A Closed-Form Formulation of HRBF-Based Surface Reconstruction by Approximate Solution

Charlie C.L. Wang

Delft University of Technology

June 21, 2016

This is a joint work with

Shengjun Liu, Central South University, China.
Charlie C.L. Wang, Delft University of Technology, The Netherlands
Guido Brunnett, Chemnitz University of Technology, Germany
Jun Wang, Nanjing University of Aeronautics and Astronautics, China

Introduction

Fast surface reconstruction from a massive number of samples is important to many applications – robotics & CAD/CAM

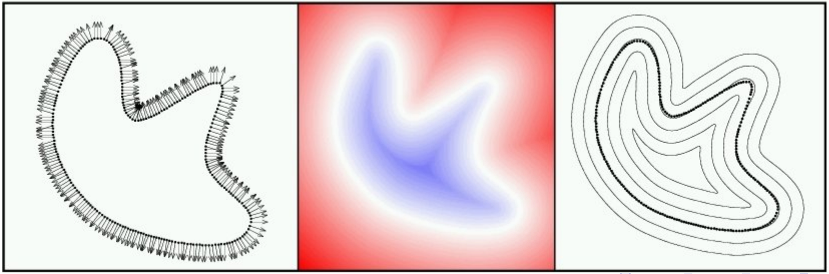
- Space exploration and path planning
- On-site inspection and compensation

Input scenario with $922k$ points

Literature Review

Fitting implicit functions to build scaler fields and extracting isosurfaces from fields as the result of surface reconstruction

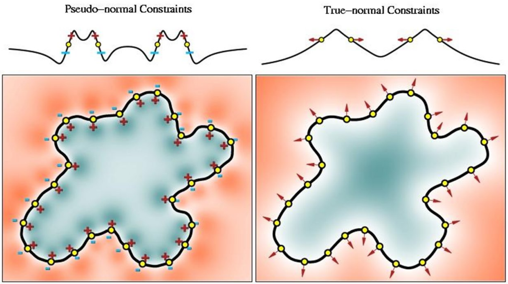
- *Radial Basis Function* (RBF) [Carr et al., 2001]
- *Multiple Partition-of-Unity* (MPU) [Ohtake et al., 2003]
- *Smooth Signed Distance* (SSD) [Calakli and Taubin, 2011]
- *Poisson* reconstruction [Kazhdan and Hoppe, 2013]



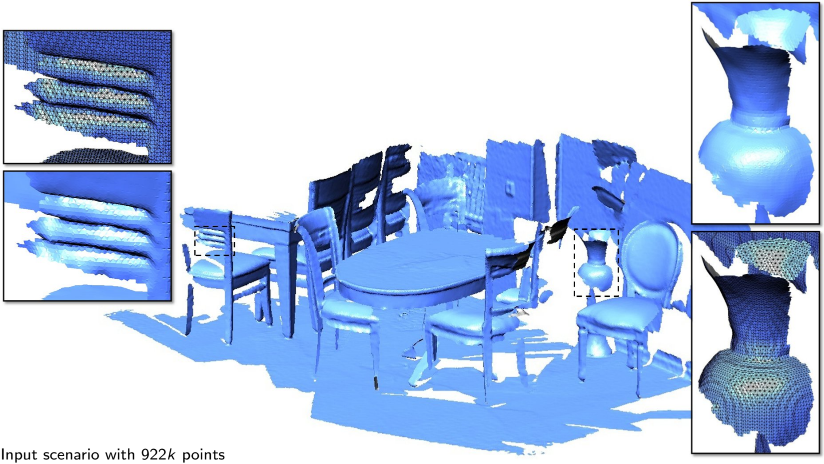
Challenges of Surface Reconstruction in Real-Time

$f(\mathbf{x}) = 0$ with different signs at different sides of the surface to be reconstructed

- Quality: indirect vs. direct enforcement on normals
- Efficiency: solving large linear systems – unstable and time-consuming



We propose a closed-form formulation to reconstruct surfaces by using *Hermite Radial Basis Functions* (HRBFs).



Input scenario with 922k points

Reconstruction on CPU within 5.5 sec. resulting in 313k triangles – 17.9× faster than the state-of-the-art *Float Scaling Surface Reconstruction* (FSSR)

HRBF Implicits

Definition

Given a set of data $\mathcal{P} = \{\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_n\}$ with unit normals $\mathcal{N} = \{\mathbf{n}_1, \mathbf{n}_2, \cdots, \mathbf{n}_n\}$, the HRBF implicits give a function f interpolating both the points and the normal vectors as

$$f(\mathbf{x}) = \sum_{j=1}^{n} \{a_j \varphi(\mathbf{x} - \mathbf{p}_j) - \langle \mathbf{b}_j, \nabla\varphi(\mathbf{x} - \mathbf{p}_j)\rangle\}, \tag{1}$$

where $\varphi : \Re^3 \mapsto \Re$ is defined by a radial basis function $\varphi(\mathbf{x}) = \phi_\rho(\|\mathbf{x}\|)$, $\langle \cdot, \cdot \rangle$ denotes the dot-product of two vectors, and ∇ is the gradient operator.

Unknown to be determined: the scalar coefficients, $a_j \in \Re$, and the vector coefficients, $\mathbf{b}_j \in \Re^3$

Kernel Function

We use a Wendland's *Compactly Supported Radial Basis Functions* (CSRBF) as the kernel function

$$\phi_\rho(r) = \phi(r/\rho)$$
$$\phi(t) = \begin{cases} (1-t)^4(4t+1), & t \in [0,1], \\ 0, & \text{otherwise}, \end{cases} \tag{2}$$

where ρ is the support size, and r is the Euclidean distance between a query point and the center of a kernel function.

Constraints of Interpolation

$$f(\mathbf{p}_i) = c \ \text{ and } \ \nabla f(\mathbf{p}_i) = \mathbf{n}_i, \ \ (i = 1, 2, \cdots, n) \tag{3}$$

This leads to a linear system

$$\begin{array}{l} \sum_{j=1}^{n} \{a_j \varphi(\mathbf{p}_i - \mathbf{p}_j) - \langle \mathbf{b}_j, \nabla\varphi(\mathbf{p}_i - \mathbf{p}_j)\rangle\} = c, \\ \sum_{j=1}^{n} \{a_j \nabla\varphi(\mathbf{p}_i - \mathbf{p}_j) - \mathbf{b}_j \mathbf{H}\varphi(\mathbf{p}_i - \mathbf{p}_j)\} = \mathbf{n}_i, \end{array} \quad (4)$$

where $i = 1, 2, \cdots, n$ and $\mathbf{H}$ is the Hessian applied on $\varphi(\cdot)$. That is

$$\mathbf{A}\boldsymbol{\lambda} = \mathbf{y}, \quad (5)$$

where $\boldsymbol{\lambda}$ and $\mathbf{y}$ are $4n$ vectors with the i-th blocks being $[a_i, \mathbf{b}_i]^T$ and $[c, \mathbf{n}_i]^T$ respectively. Each block $\mathbf{A}_{i,j}$ is a 4×4 sub-matrix corresponding to a pair of RBF centers $(\mathbf{p}_i, \mathbf{p}_j)$.

$$\begin{array}{l} \mathbf{A} = (\mathbf{A}_{i,j})_{n \times n}, \\ \mathbf{A}_{i,j} = \begin{pmatrix} \varphi(\mathbf{p}_i - \mathbf{p}_j) & -(\nabla\varphi(\mathbf{p}_i - \mathbf{p}_j))^T \\ \nabla\varphi(\mathbf{p}_i - \mathbf{p}_j) & -\mathbf{H}\varphi(\mathbf{p}_i - \mathbf{p}_j) \end{pmatrix}_{4 \times 4}. \end{array} \quad (6)$$

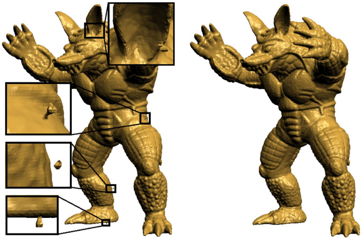
HRBF Implicits with Regularization

Regularization: Interpolation $\Rightarrow$ Approximation

A regularization term with coefficient η is added as

$$(\mathbf{A} + \eta \mathbf{I})\boldsymbol{\lambda} = \mathbf{y} \tag{7}$$

to make system better conditioned in numerical computation.



Without vs. With regularization in HRBF Interpolation

- Dimension: $4n \times 4n$
- Time-consuming
- High memory cost
- Progressive???
- Real-time computing???

Quasi-solution of Interpolation

Quasi-interpolation

Considering an exact interpolant

$$g(\mathbf{x}) = \sum_i \lambda_i \psi_i(\mathbf{x})$$

with the constraints $g(\mathbf{x}_i) = f_i$ of function values, the function $g(\mathbf{x})$ can be well approximated by letting $\lambda_i \equiv f_i$

$$\tilde{g}(\mathbf{x}) = \sum_i f_i \psi_i(\mathbf{x})$$

Recall our interpolation constraints including

- The value of function: $f(\mathbf{p}_i) = c$;
- The gradient of function: $\nabla f(\mathbf{p}_i) = \mathbf{n}_i$.

Quasi-interpolation is hard to be applied here directly.

Quasi-solution by Matrix Computation

However, quasi-interpolant with $\lambda_i \equiv f_i$ can be considered as letting the coefficient matrix approximated by $\mathbf{I}$.

HRBF Approximation

For a CSRBF $\varphi_i(\cdots)$, when there is no other center falling into the space spanned by its support ρ_i, the coefficient matrix is degenerated from $\mathbf{A}_{i,i}$ of Eq.(6) into

$$\mathbf{D}_{i,i} = diag(1, \frac{20}{\rho_i^2}, \frac{20}{\rho_i^2}, \frac{20}{\rho_i^2}) + \eta \mathbf{I}_4, \quad \mathbf{D}_{i,j} = 0 \ (i \neq j). \tag{8}$$

Thus, in the scenario of this happens at all CSRBF kernels

$$(\mathbf{A} + \eta \mathbf{I})\boldsymbol{\lambda} = \mathbf{y} \quad \Rightarrow \quad \mathbf{D}\tilde{\boldsymbol{\lambda}} = \mathbf{y} \tag{9}$$

Closed-form of HRBF with Regularization

HRBF Implicit in Closed-Form

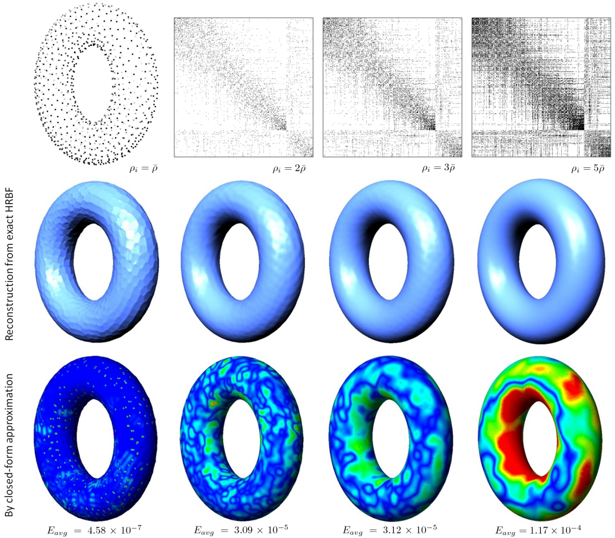
Using the fact that the zero level-set is employed in surface reconstruction (i.e., $c = 0$), an approximation function of $f(\mathbf{x})$ becomes

$$\tilde{f}(\mathbf{x}) = -\sum_{j=1}^{n} \langle \frac{\rho_j^2}{20 + \eta \rho_j^2} \mathbf{n}_j, \nabla \varphi(\mathbf{x} - \mathbf{p}_j) \rangle. \tag{10}$$

$\mathbf{D}\tilde{\boldsymbol{\lambda}} = \mathbf{y}$ leads to an approximate solution of $(\mathbf{A} + \eta \mathbf{I})\boldsymbol{\lambda} = \mathbf{y}$ with

$$\tilde{\boldsymbol{\lambda}} = \mathbf{D}^{-1}\mathbf{y} = \{\frac{c}{1+\eta}, \frac{\rho_1^2 \mathbf{n}_1}{20 + \eta \rho_1^2}, \cdots, \frac{c}{1+\eta}, \frac{\rho_n^2 \mathbf{n}_n}{20 + \eta \rho_n^2}\}.$$

The correctness relies on the error $\|\Delta\lambda\|_\infty$ with $\Delta\lambda = \boldsymbol{\lambda} - \tilde{\boldsymbol{\lambda}}$.



Study on the errors between the coefficient matrix **A** of HRBF implicit and its degenerate diagonal matrix **D**. Black dots present the elements with error greater than 10^{-3}.

Error-bound Analysis

Lemma

When Wendland's CSRBFs are used, if 1) their support sizes satisfy $\rho_{\max} < \sqrt{20}$, 2) each support region contains at most m centers of other CSRBFs, and 3)

$$\rho_{\min} > \frac{5m + \sqrt{25m^2 + 2240(1+\eta)}}{8(1+\eta)} \tag{11}$$

the error of $\Delta\lambda$, $\|\Delta\lambda\|_\infty$, is bounded by a constant.

The requirements on:

- the values of $\rho_{\min}$, $\rho_{\max}$ and η
- each support contains at most m centers of other CSRBFs

can be achieved by the parameter tuning algorithm.

Algorithm of Reconstruction I

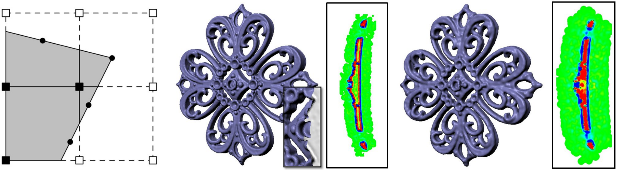
Parameter Tuning

- Determine a common temporary support size according to point density
- Select m as the maximal number of data points covered by each of these temporary supports
- Incrementally enlarge ρ_j of each CSRBF until a) will cover more than m other centers or b) will make $\max\{\rho_j\} \geq \sqrt{20}$
- Among all support sizes, the minimal is selected to check if the condition for error-bound is satisfied.
- When it is not satisfied, go back step 3) with $m = m - 1$

Algorithm of Reconstruction II

Efficient Isosurface Extraction

- Isosurface, $\tilde{f}(\mathbf{x}) \equiv 0$, can be extracted locally by limited number of kernels
- Voxels with a fixed width w, only constructed when it intersects the isosurface
- MC (or DC) can be applied only on valid voxels



By the nice property of locality, progressive reconstruction becomes possible.

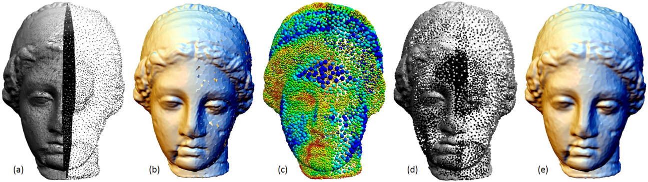
Algorithm of Reconstruction III

Adaptive Center Selection

- Applied when high non-uniformity is observed
- Adaptively select samples from input points to form a subset of centers by minimizing the degree-of-coverage:

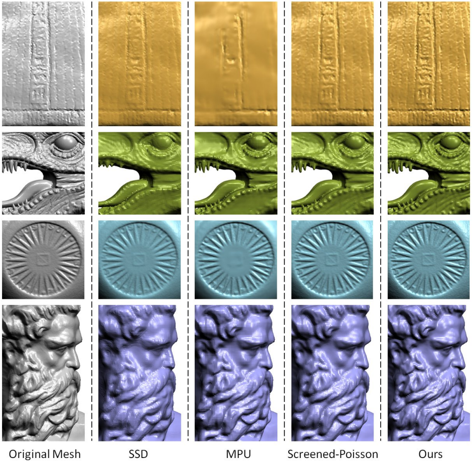
$$g(\mathbf{x}) = \sum_{k=1}^{l} \phi_{r_k}(\|\mathbf{x} - \mathbf{c}_k\|)$$

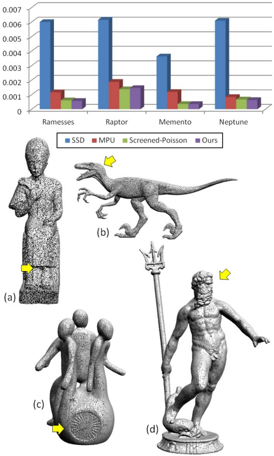
- Centers of kernels are decoupled from data points.



13, 446 centers are selected from 100, 371 highly non-uniform points

Results – Comparison on Clean Data





Accuracy similar to Screened-Poisson can be observed

Comparison for Computing Time

Tested on PC with two Intel Core i7-2600K CPUs at 3.4GHz plus 16GB RAM.

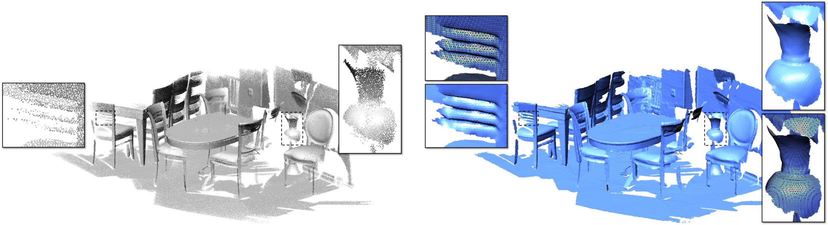
- All models are re-scaled into a bounding-box of $[-1, 1]^3 \in \Re^3$
- Reconstruction on a variety of models up to $14M$ points (in 78.9 sec.)

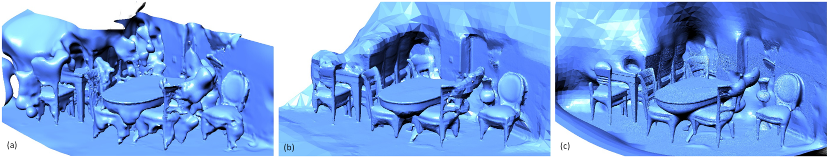
Model	Pts.	Time in Seconds*			
		SSD	MPU	Poisson	Ours
Ramesses	0.58M	14,314 ($\times 1, 724.6$)	61.2 ($\times$7.4)	40.8 ($\times$4.9)	8.3
Raptor	1.00M	1,799 ($\times$264.6)	47.2 ($\times$6.9)	31.6 ($\times$4.6)	6.8
Memento	2.52M	24,195 ($\times 1, 186.0$)	138.8 ($\times$6.8)	92.6 ($\times$4.5)	20.4
Neptune	4.98M	6,772 ($\times$358.3)	139.4 ($\times$7.4)	114.0 ($\times$6.0)	18.9

* Note that, the time reported here includes both the surface reconstruction and the mesh extraction.

† To have a fair comparison, similar number of triangles are generated for different approaches.

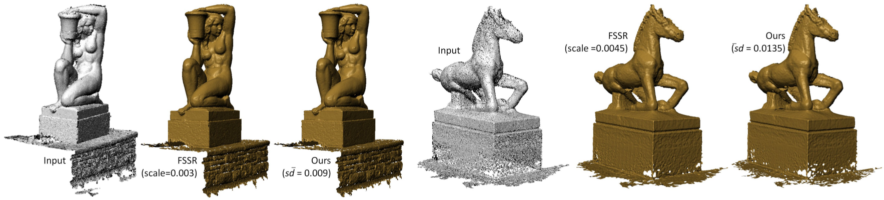
Unfair Comparison on Raw Data





Bottom, from left to right, MPU, SSD and Screened-Poisson, which are designed for reconstructing closed surfaces.

On Raw Data I

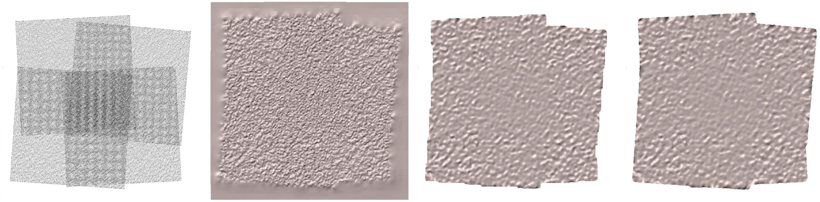


Model	Num. of Points	FSSR Num. of Triangles	FSSR Time in Seconds One-core	FSSR Time in Seconds 8-cores	Ours Num. of Triangles	Ours Time in Seconds One-core	Ours Time in Seconds 8-cores
Indoor	922.0k	319k	470.6	98.4	313k	17.1 ($\times$16.0)	5.5 ($\times$17.9)
Aquarius	253.9k	350k	407.3	89.6	375k	8.0 ($\times$7.8)	2.7 ($\times$33.2)
Horse	239.8k	241k	262.3	56.2	245k	5.4 ($\times$5.2)	1.8 ($\times$31.2)

Comparable with that obtained by *Floating Scale Surface Reconstruction* (FSSR) but is $5.2\times \sim 33.2\times$ faster.

On Raw Data II

When processing an input with significant density variation – e.g., from four synthetic scans (most-left), FSSR and ours can avoid generating unwanted artifacts caused by high frequency noises.

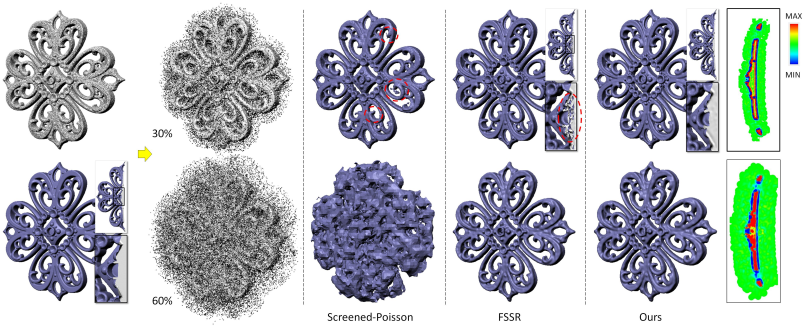


Input Screened-Poisson FSSR Ours

[†] The total time of our reconstruction is 6.81 sec. ($s = 3.0$) and 342k triangles are obtained on the resultant mesh, while FSSR takes 156.5 sec. ($\times 23$) and results in 301k triangles (scale=0.0105). Both are tested on a CPU with eight-cores.

Robustness

Reconstruction from sets (250k pts.) with different Gaussian noises.

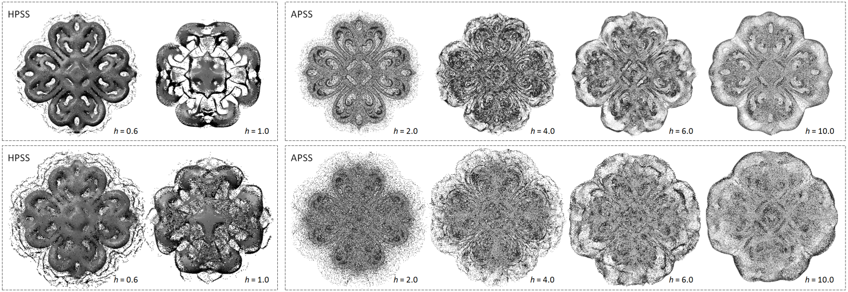


*FSSR generates some interior isolated regions (i.e., topological errors) but our method does not.

†Our method is 17.5× and 36.4× faster than FSSR on the 30% and 60% noisy models respectively.

Comparison with MLS methods

Applying *Hermite Point Set Surfaces* (HPSS) and *Algebraic Point Set Surfaces* (APSS) to the same sets of noisy data



Verification of Numerical and Geometric Errors

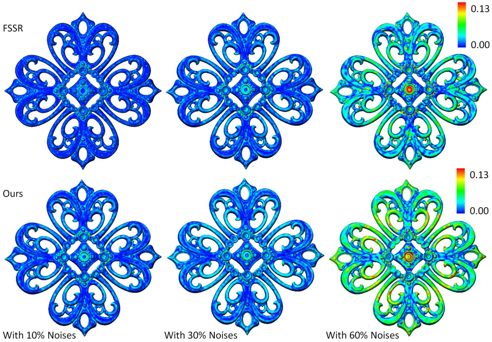
Study the real error (both numerical and geometric) on examples.

- Measure $\|\tilde{\boldsymbol{\lambda}} - \boldsymbol{\lambda}\|_\infty$ in examples shown above
- Evaluate forward-distance based errors on the results

Model	η	$\|\tilde{\boldsymbol{\lambda}} - \boldsymbol{\lambda}\|_\infty$
Ramesses	$457,616$	9.52×10^{-8}
Raptor	$1,666,700$	1.98×10^{-8}
Aquarius	$176,771$	3.46×10^{-7}
Horse	$149,459$	3.47×10^{-7}

* It can be easily found that our quasi-solution provides very accurate results on both the clean point cloud and the raw data.

† The numerical solver for computing the exact solution runs out of memory on the two examples – Momento (2.52M pts.) and Neptune (4.98M pts.).



Limitations and Challenges

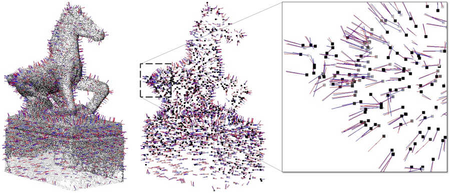
Limitations

Small fragments isolated from the main reconstruction could be formed by 1) numerical oscillation near the boundary of supporting regions and/or 2) outliers.

Conclusion Remarks

- A method can construct a signed scalar function by directly blending the positions and normals of points without any global operation – fast reconstruction.
- The computation based on CSRBF is local and robust.
- Errors between the quasi-solution and the exact one are bounded by controlling the support sizes of basis functions.
- Surface reconstruction based on our method can remove the artifacts resulted from noises (by changing the amplifier s) and non-uniformity (combining with center-selection).
- Reproducibility Stamp

Thanks for Your Questions



Charlie C.L. Wang

Department of Design Engineering
Delft University of Technology
E-mail: c.c.wang@tudelft.nl
http://homepage.tudelft.nl/h05k3/