Point Set Surfaces

1. Briefly summarize the paper's contributions. Does it address a new problem? Does it present a new approach? Does it show new types of results?

- **[AS]**
  This paper presents a new approach to define a smooth manifold surface from a set of points, using the moving least squares approach. The surface is then re-sampled by projecting the original points, possibly noisy, onto the surface, so that the points do not deviate from the surface being represented. This point set can then be down-sampled to remove redundant points, or up-sampled if a denser point cloud is required, and is used to render the surface.
- **[DS]**
  This paper presents a method to represent and render surfaces using point sets. It uses the moving least squares (MLS) method to approximate a surface locally. It presents a way to adjust the fidelity of the representation by introducing methods to increase/decrease the density of the points.
  A novel rendering technique is presented that evaluates local maps based on image resolution.
  New techniques are used to resample the surface.
- **[FP]**
  The authors propose an algorithm to project an arbitrary 3D space point over a point sampled surface. The point projection works in two phases: first, project the point to an approximate tangent plane, then, shift this projection along the normal direction. Both phases are computed using a MLS approach. In contrast to previous papers, this paper does not pursue a surface representation using geometrical primitives (triangles or tetrahedra), neither an implicit representation (i.e., a zero level set). Instead, it focus on describing how the point projection technique can be used to downsample (get a more compact representation) or upsample the point set for rendering purposes.
- **[JD]**
  The paper presents a novel approach for defining an implicit surface from a set of points. The algorithm produces a new set of points from the input by either decimating redundant/noisy points or adding points where there is low sampling density. It can be seen as an improvement algorithm rather than a surface reconstruction algorithm because the output is still a point set. The paper also presents a method of rendering the point set.
- **[LF]**
  The paper presents an approach for representing surfaces as point sets. Points are projected onto the MLS-surface they define (reducing noise). The point set is decimated, iteratively removing points are not important to the definition of the shape, and also up-sampled where necessary.
- **[MK]**
  The paper presents an extension of the MLS approach proposed by McLain to the context of surface reconstruction. At each point p, the method first finds a locally-weighted best fit-plane and then solves for the polynomial function whose graph best fits the (locally-weighted) samples. Evaluating the function at the projection of p onto the plane gives the projected position of p onto the surface.

2. What is the key insight of the paper? (in 1-2 sentences)

- **[AS]**
  The key insight of this paper is that the surface defined as the points that project onto

themselves is a 2D manifold, and this surface is infinitely smooth as long as the weighting function is infinitely smooth.

- **[DS]**
  The key insight is the introduction of the use of a point set representation universally, ie it is used throughout the entire graphical representation's life cycle. Other important insights are their projection procedure onto an MLS surface (non-linear optimization, requires less iterations) and their rendering algorithm (including up/down-sampling of points).
- **[FP]**
  Local reconstruction is done by fitting a polynomial function that represents movement along normal direction from a tangent plane. Both plane and polynomial are fitted by a MLS approach.
- **[JD]**
  The key insight is to use the definition of local maps from differential geometry approximated by MLS to define an implicit surface which can than be re-sampled. The algorithm uses 3rd and 4th order polynomials as surface patches to define the implicit surface.
- **[LF]**
  The key insight of this paper was perhaps the idea of representing a surface as a point set, attracted to an MLS surface and then reduced and elaborated as necessary to achieve a consise reconstruction.
- **[MK]**
  The key idea of this paper is that a surface can be implicitly represented as the set of fixed points of a "projection" operator.

3. What are the limitations of the method? Assumptions on the input? Lack of robustness? Demonstrated on practical data?

- **[AS]**
  The Gaussian weighting function is not properly able to account for sharp features. A Gaussian with a small radius can be used to better capture sharp features, but will also capture more noise in the surface.
- **[DS]**
  The input is a set of points that lie close to a surface.
  Showed improved boundary and normal continuity compared to other methods.
- **[FP]**
  The algorithm does not provide a surface representation using geometrical primitives. Therefore, there is no global structure from which topological properties (e.g., connected components, genus) or geometrical attributes (e.g., curvature, geodesics) could be computed.
  Other (partial) limitation is the requirement of solving a nonlinear problem for each point projection. The authors implement an iterative method to approximate local optima, but there are no guarantees on the quality of the result. In practice, the approximation worked fine on the results presented by the authors, but the rate of convergence or the quality of the approximated optima could strongly depend on the spatial distribution of the set of points.
- **[JD]**
  Point sets up to 900K points were displayed about 5 fps. The implicit surface algorithm assumes nothing about the input points and constructs normals on its own. It is unclear whether the method can handle boundaries.

- **[LF]**
  The limitations of the method are perhaps that it is susceptable to all the vulnerabilities/limitations of MLS as described. The radial weight function chosen is Gaussian and so requires specification of a locality parameter h. The authors also mention that a non-radial function might be more favorable for modeling sharp features.
- **[MK]**
  A limitation of this approach is that it uses the original formulation of projection given by Levin, which is not actually a projection.
  Also, as noted later by Amenta and Kil, the set of points that will project onto the surface are those that are within a rather narrow band of the actual surface (which may not even contain the original samples, if the input data was noisy.)
  Finally, the method doesn't explicitly generate a surface, so it cannot be directly applied to surface reconstruction.

4. Are there any guarantees on the output? (Is it manifold? does it have boundaries?)

- **[AS]**
  The output is guaranteed to be a C^\infty manifold surface.
- **[DS]**
  A smooth manifold is produced (no holes when rendering).
  Produces smooth silhouettes and normals so that there are less distortions on the boundary and reflections.
- **[FP]**
  The authors claim that the set of point which are invariant to projection (i.e., the q+(0,0)n points) form a 2D manifold (wow!). It is also conjectured that this manifold is $C\infty$ as long as the weighting function is C^{∞}, radial, decreasing and positive.
  Weighting the contributions by using the distance to q, instead of distance to r, seems to make the algorithm more robust to outliers. Dividing the projection on two stages (first fitting a plane and then a fitting a polynomial) also seems to provide a more accurate reconstruction, than trying a one-step approach.
- **[JD]**
  The output is guaranteed to be an infinitely smooth surface. The output does not have boundaries.
- **[LF]**
  The output should be a C-infinity smooth, 2-manifold given a sufficiently point density.
- **[MK]**
  Assuming that the projected points are chosen "close enough" to the surface, the resulting surface will be a smooth manifold, whose smoothness is dictated by the "gluing" weight function.

5. What is the space/time complexity of the approach?

- **[AS]**
  The space complexity is O(h), where h is the anticipated spacing between neighboring points. The time complexity in the worst case is O(|P|), since computing the coefficients of the polynomial g that best fits the surface from each piin P takes O(|P|), where P is the set of input points. However, they claim that the runtime of their implementation, using a hierarchical method inspired by solutions to the N-body problem, is heavily dependent on h.
- **[DS]**
  The projection computation is the most time-consuming step. O(N), where N is the number of points.

Frame rates are determined by number of visible representation points and number of pixels to be filled.

- **[FP]**
  The plane fitting phase is implemented by an iterative algorithm that requires computation of the weight and fitting error for each element of the point set, and solving a linear system to update the normal. Therefore, each iteration is O(N), and by assuming only few (constant) number of iterations, the plane fitting phase would be also O(N).
  The polynomial fitting phase is also O(N) since it also requires computation of the weight and fitting error at each point in the set, and solution to a linear system. However this is just done once.
  In practice, the authors implement an Octree to identify relevant points (i.e., the surface samples close to the point to be projected), and clusterize distant point in just some few representative points.
- **[JD]**
  The time complexity of the algorithm is O(N) because finding all of the neighbor points is the most intensive step. The authors note that they speed this up by using an N-body simulation technique where points far away are averaged together. The space complexity is intentionally not addressed because it is not an issue.
- **[LF]**
  The authors mention that the MLS projection procedure is the bottle neck of the method. Naively, the procedure would take O(N) time, but approached hierarchically (Octree; approximating clusters of far-away points), the growth rate slows to about constant time, dependent on the size of locality feature h (larger h, slower projection).
- **[MK]**
  The complexity of the approach (ignoring issues of upsampling for visualizations) is constant per projection point. (This is due, in part to the use of an octree that supports approximate evaluation of clusters of points when the point of evaluation is far away.)

6. How could the approach be generalized?

- **[AS]**
  Their approach can be generalized so that h is not a global parameter, but can be adapted to local feature sizes, and therefore account for differences in sampling rates.
- **[DS]**
  Can be extended to achieve multiresolution modeling.
  Can be integrated with combinatorial methods to achieve topological guarantees and possibility of smoothing out noise and small features.
- **[FP]**
  The algorithm proposed can be generalized to larger dimensions. This mean, it can be used to project points onto a D-1 manifold using sample points in a D dimensional space.
  Both the plane fitting and the polynomial fitting phase are very flexible and allow many adaptations. For instance, the weighting function could take many other functional forms, or even be locally adaptive. The polynomial degree or even taking a different functional basis (for instance a Fourier basis) could also be locally adapted to obtain results with different degrees of smoothness according to the region.
- **[JD]**
  The authors note that they have laid the groundwork for experimentation with higher polynomials than 3rd and 4th order. There seems to be no difficulty extending the method to higher dimensions.

- **[LF]**
  The authors mention that their approach be useful for rendering with higher order polynomials, so the method is perhaps easily extended to higher dimensions.
- **[MK]**
  As noted by Amenta and Kil, there are variations of the approach that define larger projection domains, though these can come at higher computational cost.

7. If you could ask the authors a question (e.g. "can you clarify" or "have you considered") about the work, what would it be?

- **[AS]**
  Is the quality improvement in the results obtained from performing the non-trivial Powell's minimization, i.e., minimizing $\sum_i (\langle n, p_i \rangle - D)^2 \theta(||p_i - q||)$, versus the time required to compute it significant compared to results obtained from performing minimization of an energy function whose weight term is independent of the unknowns, i.e., minimizing $\sum_i (\langle n, p_i \rangle - D)^2 \theta(||p_i - r||)$ versus the time required to perform this simpler minimization?
- **[DS]**
  Is there a case in which the random point selection for the up-sampling algorithm will produce a significantly different result in different iterations?
  How big of an impact does the algorithm have on memory?
- **[FP]**
  The results presented by the authors show their method work fine on smooth surfaces. This is expected due to the smooth weighting function (θ) and the use of high order polynomials (degree 3 and 4) in the fitting phase. How does the method perform on shapes with sharp edges and corners?
  The authors claim that the projection invariant point set forms a 2D manifold. Finding this manifold seems to be a very interesting problem. By sampling this manifold we may get a point set that is more reliable (i.e., less noise and less outliers) that the initial point set, and therefore would be more accurate, say, to define a triangular mesh. They present an approach to densely sample this manifold (by upsamling using Voronoi vertices). Did they consider the problem of adaptively sampling the manifold? Or sampling uniformly in the manifold metric?
  Finally, I am not fully convinced of the fundamental importance of fitting a reference point q (of course, this point is required for the computation model proposed by the authors, but may be a similar point projection algorithm can be obtained without requiring to set this point). What seems to be the most important (and may be the only thing you really need) is the normal direction. At the end of the day, the polynomial fitting is done by moving along the normal direction, and all the points in the normal direction will be projected to the same point in the surface. Why not defining the weights for both the normal fitting and the polinomial fitting based in the distance to the normal?
- **[JD]**
  How does your method handle boundaries and non-manifold surfaces?
- **[LF]**
  Could you perhaps suggest how h might be automatically specified for a given rendering scheme or acquisition modality?
- **[MK]**
  Why is it so essential that the mapping be a projection? Why would happen if one were to use the simpler, non-projecting, variant but apply it multiple times? (The solution of the linear system required to define a projection also requires an iterated approach, so why is one better than other.)