. In *Proc. PBG* (2005), 9–16.

[WSG05] Wicke M., Steinemann D., Gross M.: Efficient animation of point-based thin shells. In *Proc. Eurographics* (2005), 667–676.

[WSS08] Wang H., Scheidegger C., Silva C.: Optimal Bandwidth Selection for MLS Surfaces. In *Proc. SMI* (2008).

[YQ07a] Yang P., Qian X.: Direct computing of surface curvatures for point-set surfaces. In *Proc. PBG* (2007), 29–36.

[YQ07b] Yang P., Qian X.: Adaptive slicing of moving least squares surfaces: toward direct manufacturing of point set surfaces. In: *Proc. ASME DET & CIE* (2007).

[ZPKG02] Zwicker M., Pauly M., Knoll O., and Gross M.: Pointshop3D: An interactive system for point-based surface editing. *ACM TOG* 21, 3 (2002), 322–329.

Table 1: Summary of typical MLS surfaces techniques.

Type	Algorithm With Name	Weighting function (Equation No.)	Defining Form (Equation No.)	Input, Output	Applications
Projection MLS Surfaces	Formulation of stationary projection with two steps [Lev03]	$w(d)=e^{-d^2/h^2}$ (4)	$f(x)=x$ (13)	Sampled points, smooth curves and one quadric surface	Surface reconstruction [ABCO*03] meshing [SFS05] [SSFS06], hole filing [SLS*06] [WPH*04], simplification [MVD03] and compression [FACOS03], isometric transformation [RS08], editing [GQ03] [GHQ04a] [GHQ04b] [GHQ04c], watermark [CWPG04], splatting rendering [ABCO03*]
	Point set surfaces [ABCO*01] [ABCO*03]	$w(d)=e^{-d^2/h^2}$ (4)	$f(x)=x+(t+g(0,0))n$ (18)	Clean points with/without normals, Smooth surfaces	
	Robust MLS surfaces [FCOS05]	$w(d)=e^{-d^2/h^2}$ (4)	Classification $\mathcal{P}$ and MLS surfaces approximation of sub-regions by Equation 18	Possibly noisy point cloud, Piecewise	Surface reconstruction with sharp features [FCOS05]
	Smooth feature extraction [DHOS07]	$w(d)=e^{-d^2/h^2}$ (4)	Extraction of feature curves by leveraging the RMLS method [FCOS05]	smooth surfaces with sharp features	Overcome the the jittering artifacts of the RMLS [FCOS05] method [DHOS07]
	Data-dependent MLS surfaces [LCOL07]	$w(d)=e^{-d^2/h^2}$ (4)	$f(x)=\arg\min_{f\in\mathcal{S}_P}\sum_i w_i(\|x-p_i\|)\|f(p_i)-f_i\|^2$ (68)		Handle the singularity problem [FCOS05] in sparse or noisy point coulds, compression [LCOL07]
	Unified MLS surfaces [OSS*07]	$w(d)=e^{-d^2/h^2}$ (4)	$\arg\min_{q,f}\sum_i\|C_{H,f}(x)-p_i\|^2\, w(\|C_{H,f}(x)-p_i\|)$ (71)	Similar to [ABCO*01]	Surface reconstruction with possible sharp features [OSS*07]
	Simple point set surfaces [AA03a] [AA03b] [AA04b]	$w(d)=e^{-d^2/h^2}$ (4)	$f(x)=n(x)^{\mathrm{T}}(x-a(x))$ (26)	Similar to [ABCO*01]	Multi-resolution [DBP08], mesh free animation [MKN*04], ray-tracing rendering [AA03a] [AA03b] [WS05] [AKP*05] [PKA*05] [KAG*05] [WSG05] [DP06] [TGN*06]
	Well-defined point set surfaces[BH05]	$w(d)=e^{\frac{(25d)^2}{lfs^2(\hat{p}_i)}}$ (50)	$f(x)=n(x)^{\mathrm{T}}(x-a(x))$ (26)	(u,l) sampling without normal, Smooth surface	Sampling space of each point with unique resolving normal, and well defined surface reconstruction[BH05]
	Bounded and non-orientable MLS surfaces [AA04a]	$w(d)=e^{-d^2/h^2}$ (4)	$f(x)=n(x)^T(x-a(x))=0\wedge c(x)<\delta_c^*$ (56)	Clean models with/without normals, samples in the proximity of $\mathcal{S}_\mathcal{P}$ and $\mathcal{S}_\mathcal{P}\subset\cup B_{p_i}$, Bounded or non-orientable manifold surfaces [AA04a]	
	Anisotropic point set surface [AA06a]	$w(d)=e^{-d^2/\|H_i\|^2}$ (52)	$f(x)=n(x)^{\mathrm{T}}(x-a(x))$ (26)	Similar to [ABCO*01]	Anisotropic reconstruction with ellipsoids [AA06a]
	Hermite Point Set Surfaces [AA08]	$w(d)=d^{-k}(k\geq 2)$ (57)	$f(x)=n(x)^T(x-\tilde{a}(x))=0$ (59)	Convex input, Convex output	Hermite surface reconstruction [AA08], Initialization step of point-sampled cell complexes [AA06b]
	Point–sampled cell complexs [AA06b]	$w(d)=\begin{cases}(1-\frac{d}{h})^4(\frac{4d}{h}+1) & 0\leq d\leq h\\ 0 & d>h\end{cases}$ (61) and (63)	$Q_M(d)=\begin{cases}Q_M^\infty(\mathrm{x}) & b_j^T(Q_M^\infty(x)-q_j^{D-1}(x))\geq 0\\ q_j^{D-1} & b_j^T(Q_M^\infty(x)-q_j^{D-1}(x))<0\end{cases}$ (65)	Point samples, Cell complex	Modeling a piecewise smooth surface as a cell complex [AA06b]
	Defining point set surfaces [AK04a]	$w(d)=e^{-d^2/h^2}$ (4)	$f(x)=n(x)^{\mathrm{T}}(\frac{\partial e_{MLS}(y,n(x))}{\partial y}\|_x)$ (30)	Similar to [ABCO*01]	Reconstruction [AK04a] [YQ07b], out-of-core [FCS07], surface matching [LG05], surfel rendering [AK04a]
	Extremal MLS surfaces of converging projection [DGS05]	$w(d)=e^{-d^2/h^2}$ (4)	$f(x)=n(x)^{\mathrm{T}}(\frac{\partial e_{MLS}(y,n(x))}{\partial y}\|_x)$ (30)	Similar to [ABCO*01]	Theoretical guarantees for Projection MLS surfaces [DGS05]
	Adaptive MLS Projection [PGK02] [PKKG03] [ZPKG02]	$w(d)=e^{-d^2/h_r^2}$ (61)	$f(x)=\sum_i w(\|x-p_i\|)\left(n_i^{\mathrm{T}}(x-p_i)\right)^2$ (33)	Similar to [ABCO*01]	Simplication [PKG02], modeling and CSG [PKKG03], multi-scale surface representation [PKG06]
	Variant point set surfaces [MHL*05]	$w(\|x-p_i\|)=e^{-\frac{\|x-p_i\|^2}{h^2}}\cdot e^{-\frac{\|x-\phi_{p_i}(x)\|^2}{(h^*)^2}}$ (69)	$f(x)=\sum_i w(\|x-p_i\|)\left(n_i^{\mathrm{T}}(x-p_i)\right)^2$ (33)	Similar to [ABCO*01]	Smoothment and enhancemet of the surface, and preserving sharp features [MHL05]
	Domain of point set surfaces [AK04b]	$w(d)=e^{-d^2/h^2}$ (4)	The energy function of center of mass (48) and line integral (49) meeting Equation 29	Similar to [ABCO*01]	The conditions and properties of MLS surfaces [AK04b]
Implicit MLS Surfaces	Implicit MLS surfaces [SO´S04]	$w(d)=\left(\frac{1}{d^2+\varepsilon^2}\right)^2$ (34)	$f(x)=\frac{\sum_i\int_{Poly_i}w(\|x-p\|)(f_i+n_i^{\mathrm{T}}(x-p_i)^T)dp}{\sum_i\int_{Poly_i}w(\|x-p\|)dp}$ (37)	Polygon soup, Smooth surfaces	Surface reconstruction, level of detal, bounding volume creation, deformable object simulation [SO´S04]
	Uniform Implict MLS surfaces [Kul05]	$w(d)=e^{-d^2/(\rho\cdot lfs)^2}/ns_i$ (39)	$f(x)=\frac{\sum_i w(\|x-p_i\|)\left(n_i^{\mathrm{T}}(x-p_i)\right)^2}{\sum_i w(\|x-p_i\|)}$ (38)	Uniform sampling space, Smooth surface	Theoretical guarantees for IMLS algorithm in the uniform (ε,α) sampling condition
	Adaptive Implicit MLS surfaces [DS05]	$w(d)=e^{-\sqrt{2}d^2/\left(\rho^2 lfs(\hat{x})lfs(\hat{p}_i)\right)}$ (40)	$f(x)=\frac{\sum_i w(\|x-p_i\|)\left(n_i^{\mathrm{T}}(x-p_i)\right)^2}{\sum_i w(\|x-p_i\|)}$ (38)	adaptive sampling space, Smooth surface	Theoretical guarantees for IMLS algorithm in the adaptive (ε,α) sampling condition
	Algebraic point set surfaces [GG07]	$w(d)=\begin{cases}\left(1-\left(\frac{d}{h_i(x)}\right)^2\right)^4 & x<1\\ 0 & \text{otherwise}\end{cases}$ (42)	$f(x)=[1,x^T,x^Tx]u(x)=0$ (41)	Similar to [ABCO*01]	Surface representation with sharp features [GG07], dynamic splatting rendering [GGG08]