Optimization Model for Polyline Offset

Zitao Xu

Abstract

Classical offset algorithms focus on constructing offset curves and surfaces. Optimization-based geometry processing has also been extensively studied for mesh fairing, deformation, and parameterization.

The present work differs from both directions by introducing a least-squares formulation whose unknowns are tangential displacements of offset vertices, yielding a globally optimized offset configuration.

Keywords: optimization model, least square, rectangle quad

1. Introduction

Quality polyline offset is required in many applications, such as quality quad mesh generation, in which the offset quads need to be well and evenly shaped, or less distorted. Accordingly, various approaches have been presented in the past decades for polyline offset and quad mesh generation, which are quite related.

Previous work on offset curves and surfaces has focused on geometric construction issues such as exact and approximate offsets, continuity preservation, singularity handling, self-intersection detection, and trimming [4,5]. More broadly, optimization-based geometry processing has been widely applied to mesh fairing, deformation, parameterization, interpolation, and discrete geometric modeling [6,7,8,9].

The present work is inspired by the optimization philosophy of these methods, but addresses a different problem. Instead of optimizing vertex positions directly, the proposed formulation introduces tangential displacement variables for offset vertices and determines them through a least-squares optimization model. The resulting sparse linear system computes a global minimizer of the proposed objective, producing offset configurations with improved quadrilateral regularity and reduced shearing.

The method is straightforward to implement and computationally efficient. The resulting linear system has a tridiagonal structure, allowing it to be solved in O(n) time using standard tridiagonal solvers. Although this article focuses on polyline offsets in order to present the underlying idea clearly, the same formulation has already been extended and applied successfully to quadrilateral mesh surface offsets, demonstrating that the approach is not restricted to two-dimensional geometry.

The underlying optimization framework appears sufficiently general to support other geometric optimization applications. Details of the quadrilateral surface-offset formulation and related extensions will be presented in future work.

This article focuses on polyline offsets because they provide the simplest setting for presenting the underlying formulation and its essential properties.

To intuitively understand this optimization model, let's begin with a simple and direct offsetting of each vertex of a polyline along its discrete normal direction, disregarding quality and distortion, see Fig. 1 (a).

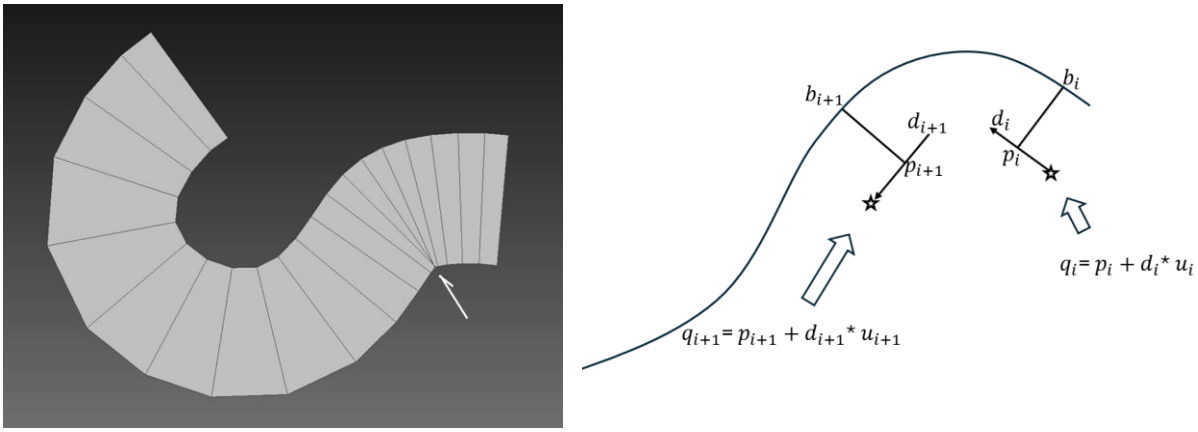

Fig. 1, (a): distorted rectangles with crimping; (b): idea to avoid crimping

A key observation is that crimping can be reduced by allowing offset vertices to move along their tangent directions. As illustrated in Fig. 1(b), tangential displacement provides additional degrees of freedom that can be optimized to improve quadrilateral regularity, i.e. by moving crimped segment p_i, p_{i+1} apart along their tangent d_i, d_{i+1} to become q_i, q_{i+1}.

If the goal is to make overall offset quads in Fig. 1 (a) "as rectangle as possible", it includes a necessary condition that offset segment p_i, p_{i+1} has same or "as same as possible" length as base segment b_i, b_{i+1} length. Therefore, let's start to build a model with this condition, then fine-tune the model later.

2. Optimize Edge Length

To achieve above goal, it can build an error measurement function to sum all differences, each of which is the difference between base segment $(b_{i+1} - b_i)$ length and the expected offset segment $(q_{i+1} - q_i)$ length in Fig 1 (b), and then minimize this function to find a global minimizer of the proposed objective.

Vertex p_i is extended from base point b_i in the perpendicular direction of b_{i-1}, b_{i+1}. This is a reasonable way to locate p_i as known. Same for p_{i+1}.

Assume that the optimal position is obtained by moving p_i alone it's tangent vector d_i, which can be defined parallel to its bottom b_{i-1}, b_{i+1}. See Fig. 1 (b) again. The length difference between base and offset segments $|(q_{i+1} - q_i)| - |(b_{i+1} - b_i)|$ can be hard to optimize, but this can be approximated by $|(q_{i+1} - q_i) - (b_{i+1} - b_i)|$, when base and offset segments are almost parallel, which practically often is true, then this optimization is more feasible.

Define an optimization function, or error measurement function, as sum of square of each item:

$$F_l(u) = \sum_{i=1}^{n-1}((q_{i+1} - q_i) - (b_{i+1} - b_i))^2$$

$$= \sum_{i=1}^{n-1}(C - d_{i+1}u_{i+1} + d_i u_i)^2$$

2.1.1

where u is vector $(u_1, \ldots, u_i, \ldots u_n)$, the moving amount at each tangent $(d_1, \ldots, d_i, \ldots d_n)$, and the known constant: $C = (p_{i+1} - p_i) - (b_{i+1} - b_i)$. Let $\frac{\partial}{\partial u} F_l = 0$, we get the following:

$$\frac{\partial}{\partial u_i} F_l(u) =$$

$$\sum_{i=1}^{n-1} 2d_i^2 u_i + 2Cd_i - 2d_{i+1}d_i u_{i+1} = 0$$

2.1.2

$$\frac{\partial}{\partial u_{i+1}} F_l(u) =$$

$$\sum_{i=1}^{n-1} 2d_{i+1}^2 u_{i+1} - 2Cd_{i+1} - 2d_{i+1}d_i u_i = 0$$

2.1.3

Fortunately, above is a linear system: $Bu = W$, where matrix B are those items with u_i, u_{i+1}, and vector W are those items with constant C in 2.1.2 and 2.1.3

Solving this linear system gets the minimum of 2.1.1, minimizing the sum of differences between base and offset segment lengths to achieve a global minimizer of the proposed objective.

But this may shear. Some result is shown in Fig 3 (a), which faithfully does what equation 2.1.1 minimization tries to achieve. However, the same base and offset segment length is a necessary but not a sufficient condition, which does not guarantee to align vertically!

3. Optimize Shearing

To solve the shearing issue, an additional term is introduced to "hook" the middle point (star in Fig. 2) of an offset edge unchanged, in order to only let the offset edge length change. Define:

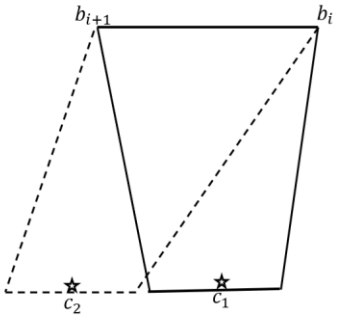

Fig. 2 offset quad shearing

$$F_m(u) = \sum_{i=1}^{n-1}((p_{i+1} + d_{i+1}u_{i+1} + p_i + d_i u_i)/2 \; - \; (p_{i+1} + p_i)/2\,)^2$$
$$= \sum_{i=1}^{n-1}(d_{i+1}u_{i+1} + d_i u_i)^2/4$$

2.2.1

$F_m(u)$ is to measure shearing magnitude, the distance between c_1 and c_2 in Fig 2. Get its derivatives:

$$\frac{\partial}{\partial u_i}F_m(u) = \sum_{i=1}^{n-1}(d_i^2 u_i + d_{i+1}d_i u_{i+1})/2$$

2.2.2

$$\frac{\partial}{\partial u_{i+1}}F_m(u) = \sum_{i=1}^{n-1}(d_{i+1}^2 u_{i+1} + d_{i+1}d_i u_i)/2$$

2.2.3

Modify optimization function from $F = F_l$ to $F = F_l + F_m$, and get F' below:

$$\frac{\partial}{\partial u_i}F = \frac{\partial}{\partial u_i}F_l + \frac{\partial}{\partial u_i}F_m = \sum_{i=1}^{n-1} 2.5d_i^2 u_i \; - \; 1.5d_{i+1}d_i u_{i+1} + 2Cd_i$$

2.3.1

$$\frac{\partial}{\partial u_{i+1}}F = \frac{\partial}{\partial u_{i+1}}F_l + \frac{\partial}{\partial u_{i+1}}F_m = \sum_{i=1}^{n-1} 2.5d_{i+1}^2 u_{i+1} - 1.5d_{i+1}d_i u_i - 2Cd_{i+1}$$ 2.3.2

Let $F' = 0$. Fortunately, this is also a linear system:

$$Bu = W$$ 2.3.3

where matrix B are those items with $u_i, u_{i+1},$ and vector W are those items with constant C in 2.3.1 and 2.3.2. Solving this linear system gets $(u_1 \ldots, u_i, \ldots u_n)$, which yields much better result, see Fig 3. (b) improved from Fig. 3 (a).

The coefficient matrix B is tridiagonal because each variable $u_i$interacts only with its two neighboring variables $u_{i-1}$and u_{i+1}. Therefore, the system can be solved in O(n) time using standard tridiagonal solvers.

The actual C++ code for 2.3.1 and 2.3.2 is below:

```
for (int i=0; i<(n-1); i++) {
    int j=i+1;
    d1 = d[j]; d0=d[i]; C = (b[j]-b[i]) – (p[j]-p[i]);
    B[i][i]  += 2.5 * d0 * d0;
    B[i][j]  -= 1.5 * d0 * d1;
    W[i]    -= 2.0 * C * d0;
    B[j][i]  -= 1.5 * d0 * d1;
    B[j][j]  += 2.5 * d1 * d1
    W[j]    += 2.0 * C * d1;
}
```

where: *B* is left side *(n x n)* matrix, *W* is right side vector of *n* in 2.3.3, *p[i]* and *b[i]* are bottom and top vertex arrays, see Fig.1 (b), and *d[i]* is tangent array there. These 6 assignments are 1-to-1 match to 6 right-side terms in 2.3.1 and 2.3.2, and note, signs to *W* are reversed. The operator * functions dot product of (*x,y*) points. Solving u in 2.3.3 completes the whole job, which is the *u[i]* array in Fig.1 (b).

The result achieved what is expected in shearing optimization: hooking the middle point of the offset edge unchanged as much as possible.

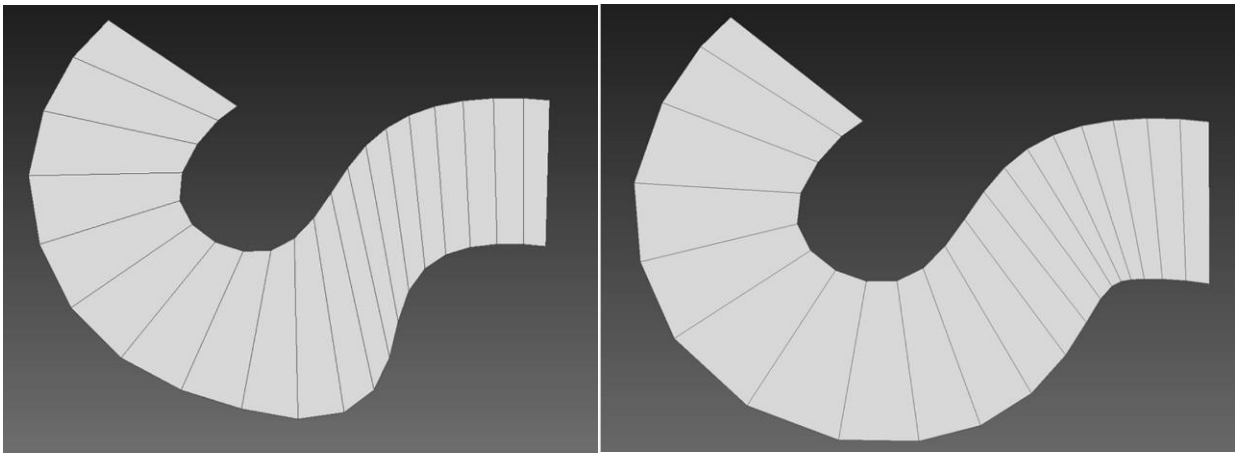

Fig. 3. (a) $F = F_l$ with shearing; (b) $F = F_l + F_m$ shearing improved.

Note, this optimization process can be applied multiple times: that is to start from adjusted point, i.e. simply let: $P_i = Q_i$, see Fig. 1 (b), then repeat the process.

For each quadrilateral, a simple distortion measure is defined as the ratio between the lengths of the corresponding base edge and offset edge. The smaller of the two lengths is used as the denominator, so the ratio is always greater than or equal to one for a valid quadrilateral. A value of 1.0 indicates equal edge lengths, while larger values indicate increasing geometric distortion. Negative values indicate inverted quadrilaterals in which the two side edges cross each other. The table, Fig. 4 (a), shows 6 worst quads in Fig.1 (a), vs Fig 3 (b). The first column is initial state; the second column is the result after applying proposed processing once; the third column applying twice; the last column is improvement factor.

Initial	Once	Twice	Improvement Factor
7.362	4.969	2.953	2.494 times
-44.384	9.361	2.786	15.931 times
-52.389	5.123	1.745	30.002 times
3.436	1.994	1.152	2.982 times
3.791	3.392	2.820	1.117 times
3.550	3.135	2.551	1.391 times

Matrix	Initial	Twice
Worst quad	52.389	1.745
Mean of six worst	19.152	2.335

Fig. 4, (a) six worst quads optimization ratios, and improvement factors; (b) max and mean ratios

The optimization significantly reduces geometric distortion. In the most severe cases, inverted quadrilaterals are corrected after one optimization, and the distortion measure improves by more than a factor of thirty after two optimization passes. Because the optimization is performed on the updated geometry at each iteration, repeated application progressively refines the configuration toward lower distortion.

4. Conclusion

This article presents a least-squares optimization formulation for improving polyline offset quality through tangential displacement of offset vertices. The resulting sparse linear system computes the

global minimizer of the proposed quadratic objective and produces offset configurations with improved quadrilateral regularity and reduced shearing.

The formulation is straightforward to implement and computationally efficient. Although demonstrated here on polyline offsets, the same framework has been successfully extended to quadrilateral mesh surface offsets, suggesting that the underlying optimization strategy may be applicable to a broader class of geometric optimization problems. Future work will investigate these extensions in greater detail.

References

[1] Hoschek, Josef and Lasser, Dieter, Fundamentals of Computer Aided Geometric Design, AK Peters, Ltd, 1993
[2] Piegl, Les and Tiller, Wayne, The NURBS Book, Springer, 1995
[3] Gerald Farin, Curves and Surfaces for Computer-Aided Geometric Design, 1993
[4] Helmut Pottmann, Offset Curves and Surfaces: A Brief Survey, International Journal of Computational Geometry & Applications, 1995.
[5] Elber, G., Lee, I.-K., Kim, M.-S., "An Overview of Offset Curves and Surfaces", Computer-Aided Design, Vol 31, No3, pp.165-173, 1999
[6] Botsch et al., Polygon Mesh Processing, AK Peters, 2010.
[7] Olga Sorkine and Marc Alexa, As-Rigid-As-Possible Surface Modeling, Eurographics, 2007.
[8] Michael S. Floater, Mean Value Coordinates, Computer Aided Geometric Design, 2003.
[9] Bergou, et al., Discrete Quadratic Curvature Energies, Siggraph 2006 Course Notes.