Splat representation of parametric surfaces

Dolors Ayala, Núria Pla and Marc Vigo

Abstract

Point-based geometry representations and their splat-based generalizations have become a suitable technique both for modeling and rendering complex 3D shapes. So, it seems interesting to convert other kind of models to a point or splat-based representations.

In this work, we present an approach to convert a parametric surface to an elliptical splat-based representation. Although this conversion supposes a loss of information going from an analytical to an approximate model, it will allow to locally modify zones with complex features, to mix surface and splat-based models and to take advantage of the existing point-based rendering methods.

The presented approach works in the parametric space and performs an adaptive sampling based on the surface curvature and a given error tolerance. The goal is to obtain an optimized set of elliptical splats that completely covers the surface. Two strategies are presented, one based on a quadrangular subdivision of the parametric space and the other on power Voronoi diagrams. Finally, some open problems are enumerated.

1 Introduction

Recently point-based models have caught the attention of the computer graphics community. The point primitive is simple and powerful. It allows to represent very complex shapes. It is becoming a serious rival of the very extended triangular meshes. It has the advantage that we can get rid of topological information.

From [1], a comparative can be made between triangle meshes and point-based models. Triangle meshes are a simple and efficient representation, they have hardware rendering pipeline support and the triangle seems to be the most widely accepted graphics primitive. They allow multi-resolution and compression techniques to be developed based on this model. On the other hand, point-based models are even simpler representations because no connectivity information is needed, they can take advantage of efficient splatting rendering working in image space with reconstruction techniques as well as in object-space with re-sampling and they show the same capabilities for multi-resolution and compression techniques [1], [4].

Recently a great number of proposals have been presented based on this primitive going from modelling methods to efficient rendering approaches and also taking into account issues as shape modeling, simplification methods and editing operators among others.

There are two basic kinds of primitives: plain point primitives with a simple 3D point location geometry associated and splat primitives which are associated with a normal vector and one or two radii, depending on if they are circular or elliptical. Both primitives can also carry color and other material properties for visualization purposes.

The motivation of this work is based on the ascending interest in this kind of representation. As far as we know, the application that more intensively has motivated the study of point-based models is to have a suitable model that could represent real objects captured from 3D acquisition systems (see Figure 1, left).

On the other hand, synthetic, CAD surface representations (see Figure 1, right) are still mostly represented by triangle meshes and we think that it could be interesting to be able to represent them also with a point based geometry. Therefore, as it seems that point-based models could complement or even substitute triangular meshes, it can be interesting to devise conversion algorithms from surface to point-based representations as it has also been interesting to convert them into triangular meshes.

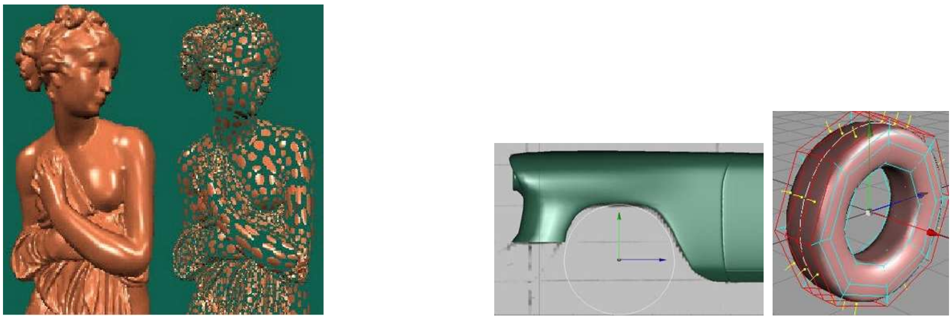

Figure 1: Left: 3D model acquisition. Right: two CAD models

Some advantages of it will be that we will be able to locally modify zones of complex features and to have mixed scenes with modelled surfaces and genuine point-based models, i. e. obtained from capture systems.

2 Outline of the method

The problem can be stated as follows: given a parametric surface, convert it to a splat representation. The strategy consists in covering the parametric space (PS) with a set of circular splats by applying an incremental method. A similar problem that converts a parametric surface to a triangular mesh has been studied in [2], [7], [6].

The following properties have to be fulfilled:

- full coverage of the PS with a set of circular splats that will be elliptical splats in the image space
- admissibility (with a given tolerance ϵ) for any point p of the surface, there exists a point q in the point-based representation such that $\|pq\| < \epsilon$
- curvature adaptive
- number of splats optimization

The algorithmic approach is as follows. We first compute admissibility bounds based on second derivatives across the PS. They will allow us to control the admissible splat radius at any point of the PS and we will be able to compute them in a discretized or exact way.

Then we will apply an heuristic incremental method. We have developed two approaches: one plain point-based and the other based on power Voronoi diagrams.

3 Admissibility bounds

To obtain the admissibility bounds, we consider that the circular splats placed in the PS are ellipses on a plane, Π tangent to the surface at the image of the center point (u_0, v_0). See figure 2.

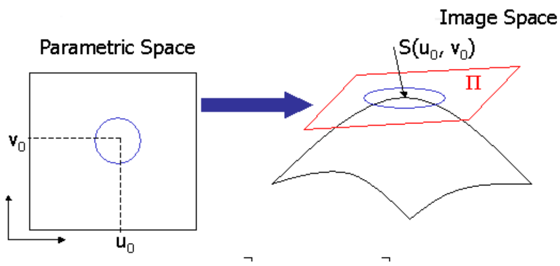

Figure 2: Splat in parametric and image spaces

The equation of the tangent plane Π is:

$$\Pi(u,v) = S(u,v) + \left[\frac{\partial S}{\partial u}\right]_{(u_0,v_0)} (u-u_0) + \left[\frac{\partial S}{\partial v}\right]_{(u_0,v_0)} (v-v_0)$$

The imposed condition is that the distance between this plane and the surface is bounded by an admissible error ϵ. We instead use the distance between a point on the surface and the corresponding point on the tangent plane (see Figure 3). Then, we have that:

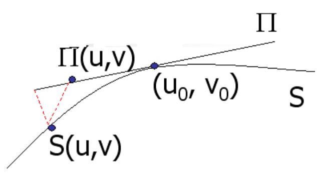

Figure 3: Admissibility condition

$$\|S(u,v) - \Pi\| \leq \|S(u,v) - \Pi(u,v)\| < \epsilon$$

And applying the Taylor series approximation,

$$\|S(u,v) - \Pi(u,v)\| \leq \frac{1}{2}\|M_{uu}(u-u_0)^2 + 2M_{uv}(u-u_0)(v-v_0) + M_{vv}(v-v_0)^2\| \leq \epsilon$$

M_{uu}, M_{uv}, M_{vv} are the second derivatives at an arbitrary point of the splat centered at (u_0, v_0) with the radius determined by (u, v).

From this expression, we can obtain a maximum admissible radius for the splat centered at any point p of the PS and we represent it by the function ρ:

$$\rho(p) = \sqrt{\frac{2\epsilon}{M_{uu} + 2M_{uv} + M_{vv}}}$$

Actually this function is valid for any point individually, it is the condition imposed by a point. But, as we want a maximum admissible radius for a zone represented by a splat, we have to devise a more restrictive bound that takes into account the ρ function of all those points in the splat centered at p with radius $\rho(p)$

$R(p)$ is this new bound and it is defined by the following expression:

$$R(p) = min_{q \in C_\rho(p)} \{max\{\|pq\|, \rho(q)\}\}$$

$C_\rho(p)$ stands for the circle centered at p with radius ρ.

4 Point-based approach

Our first approach is inspired in the work presented by Wu and Kobbelt [8] that obtains a set of splats from a given set of sufficiently dense sampled points. They define a splat for each point and apply a growing procedure for each splat based on a coverage relation in such a way that any splat is able to cover several points and, therefore, this initial set of splats is overestimated. To select an active subset of it they apply an heuristic based on the incremental surface area contribution (*isac*): at each step the splat with maximum *isac* is selected and the *isac* of the remaining candidates is updated.

In our case, we have all the continuous PS but we will discretize it into a quadrangular grid in order to obtain a finite number of candidates. We also will define the concept of *isac* [3] analogously as in the mentioned approach and then splats will be selected by descending *isac* until the whole PS is covered.

This quadrangular grid has to guarantee that there exists a finite number of admissible splats that covers the PS. Therefore, we will compute a sampling distance d in such a way that the PS could be fully covered with circles with the minimum admissible radius r, centered at the grid points.

This minimum radius will be computed from the maximum values of second derivatives in the way shown in the previous section:

$$r = \sqrt{\frac{2\epsilon}{M}}, M = max_{q \in [0,1]^2} \|M_{uu}(q) + 2M_{uv}(q) + M_{vv}(q)\|$$

and from it we get the distance and number of grid divisions:

$$d = \frac{1}{n} = \sqrt{2}r = \sqrt{\frac{4\epsilon}{M}}$$

$$n = \left[\sqrt{\frac{M}{4\epsilon}}\right] + 1$$

This grid (the black one in Figure 4 left) is suitable to locate the splats centers, but if we base our covering strategy only on these points, it will produce holes (see Figure 4 middle). Wu and Kobbelt have also to tackle with a similar problem and they perform a reduction of the size of the splat by not considering those points belonging to the boundary of the convex hull of the splat points.

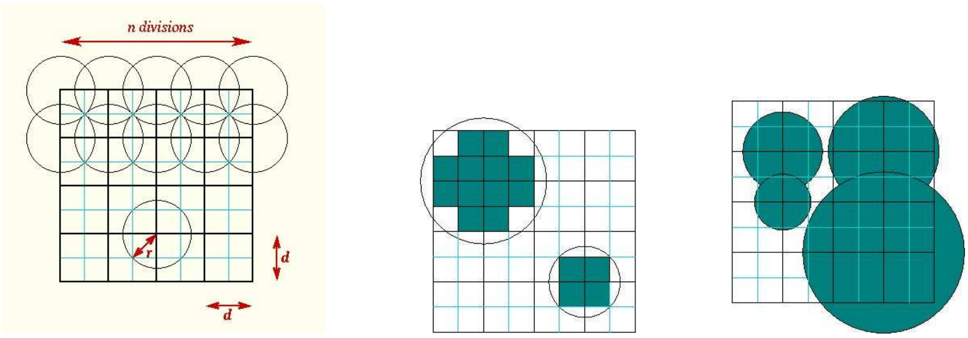

Figure 4: The quadrangular grids considered

We deal with this problem by considering quadrangular regions (pixels) instead of points. So we further subdivide the mesh using a user defined factor greater or equal of two (the green grid shown in Figure 4 is one half of the initial one). Then, each splat is assigned with the set of pixels, instead of points, it covers (see Figure 4 right).

To sum up, this point-based strategy begins by obtaining the set of candidate splats centered at points of the initial grid with the corresponding radius computed with the R function (see Section 3. Then, two lists are generated and maintained (as in [8]). There is an SplatList in which each splat is assigned the set of quads completely covered by it and which is ordered in descending number of quads. But there is also a dual QuadList in which each quad is assigned the set of splats that covers it.

The first splat selected is that with the maximum number of quads and the next splat selection follows also the criterion of *isac* and the selected splat is that one with the maximum number of quads not already covered. At each step both lists are updated and therefore the selected splat is always the splat on the head of the SplatList. The end condition is fulfilled when all the quads have been covered, which happens when the QuadList is empty. Next, there is an outline of the main algorithm:

$Obtain(SplatList, QuadList)$
$SelectFirstSplat(SplatList, Si)$
while $(ListEmpty(QuadList)$ **do**
    for Qj **in** Si **do**
        for Sk **in** $Coverage(Qj)$ **do**
            $RemoveList(SplatList, Sk, Qj)$
        endfor
        $RemoveList(QuadList, Qj)$
    endfor
    $ReSortList(SplatList)$
    $SelectFirstSplat(SplatList, Si)$

endwhile

5 Voronoi-based approach

The second approach is based on the Power Voronoi Diagram (PVD) [5] which is computed using the additively weighted power distance $d_{W2}(p, q) = pq^2 - w_p$, taking as weights the computed R function, $w_p = R(p)^2$. This distance, in general is neither symmetric nor positive and the associated PVD has the following properties (see Figure 5).

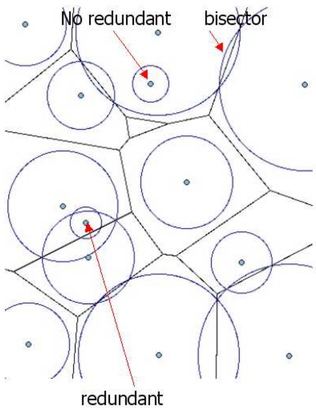

Figure 5: A Power Voronoi diagram showing a redundant vertex and the bisector of two tiles with intersected circles

- the tile size is proportional to the weight
- the bisector of two tiles is such that if the corresponding circles intersect, it passes through their intersection points
- Some points may be redundant with a corresponding empty tile. Moreover, the circle corresponding to a redundant point is always interior to other circles. But the contrary is not true: there can be points with the corresponding circle interior to other circles that aren't redundant points.
- the dual of the PVD is the so called regular triangulation

The set of splats considered in this strategy is the set of circles of one such PVD and all circles interior to other ones are actually redundant splats, although some of them are redundant points and some others aren't. Figure 5 shows both cases.

The strategy for first splat selection is to select the four borders of the PS (the blue ones in Figure 6). It is a suitable initial set S from which we can construct the first $PVD(S)$. Then, the selection of the next splat and the end condition will be guided by this PVD. At each step, new splat centers are added to S and the $PVD(S)$ is updated locally with an incremental process.

For the next splat selection we have devised the following heuristic. Let CS be the set of candidates, defined as

$$CS = VV + IVP - CV$$

VV and IVP stand for the set of Voronoi vertices and the set of intersection points between Voronoi edges and the PS bounds, respectively and IVP stands for the set of vertices belonging to the previous sets and not already covered. Figure 6 shows the CS set for one step of the method.

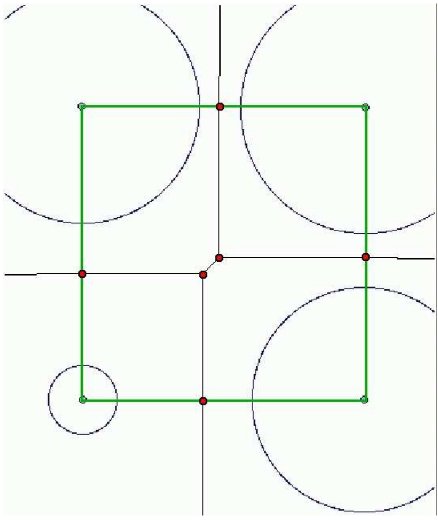

Figure 6: First step of the method: there are two Voronoi vertices and four intersection points. There is any point belonging to CV

Then, the most simple and straightforward strategy to select a candidate would be to select that one with the maximum radius. But this strategy wouldn't take into account the overlapping between splats. On the other hand, considering the overlapping in an exact way would imply doing Boolean operations between splats. So, we have devised a strategy that takes into account the possible overlapping in an approximate way.

For the selection of a candidate we could test all the splats already placed, but we select a proper subset of them, referred as its neighborhood, $N(x) = \{y \mid y \in S \wedge x \in VT(y)\}, x \in SC$. $VT(y)$ stands for a Voronoi tile having y as a vertex, i. e., such that x belongs to the Voronoi region of y. Figure 7 left shows an example of such neighborhood. Then, we select the candidate sx which fulfils the following expression:

$$sx = max_{s \in SC}\{min\{R(x), min_{y \in N(x)}\{\|xy\| - R(y)\}\}\}$$

This expression can be interpreted in a geometric way. Suppose that we assign to each candidate splat a radius corresponding to the maximum circle that does not overlap with its neighbors (see Figure 7 right). Then, for each neighbor y we compute the distance between centers and subtract the radius of y and, finally, the minimum of these values, bounded by the radius of the candidate, is the new radius assigned to the splat. The selected splat is the candidate with the maximum value of this new shrunk radius. Figure 8 shows two steps of the method.

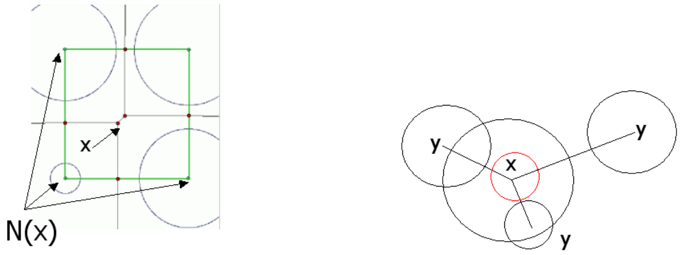

Figure 7: The neighborhood of a candidate x (left) and the geometric interpretation of the new associated radius, in red, to x (right)

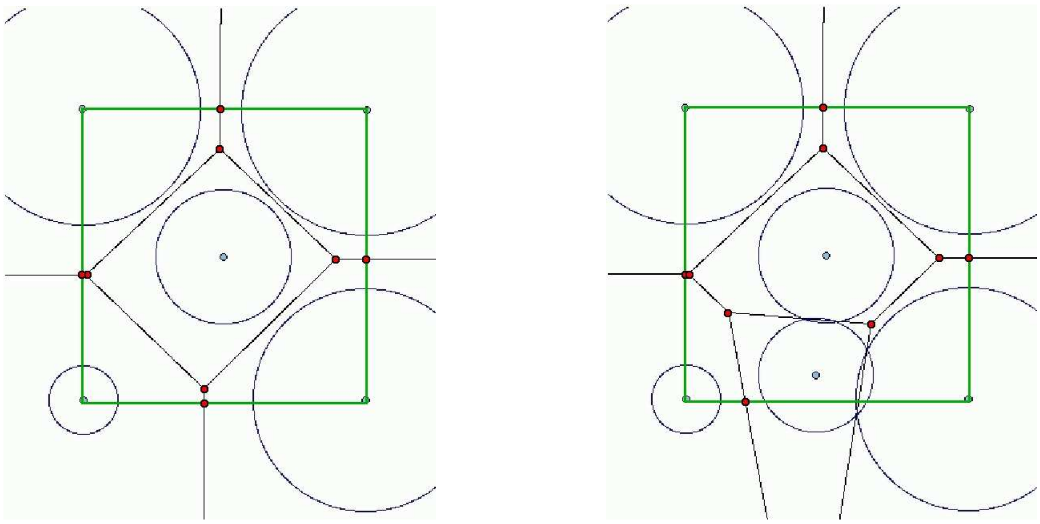

Figure 8: Two more steps of the method. On the right we can see an intersection point that belongs to CV, i.e. it is already covered (it is not marked in red).

The algorithm finishes when the whole PS is covered, i. e., each Voronoi tile falls inside its corresponding circle (see Figure 9). Now we are going to prove the correctness and convergence of it.

For the correctness we have to prove the following double implication: the PS is fully covered $\Leftrightarrow$ SC is empty, which is the same as that each Voronoi tile falls inside its corresponding circle. It can be proved by the two following properties that can be easily derived from the definition of the additively weighted power distance:

- if a point in a Voronoi edge belongs to one circle, it belongs to the two circles of the VT sharing the edge
- if a Voronoi vertex belongs to a circle, it belongs to all the circles of the VT converging in it

To prove the convergence, we first have to observe that the selected splats are always centered in zones not already covered. Then we have to devise a finite bound for the number of splats placed in this way, ns.

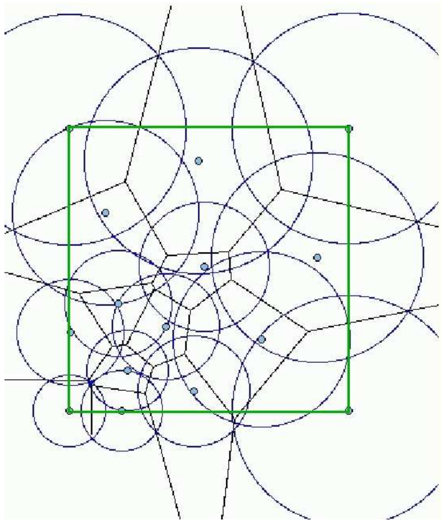

Figure 9: A case with the whole parametric space covered

If we consider the worst case, i.e., that all the placed splats have the minimum admissible radius r_{min}, then any two points will be at a maximum distance of r_{min}. Then we can consider the number of disjoint circles with radius $r_{min/2}$ that we can place centered at any point, $nmin$ and this number, which is an upper bound of ns, is also bounded as shows the following expression:

$$ns \leq nmin < 1/(\pi(r_{min}/2)^2)$$

The algorithm can generate redundant splats, although the strategy followed seems that there will be a little number of them. Some of them correspond to redundant Voronoi vertices and can be removed when updating the PVD. The remaining only could be removed with a post-process.

6 Conclusions and future work

The work is still in progress and there are several open questions. We can guarantee the coverage of the PS but we have to study what will happen in the image space, specially in zones with large curvature. Moreover, splats have to be clipping at the contour of patches and we also want to consider the application of the algorithms to surfaces with several patches. Finally, we want to study the possibility of enabling to displace the splats along the normal vector direction allowing them to cut the surface. It will require a different computation of the R function as the distance between the surface and the tangent plane will be defined in a different way.

7 Acknowledgments

This work has been partially supported by the projects MAT2002-04297-C03-02 and IM3: Imagen Molecular y Multimodalidad from the Spanish government and by the

CREBEC from the Catalan government. We also would thank to the Dagstuhl centre and the organizing committee of Geometric modeling to give the opportunity of publishing this work

References

[1] M. Alexa, M. Gross, M. Pauly, H. Pfister, M. Stamminger, and M. Zwicker. Point-based computer graphics siggraph 2004 course notes, 2004.

[2] P. Brunet and M. Vigo. Piecewise linear approximation of trimmed surfaces. *Computing Suppl.*, 10:341 – 356, 1995.

[3] X. Dcoret, F. Durand, F. X. Sillion, and J. Dorsey. Billboard clouds for extreme model simplification. *ACM Trans. on Graphics*, pages 689 – 696, 2003.

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

[5] A. Okabe, B. Boots, and K. Sugihara. *Spatial Tesselations. Concepts and appliactions of Voronoi Diagrams.* J. Wiley & Sons, Ltd., 1992.

[6] M. Vigo, N. Pla, and D. Ayala. Two triangulation methods based on edge refinement. *Computing*, 72:221 – 233, 2004.

[7] M. Vigo, N. Pla, and P. Brunet. Directional adaptive surface triangulation. *Computer-aided geometric design*, 16:107 – 126, 1999.

[8] J. Wu and L. Kobbelt. Optimized sub-sampling of point sets for surface splatting. *Computer Graphics Forum*, 23(3):643 – 652, 2004.