Interpolating and Approximating Implicit Surfaces from Polygon Soup

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 an approach to construct implicit representations of surfaces from polygonal data. The implicit surface, which may interpolate or approximate the input polygons depending on user choice, is represented using MLS with constraints integrated over the polygons. The paper also presents a method for enforcing normal constraints and an iterative procedure that ensures that the implicit surface encloses all the input vertices, converging to a circumscribing ellipsoid at the smoothest approximations.
- **[DS]**
  The paper presents a method to build interpolating or approximating implicit surfaces from polygonal data. The approximating algorithm has a user-defined parameter that specifies the minimum feature size to keep in the resulting surface. The implicit functions are obtained through a moving least squares algorithm with constraints integrated over the given polygons (not just point constraints).
  The paper also shows an improved method to enforce normal constraints and an iterative procedure that ensure the output surface tightly encloses the input vertices.
- **[FP]**
  This paper proposes a method to implicitly define a surface from a set of polygons. In contrast to point sampling, "polygon sampling" provides a more compact description of large patches of the surface (especially planar patches), and avoids undersampling artifacts of traditional point sampling. The first contribution of the authors is the extension of MLS to fit a function defined on a set of polygons: by looking at each polygon as an infinitely dense point set, computation of the weights in the MLS framework is now done in a continuous domain. The second contribution is the introduction of a signed distance function that is based in a different rationale than previous approaches: the signed distance to the surface is given by a weighted average of the distances to planar patches.
- **[JD]**
- **[LF]**
  The paper introduces a new approach for building implicit surfaces from polygon surfaces which allows control over how approximating vs. interpolatory the method should be. (User defines a threshold size. below threshold, approximating, above interpolating.)
- **[MK]**
  Like previous work, this paper proposes a new approach to surface fitting that seeks to fit an implicit function to input data by prescribing that the function evaluate to zero at the input and avoiding the trivial solution by incorporating gradient/normal constraints. However, unlike previous work, this work proposes to solve the problem when the input consists of surface patches rather than surface points.
  The authors further push on this approach to allow the definition of tight enclosing envelopes, by iteratively adjusting solving for the implicit function and adjusting the constraint values.

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

- **[AS]**
  The key insight of this paper is the representation of implicit surfaces using MLS with constraints not over points, but over entire surfaces of polygons. Their setup is neat

because only the construction of the system is difficult, since it involves integrals rather than summations, but solving the system only requires inverting a 4x4 matrix, as in the case of constraints over points.

- **[DS]**
  The implicit surface representations are obtain through a moving least squares algorithm that uses constraints integrated over the polygons. These force the resulting function to have a specific value over the surface region of a polygon, and forces the function's upward gradient to match the polygon's outward normal.

- **[FP]**
  Surface is reconstructed from an implicit function that interpolates or approximates a set of polygons. Signed distance is computed as a weighted average of distances to planar patches.
- **[JD]**
- **[LF]**
  The key insight of the paper is to perform least squares integration over not just the input samples but the entire polygon which defines those samples.
- **[MK]**
  This work makes two observations. First, that the earlier implicit methods that used points as constraints can be extended to using surface patches as constraints (replacing the discrete summation over the points in the definition of the normal equation with an integration over the patches). Second, the authors propose a generalization of previous (implicit) MLS methods by considering the case that the data given at the constraint points are not values but functions. (That is, the generalization sees earlier MLS approaches as assuming an input of a 0-th order Taylor expansion of the implicit function at the constraint points and extends this by allowing higher-order expansions.)

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

- **[AS]**
  One limitation of their method is that they only allow for the fitting of constant functions in their MLS adaptation, rather than polynomials, as in regular MLS fitting.
- **[DS]**
  Extracting small features from the surface representation requires a very fine resolution and produces models with a very large number of polygons.
- **[FP]**
  Integration of the weights on each polygon is not done in closed form. It relies on numerical methods that approximate the value of the integral. In the case of polygon set interpolation, the weight function blow up near the polygon set, so numerical integration require dense sampling to obtain accurate results.
  The weighting function contains a parameter ϵ which controls the global smoothness of the reconstruction. This parameter seems to be scale dependent and it is not clear how to set up this parameter to attain a desired degree of smoothness. Also increasing ϵ is not enough to obtain a smooth approximation of the surface. Additional corrections are required to align the reconstruction with the original data surface and to cover all the vertices.
- **[JD]**
- **[LF]**
  As noted by the authors, if a polygon surface self-intersects, the interpolating surface will

have saddle artifacts at those intersections. There are no guarantees on how well this method extrapolates (fills holes and gaps).

- **[MK]**
  As the authors observe, the choice of RBF kernel results in an integrand that cannot be evaluated in closed form over a triangle. As a result, a partial-quadrature approach is used, computing the along appropriately space 1D line segments in the triangle and then summing.

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

- **[AS]**
  The output surface is guaranteed to be watertight.
- **[DS]**
  The algorithm will produce interpolating surfaces that will fill gaps and holes to make the surface tight, or approximating surfaces that will smooth out features below a specified size.
- **[FP]**
  Since the approach is based in an implicit description of the surface, it guarantees watertight results. For the approximation case (i.e, ϵ>0), the weighting function is infinitely smooth and continuous, so the implicitly defined surface should also have this properties.
  Since the method input is an arbitrary polygon set which may contains intersecting polygons, the final reconstruction may be non-manifold.
- **[JD]**
- **[LF]**
  The output surface is water-tight and manifold.
- **[MK]**
  As with all implicit based reconstructions, the output is a manifold without boundary, so long as the gradient is non-vanishing at the zero-level set.

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

- **[AS]**
  The space and time complexity of this approach are similar to the space and time complexity of the regular MLS approach, since the only major difference between the two methods is in the setup, which in this case requires the solution of integrals. Therefore, the time complexity for this approach is $O(N)$ for each iteration, and the space complexity is also $O(N)$.
- **[DS]**
  Not clearly specified, but depends on whether it's an approximating or interpolating scheme, and also on the size of the features to keep in the representation.
- **[FP]**
  Implicit function evaluation at certain point x requires, for each polygon, computing the distance to x and evaluating the respective weights integral $\int(|p-x|)dp$. Since integrals are computed numerically, it would be very expensive evaluate the integrals for all polygons. Instead, the authors use a KD tree structure to identify close polygons where weights integral are computed numerically and for the further polygons the weight used is an approximated average weight.
- **[JD]**
- **[LF]**
  The growth rate of the algorithm in time/space is not stated in the paper, but

performance results on various dataset and parameter combinations are provided in Table 1.

- **[MK]**
  I believe that the method is linear (assuming that nearest neighbors can be computed in constant time).

6. How could the approach be generalized?

- **[AS]**
  This approach pushes the complexity to the points in the point cloud, which can have functions of arbitrary degree associated to them, but the fitting function is restricted to a constant function. Therefore, this approach can be generalized by allowing the fitting function to also be a polynomial of arbitrary degree.
- **[DS]**
  There needs to be a better way to obtain a polygon mesh from the surface representation that is efficient for obtain small features.
- **[FP]**
  A first generalization would be the use of locally adjusted epsilon. The authors use a global epsilon parameter to control the smoothness of the implicitly defined surface. However, the implicit function depends more in the local structure of the polygon set (i.e., in the polygons near to the evaluation point) than in the global structure. Therefore, it is reasonable that by locally adjusting the epsilon parameter we will obtain reconstructions with adaptive smoothness.
  A second generalization would be a mixed technique to define the weights and implicit functions using both point samples and polygons. In the current approach, isolated point samples have zero weight since they are zero measure elements, however they could be adjusted to have a positive weight. We could also fit a surface around a sample point p (a plane or polynomial surface), and use this base surface to determine the distance prediction given by this sample point p for any 3D point x.
- **[JD]**
- **[LF]**
  The feature parameter, epsilon, can be adjusted to get results of differing levels of smoothness (enclosing meshes of fewer points), which are useful for a wide range of applications.
- **[MK]**
  One could consider filters which can be integrated in closed form over triangles.
  It should also be possible to extend this work to non-planar constraint patches.

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]**
  Have you tried formulating your MLS adaptation using fitting functions of higher degree than a constant function?
- **[DS]**
  How do other weight function choices affect the results?
  In the animation discussion, once you get an envelope for the deformable object, how would you translate that into an animation of the actual object?
- **[FP]**
  The authors present two approaches to implicitly define a signed distance to a surface: use surface samples to fit a function and evaluate it at your point (i.e., traditional MLS),

or, use surface samples to predict independent distances and take a weighted average. How can these approaches be unified?

- **[JD]**
- **[LF]**
  Did you consider other weighting functions, other ways of overcoming the near-zero singularity issue?
- **[MK]**
  The authors assert that there method generalizes IMLS by allowing for the assumption of higher-order function approximations at the input points. That is, in the formulation of the MLS procedure, it pushes the evaluation of the fit-functions from the constraint points to the evaluation point:
  $$\arg\min_{f_p\in F}\sum_{q\in Q} w(|p-q|)\left|f_p(q)-f_q\right|^2 \Rightarrow \arg\min_{f_p\in F}\sum_{q\in Q} w(|p-q|)\left|f_p(p)-f_q(p)\right|^2$$
  Where Q is the set of constraint points, $f_q$are the prescribed values/functions at the constraint points, and F is the space of functions that are for the local fitting.
  The problem with this formulation is that while the previous formulations of MLS essentially assume that the f_q are constant functions, while allowing the space F to be arbitrary, here the assumption is that the f_q are arbitrary functions while the space F is assumed to be the space of constant functions. (For example, if one were to try to fit a higher-order polynomial using the new formulation, the resulting matrix giving the coefficients would necessarily be singular, since the evaluation occurs at the point p and hence does not see anything aside from the constant term of f_p.)
  The only way we could understand to actually consider a different generalization:
  $$\arg\min_{f_p\in F}\sum_{q\in Q} w_1(|p-q|)\cdot\int w_2(|p-r|)\cdot\left|f_p(r)-f_q(r)\right|^2 dr$$

  This way, the fit compares the evaluation of the f_q in the vicinity of the point p, thereby taking into account the higher terms of f_p. Though, naturally, this formulation is somewhat hairier. (In the context of this formulation, the proposed approach uses $w_1 = w$ and $w_2 = \delta$.)