A Real-Time Screened-Poisson Solver for Interactive Surface Editing

Ming Chuang[*] Michael Kazhdan[†]
Johns Hopkins University

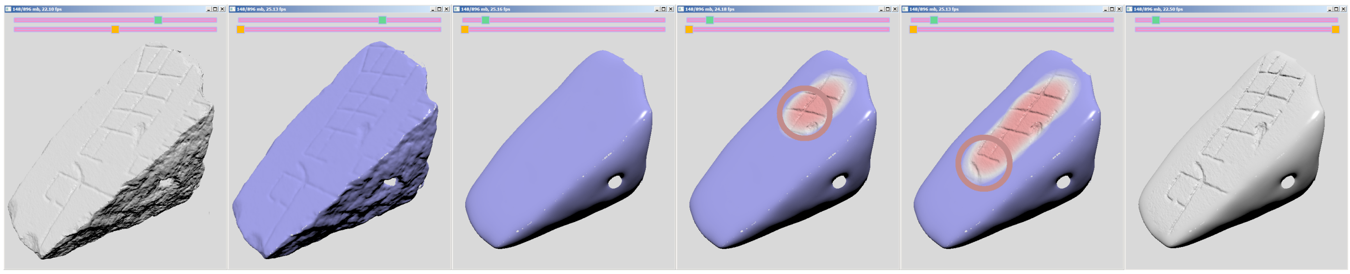

Figure 1: *Example system use: Starting with the original model (top left), a user applies global smoothing to the geometry by dragging the gradient scale slider (top center). To have the smoothing effect the lower geometric frequencies, the user drags the fidelity slider (top right). The user then specifies that the top face of the tablet should be sharpened by selecting the "spray-can" interface with a gradient amplification setting and spraying the sharpening constraint across the surface to obtain the final model (bottom row).*

Abstract

We present a novel framework for editing geometric detail. Drawing on previous work in image- and geometry-processing, we show that the frequency-space filtering performed in mesh smoothing and sharpening can be formulated as the solution to a Poisson-like system of equations. Using this observation, we design an interactive system that supports both global and local modulation of surface detail, expressing the position of vertices on the edited geometry as the solution to a large linear system of equations.

1 Introduction

We describe a real-time system for editing large 3D meshes. The system supports both global and local modulation of surface detail, expressing the position of vertices on the edited geometry as the solution to a linear system of equations defined over the surface. The interactivity of our system is enabled by the design of an efficient sparse linear solver, providing an interface with which users can explore a broad landscape of possible surface modifications.

The design of our system is motivated by spectral approaches in geometry-processing that draw on the parallels between the eigenvectors of the mesh Laplacian and the Fourier basis [Taubin 1995], and recent work in image-processing that relates Fourier-space filtering of images to the solution of a Poisson-like equation [Bhat et al. 2008]. Given a (coordinate) function F, the goal is to find the best-fit (new coordinate) function G such that the following objective function can be minimized:

$$\alpha \|G-F\|^2 + \|\nabla G - \beta \nabla F\|^2$$

where α is a non-negative weight that represents the fiedelity to the original function F, and β is a gradient scaling term. Applying the Euler-Lagrange equation, the minimizer is in fact the solution to a screened-Poisson equation:

$$(\alpha I - \Delta)G = \alpha F - \beta \Delta F.$$

Such formulation can not only trade off between sharpening and smoothing by controlling the value of the gradient scale β, but also change the level-of detail at which the gradient modulation come into effect by controlling the fidelity weights α. Another important benefit is the *spatially-varying* edits can now be achievable by setting β as a digonal matrix that accepts locally adaptive weights.

[*]e-mail: ming@cs.jhu.edu
[†]e-mail: misha@cs.jhu.edu

2 Implementation

The feasibility of turning the idea into a real-time application relies on how fast we can solve the propsed linear system. To this end, we leverage the regularity of the octree of the finite-elements system introduced in [Chuang et al. 2009], and implement a *parallelized* multigrid solver that can relax our screened-Poisson system at interactive frame-rates.

In particular, we decompose the domain into barely overlapped regions and solve them parallelly while carefully maintaining the correctness of the Gauss-Seidel relaxation. Our implementation also achieves a cache-friendly memory access pattern via the technique of the temporal blocking that carefully reuses cache data to its limit without hampering the integrity of the solution.

To evaluate the new coordinate function, as required in the forementioned finite-elements system, we exploit the horsepower of parallelation in modern programable video cards. Instead of streaming new vertex coordinates from CPU to GPU at each frame, we send only the element coefficients and perform the evaluation on the GPU side, then release the memory from CUDA to OpenGL for the visualization purpose. Such arrangement, along with the predescribed fast linear system solver, grants us the ability of processing meshes consisting of up to one million vertices at a rate of over 25 frames-per-second.

References

BHAT, P., CURLESS, B., COHEN, M., AND ZITNICK, L. 2008. Fourier analysis of the 2D screened Poisson equation for gradient domain problems. In *Proceedings of the 10th European Conference on Computer Vision*, 114–128.

CHUANG, M., LUO, L., BROWN, B., RUSINKIEWICZ, S., AND KAZHDAN, M. 2009. Estimating the Laplace-Beltrami operator by restricting 3D functions. *Computer Graphics Forum (Symposium on Geometry Processing)*, 1475–1484.

TAUBIN, G. 1995. A signal processing approach to fair surface design. In *ACM SIGGRAPH Conference Proceedings*, 351–358.