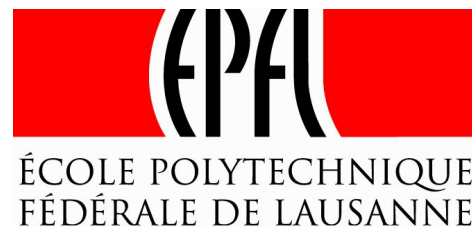
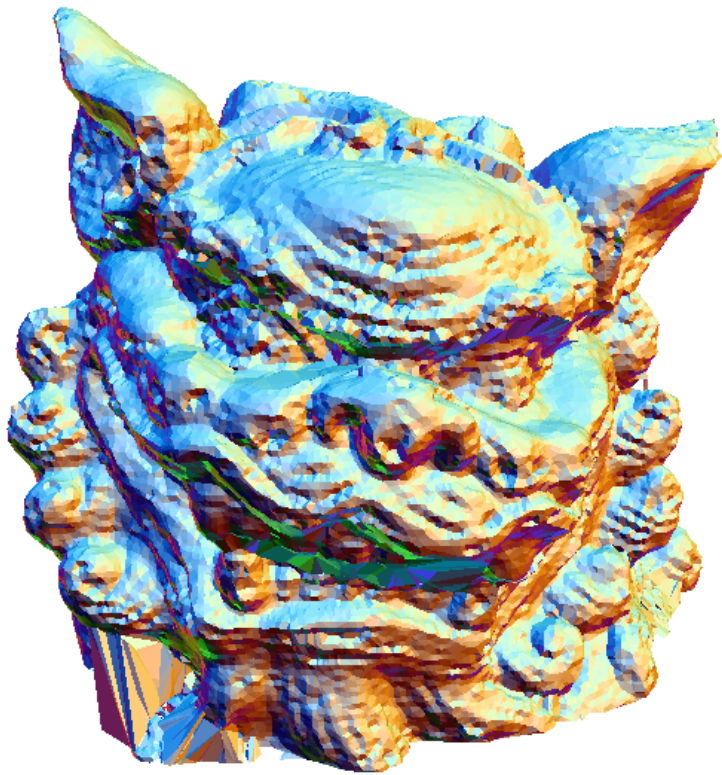


Guided Mesh Normal Filtering

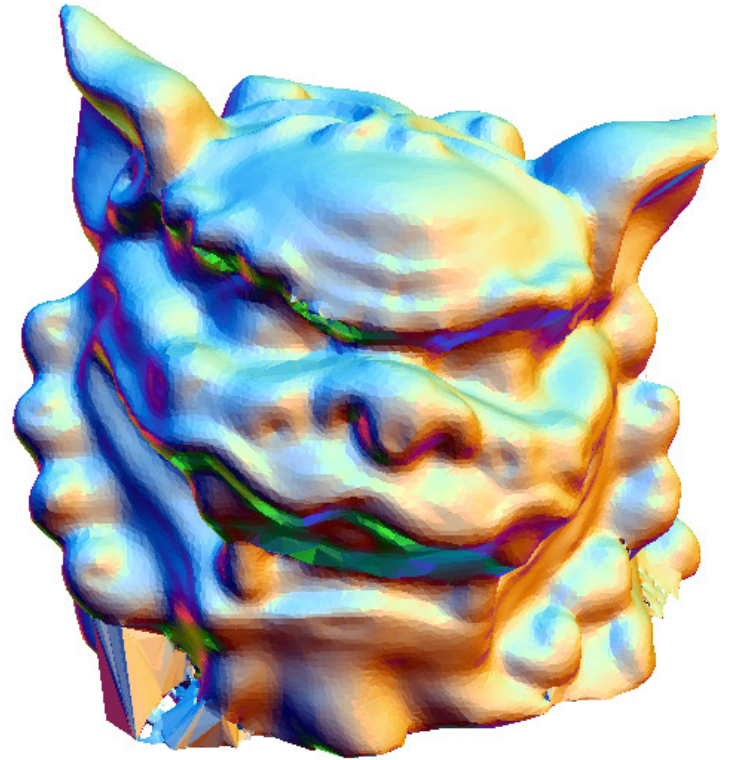
Wangyu Zhang	USTC
Bailin Deng	EPFL, University of Hull
<u>Juyong Zhang</u>	USTC
Sofien Bouaziz	EPFL
Ligang Liu	USTC



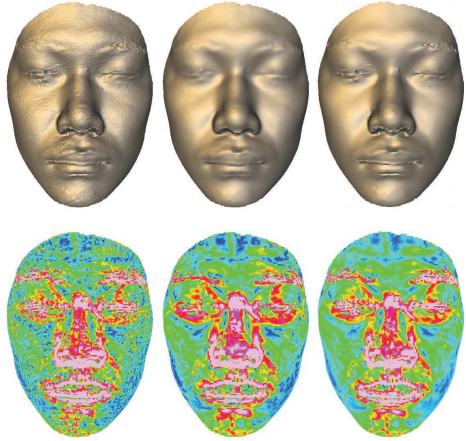
Filtering is necessary



Filtering



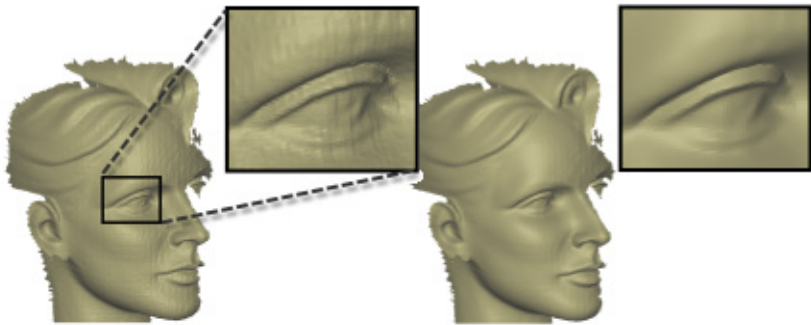
Related Work



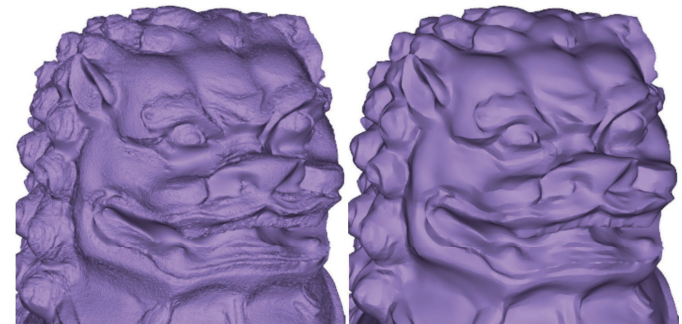
Bilateral mesh denoising
[Fleishman et al. 2003]



Non-iterative, feature-preserving mesh smoothing
[Jones et al. 2003]



Bilateral Normal Filtering for Mesh Denoising
[Zheng et al. 2011]

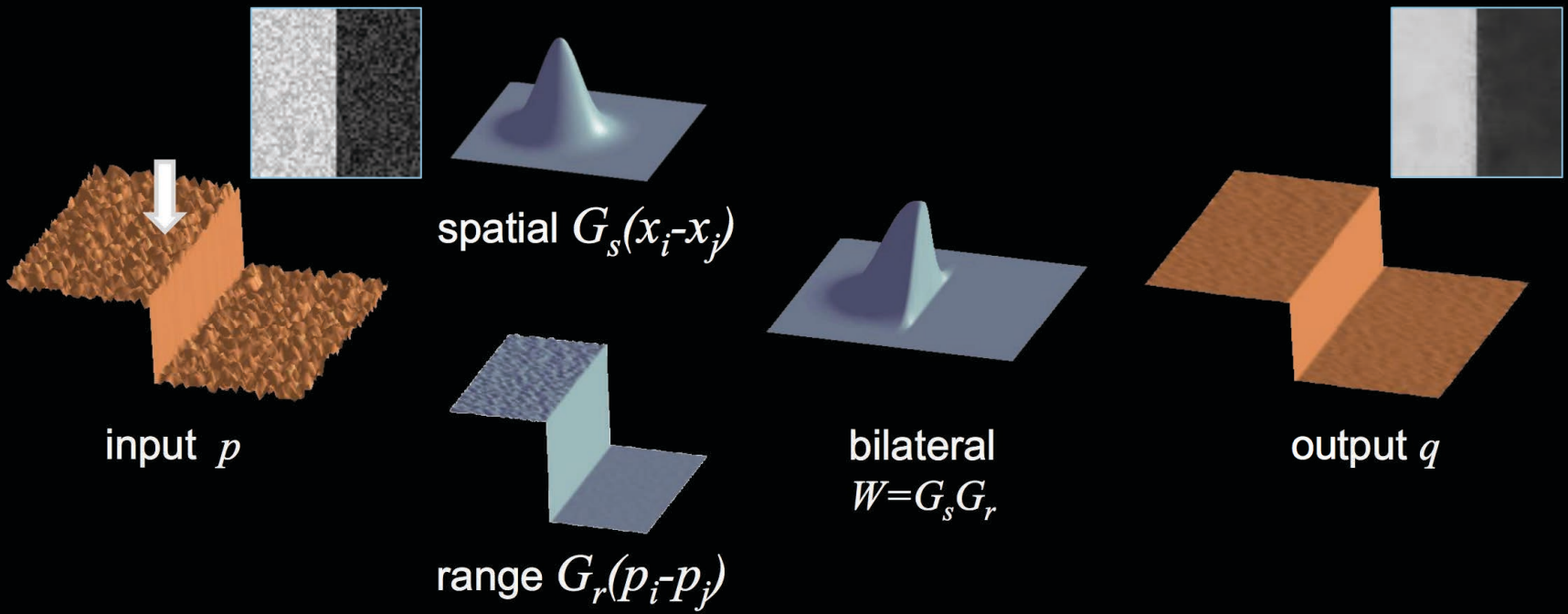


Mesh Denoising via L0 Minimization
[He & Schaefer 2013]

Bilateral filter

- Bilateral filter

$$q_i = \sum_{j \in N(i)} W_{ij}(p) p_j$$

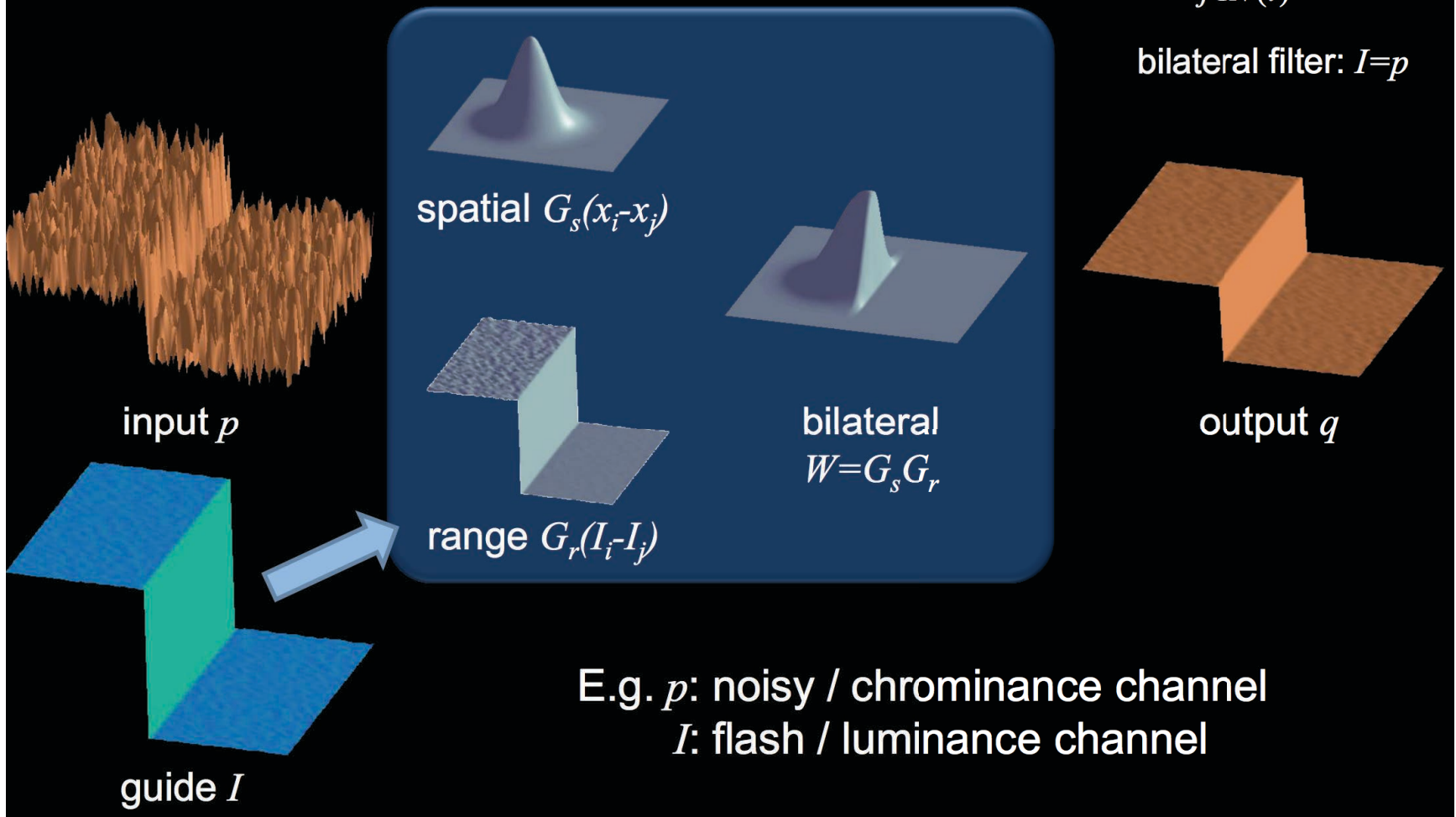


Joint bilateral filter

- Joint bilateral filter [Petschnigg et al. 2004]

$$q_i = \sum_{j \in N(i)} W_{ij}(I) p_j$$

bilateral filter: $I=p$

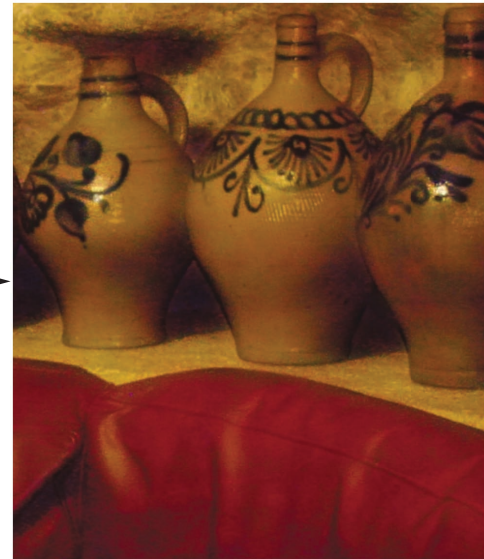
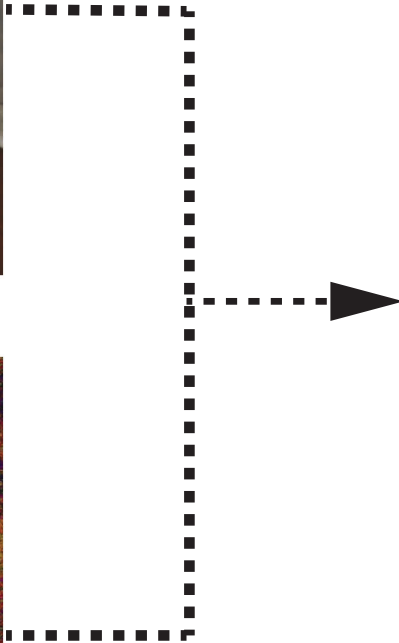




Flash



No-Flash

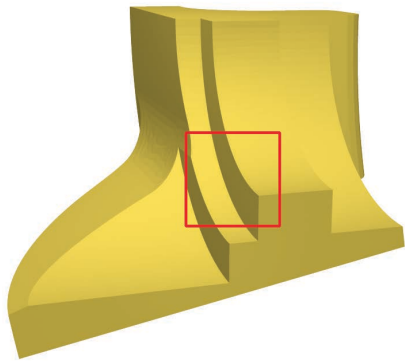


Result

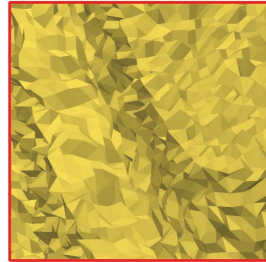
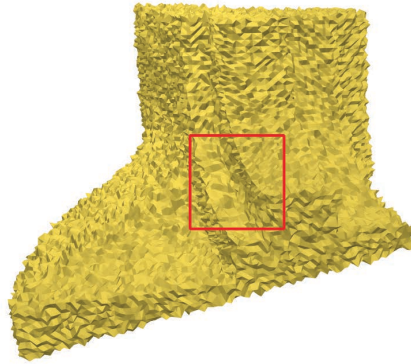
The role of guidance

- The success of joint bilateral filtering is heavily dependent on the guidance signal.
- The guidance signal should provide a robust estimation about the features of the output signal

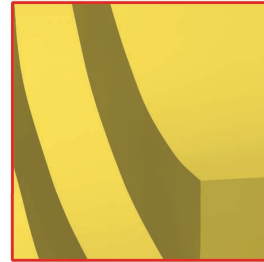
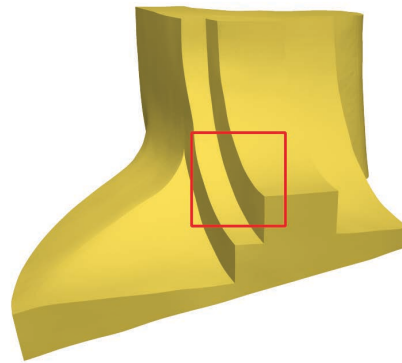
The importance of guidance



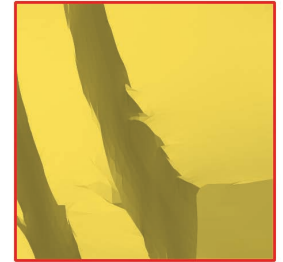
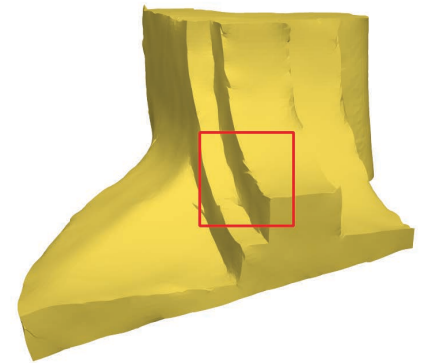
Original



Noisy



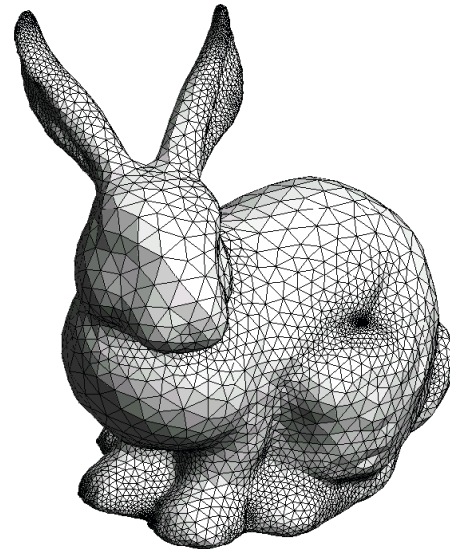
With Ground Truth Normals



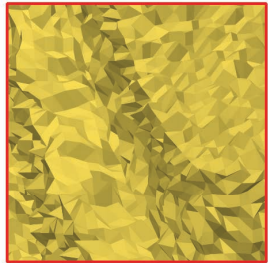
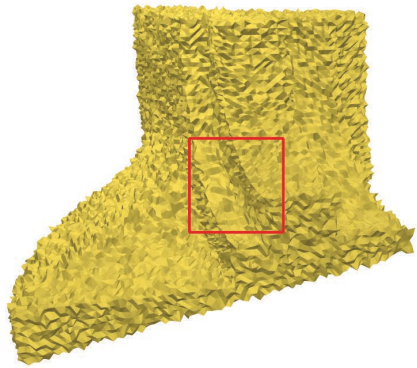
With Noisy Normals

Guidance geometry

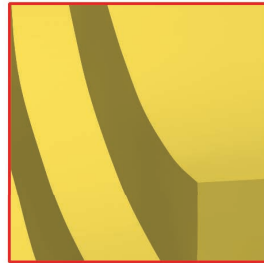
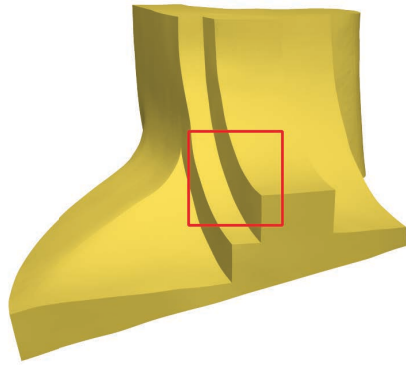
- Contrary to the case of images, such guidance geometry is not easily available from measure devices.
- It often has to be constructed computationally.



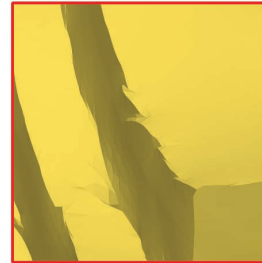
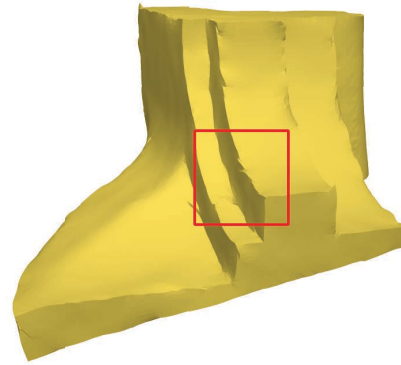
Example



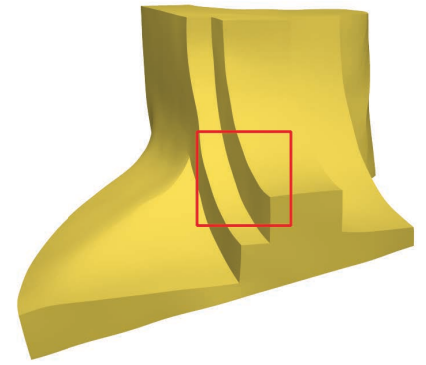
Noisy



With Ground Truth Normals

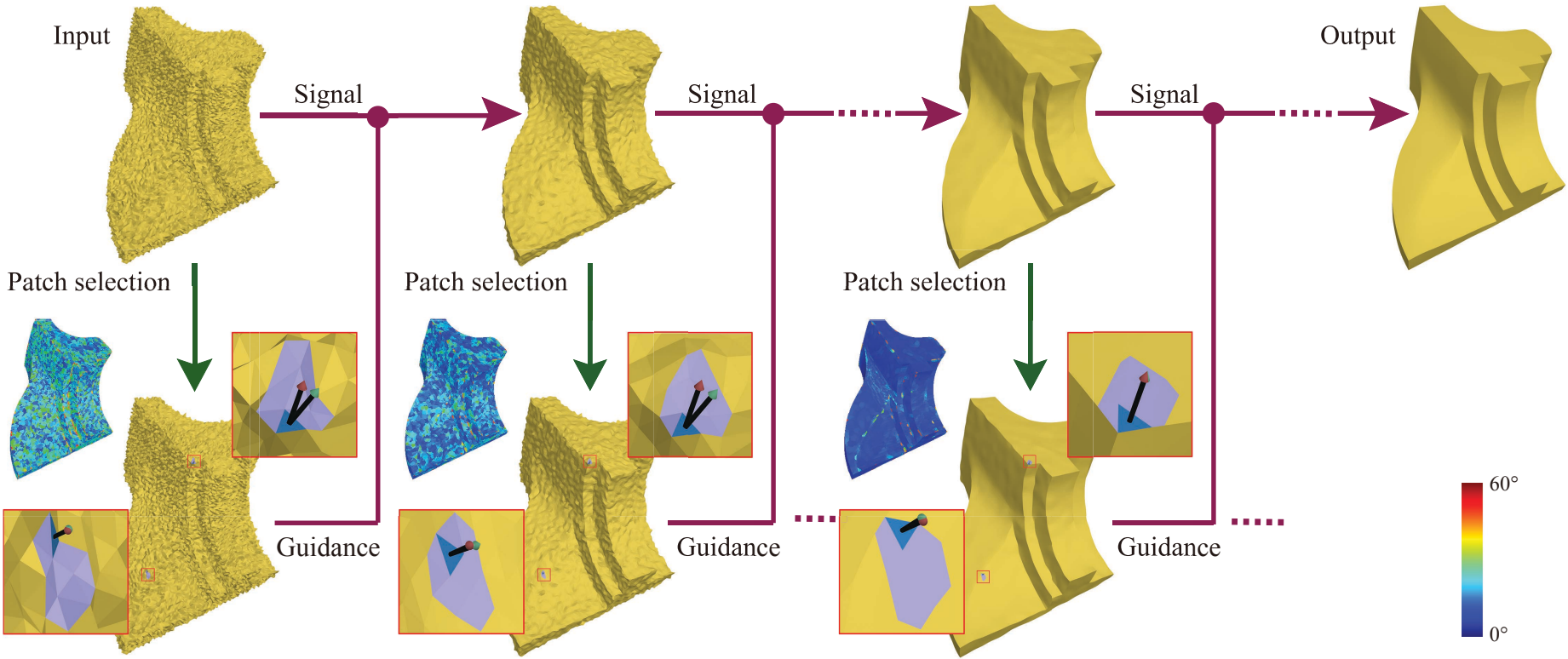


With Noisy Normals

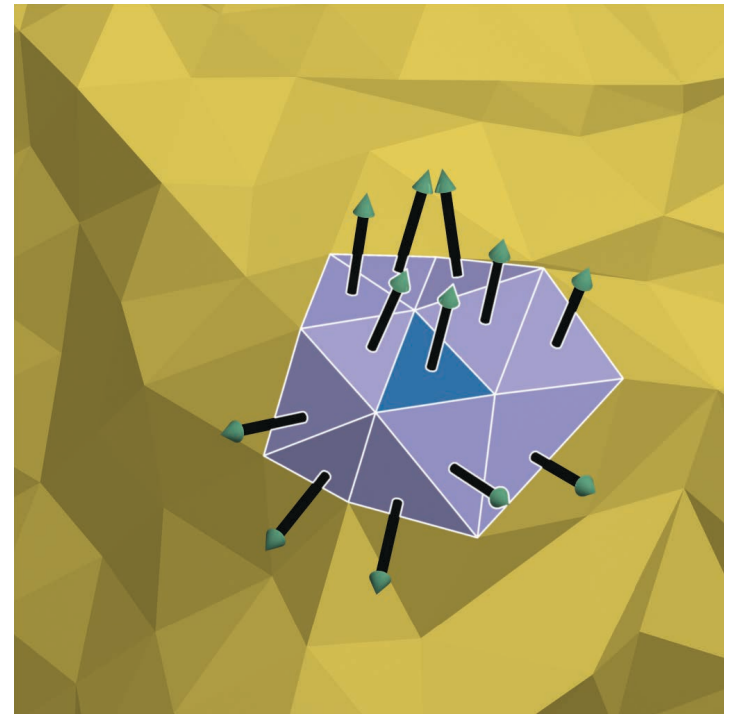
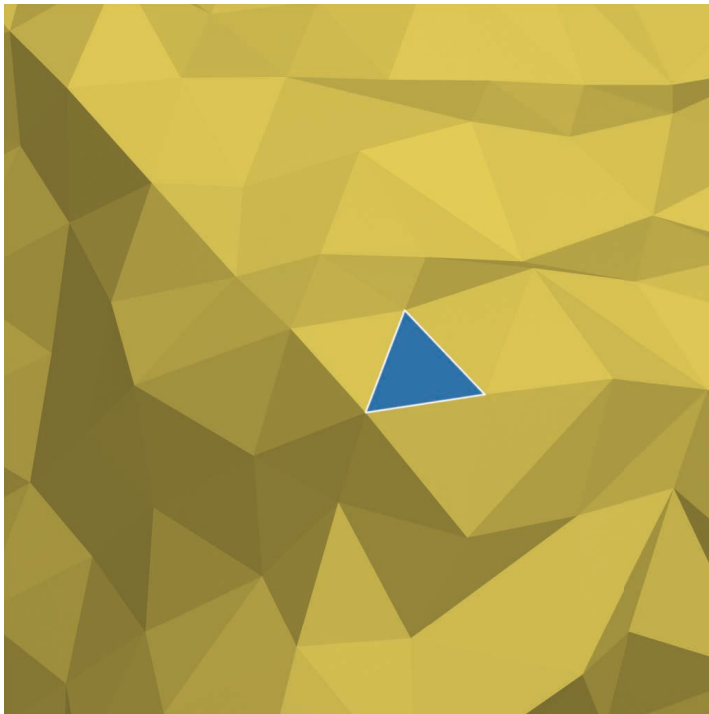


Ours

Denoising pipeline

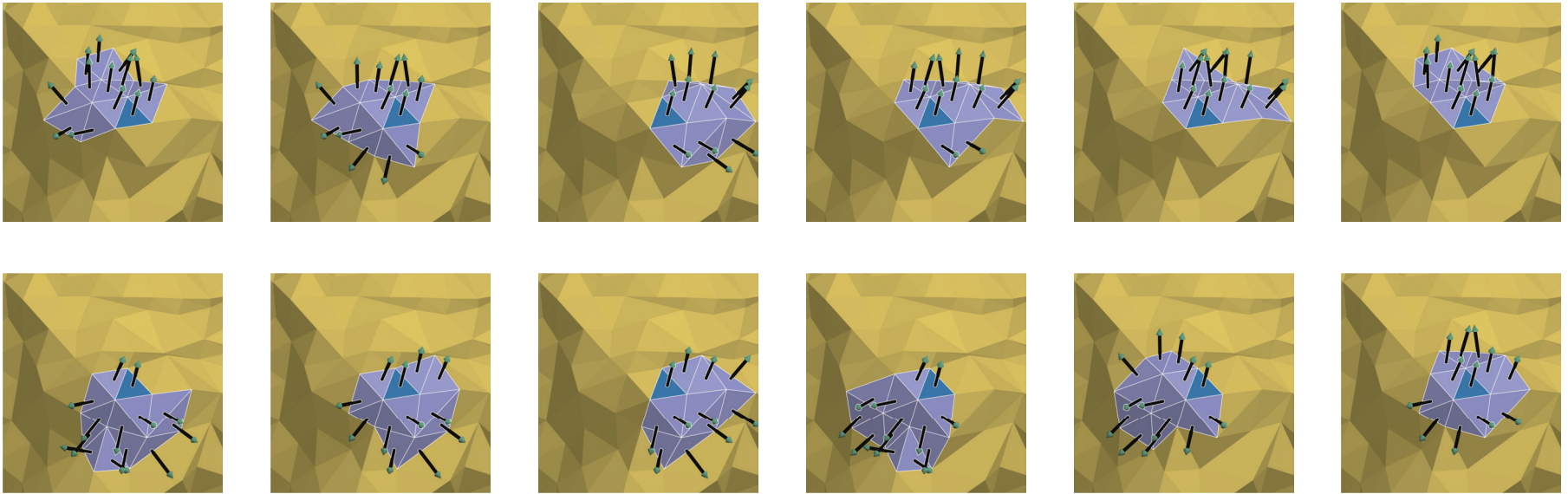


Guidance normal computation



Normal consistency

For each candidate patch $\mathcal{P} \in \mathcal{C}(f_i)$



we measure the consistency of its normals using

$$\mathcal{H}(\mathcal{P}) = \Phi(\mathcal{P}) \cdot \mathcal{R}(\mathcal{P})$$

Maximum normal difference

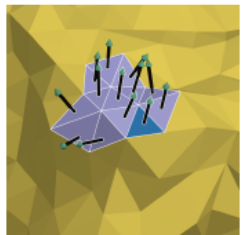
$$\Phi(\mathcal{P}) = \max_{f_j, f_k \in \mathcal{P}} \|\mathbf{n}_j - \mathbf{n}_k\|$$

Edge saliency measurement

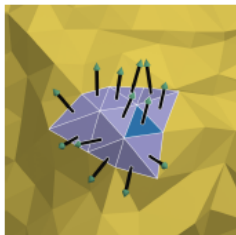
$$\mathcal{R}(\mathcal{P}) = \frac{\max_{e_j \in E_{\mathcal{P}}} \varphi(e_j)}{\varepsilon + \sum_{e_j \in E_{\mathcal{P}}} \varphi(e_j)}$$

$$\varphi(e_j) = \|\mathbf{n}_{j_1} - \mathbf{n}_{j_2}\|$$

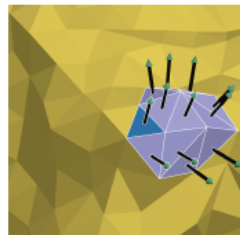
Patch selection



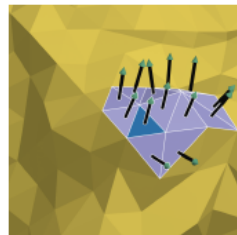
$\mathcal{H} = 0.323294$



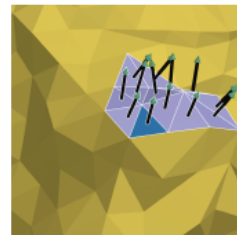
$\mathcal{H} = 0.297298$



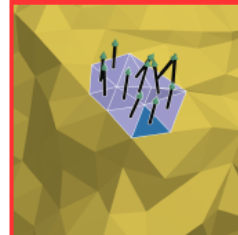
$\mathcal{H} = 0.312721$



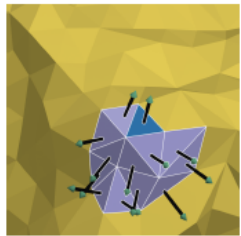
$\mathcal{H} = 0.303668$



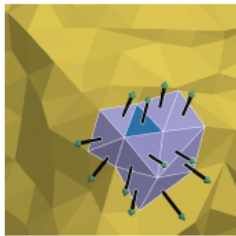
$\mathcal{H} = 0.146865$



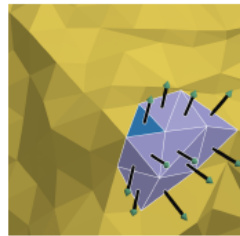
$\mathcal{H} = 0.136731$



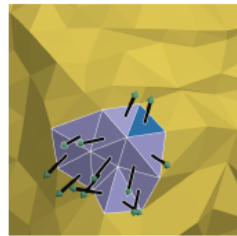
$\mathcal{H} = 0.274781$



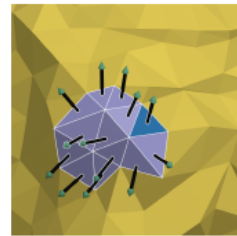
$\mathcal{H} = 0.293526$



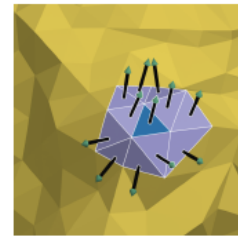
$\mathcal{H} = 0.304013$



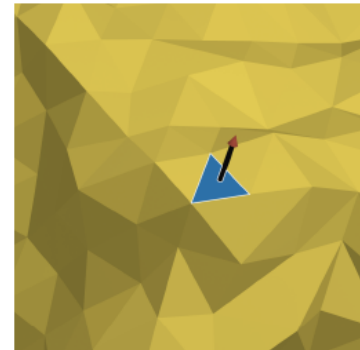
$\mathcal{H} = 0.276286$



$\mathcal{H} = 0.310027$



$\mathcal{H} = 0.332381$



Guidance normal

Normal filtering

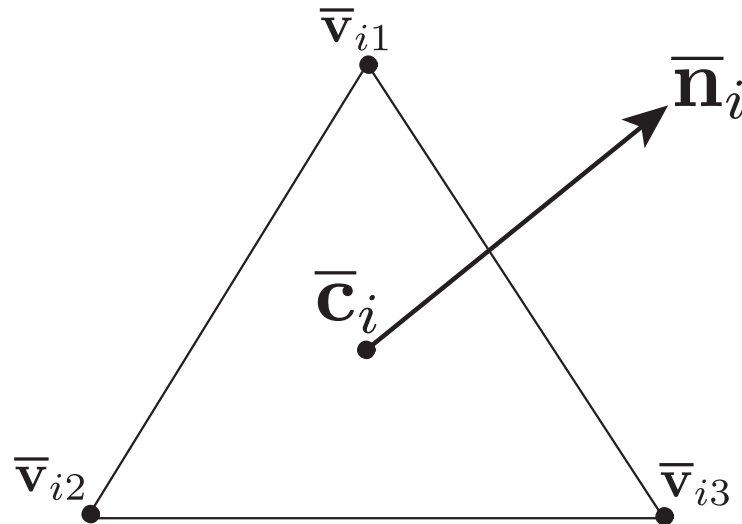
- Our normal filtering computes a new unit normal for each face via joint bilateral filter:

$$\bar{\mathbf{n}}_i = \frac{1}{W_i} \sum_{f_j \in \mathcal{N}_i} A_j K_s(\mathbf{c}_i, \mathbf{c}_j) K_r(\mathbf{g}_i, \mathbf{g}_j) \mathbf{n}_j$$

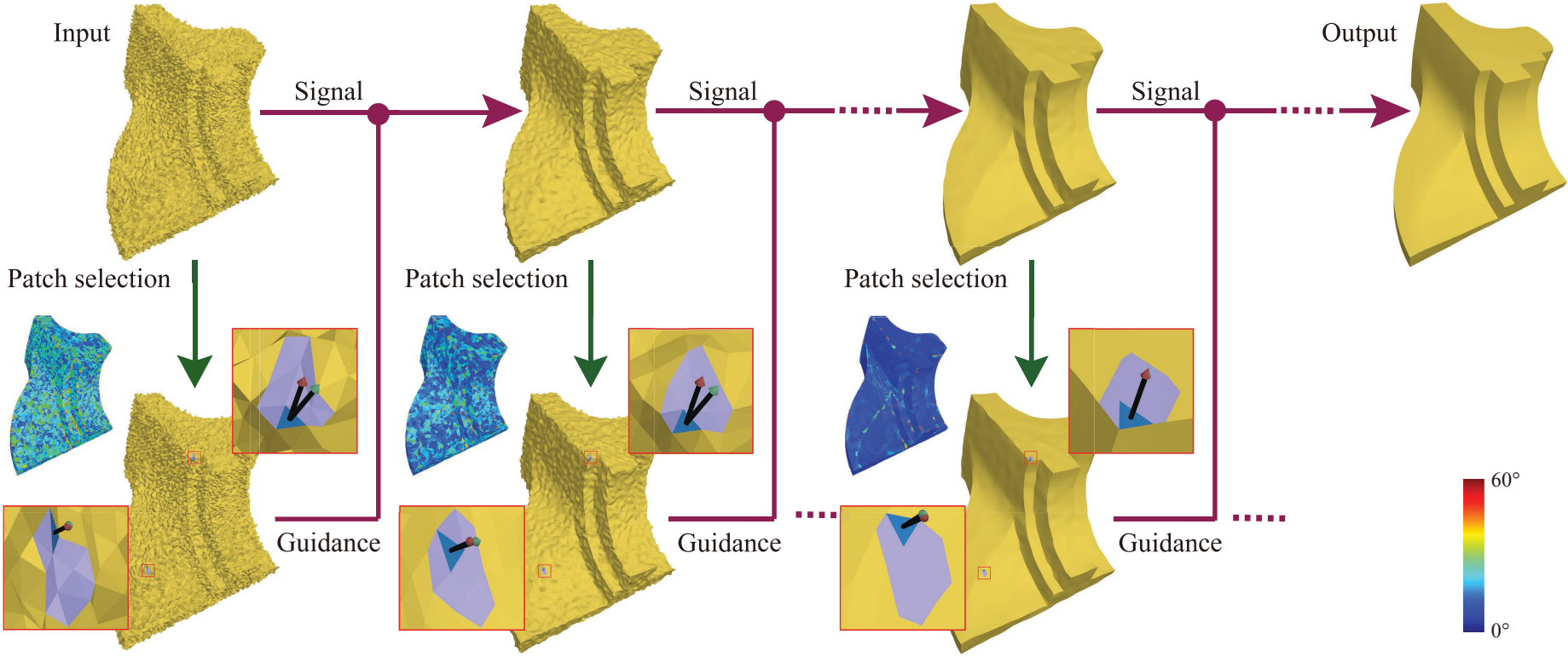
Updating vertices

- Based on the filtering face normals, the vertex positions are updated by minimizing the ℓ_2 error of the compatibility conditions:

$$\bar{\mathbf{n}}_i \cdot (\bar{\mathbf{v}}_{i_k} - \bar{\mathbf{c}}_i) = 0 \quad (k = 1, 2, 3)$$



Recap: pipeline

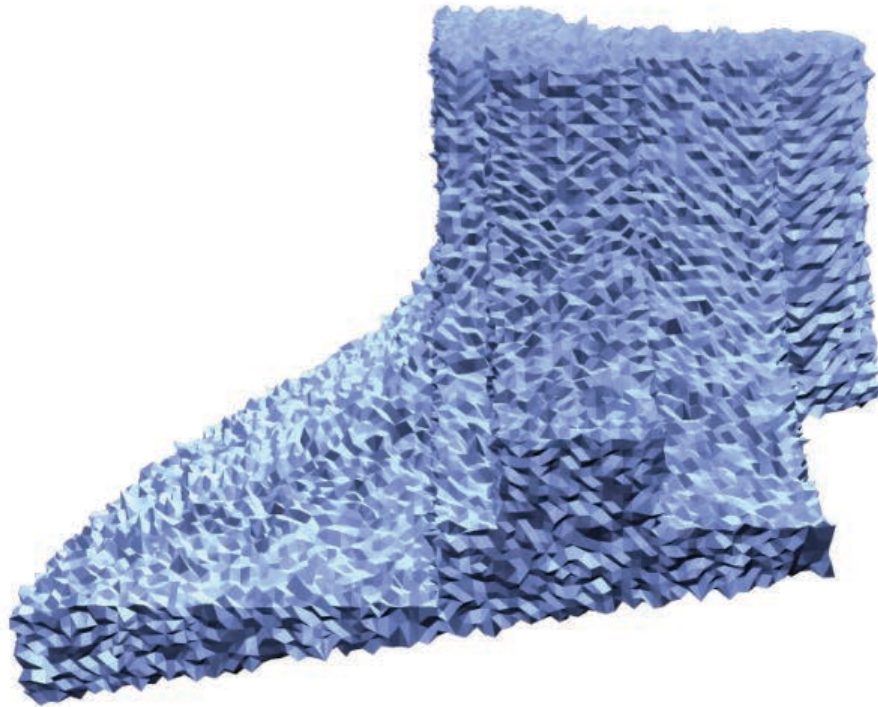
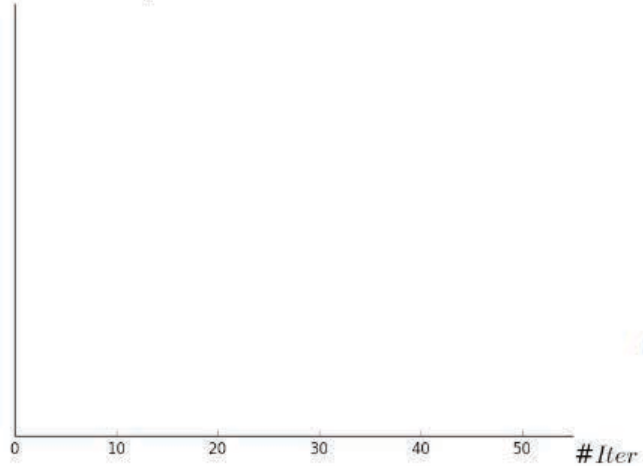


Denoising process

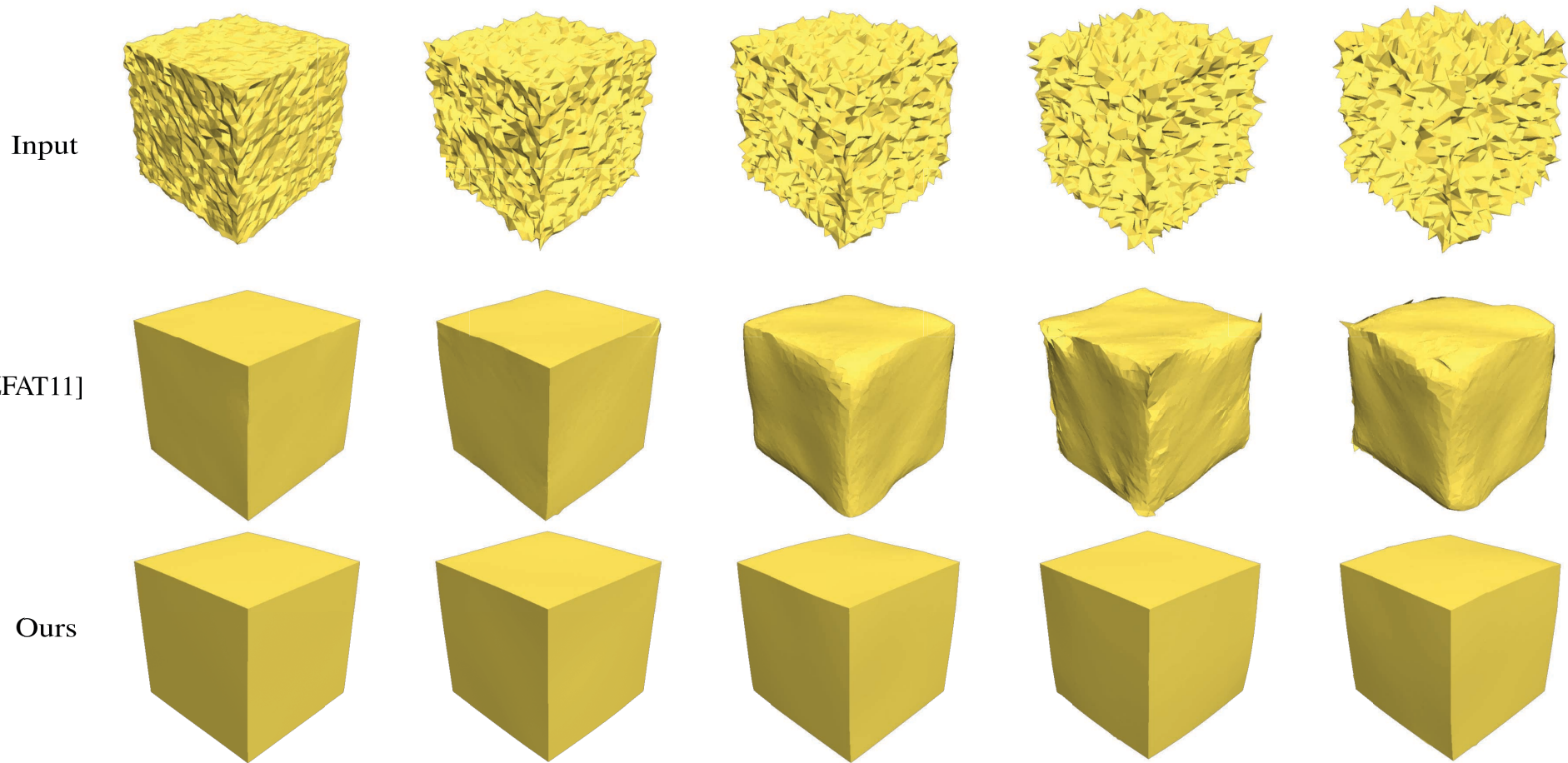
Denoising Process

Input

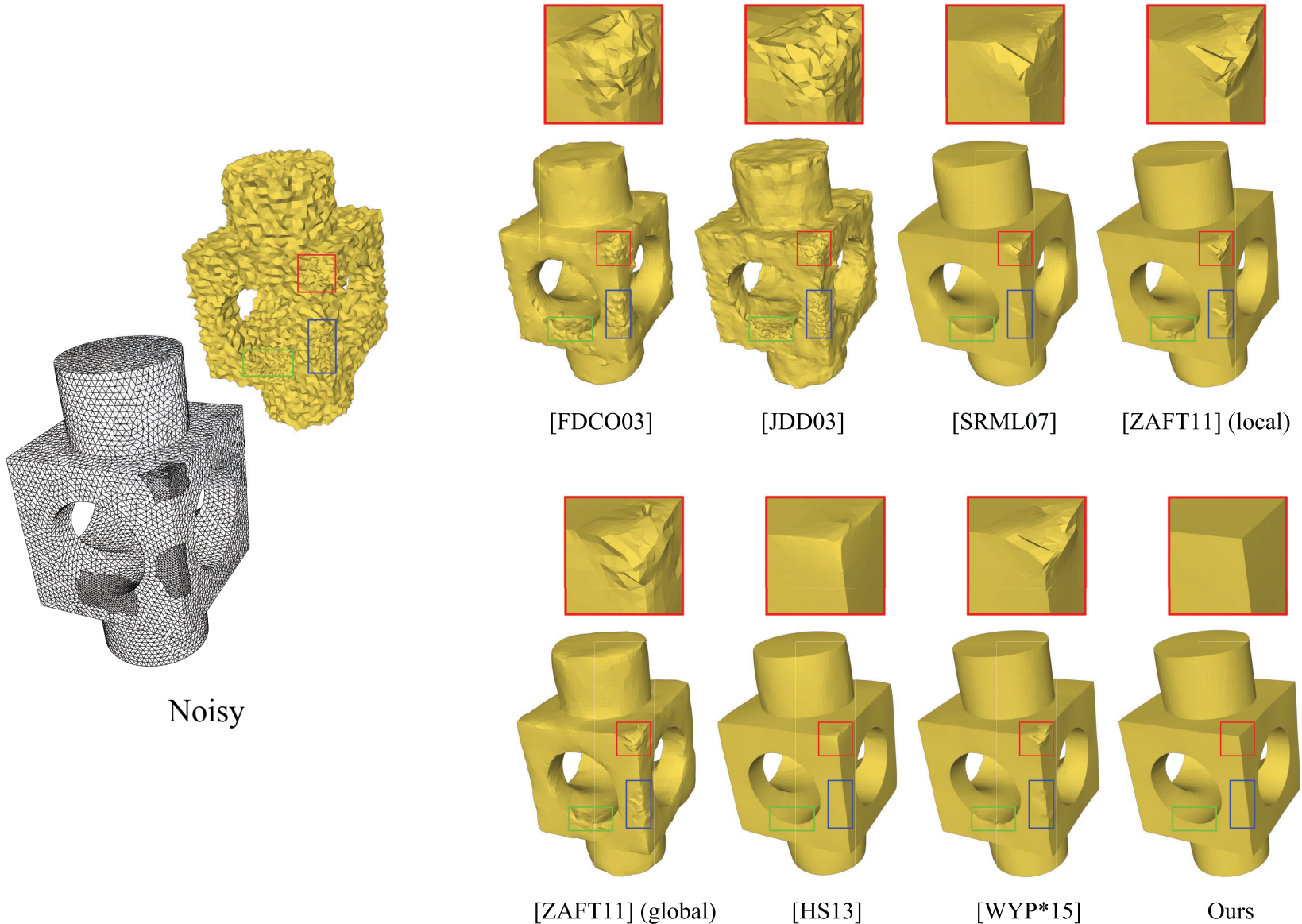
Error Metric l_2



Results: vs bilateral filter



Results: comparisons



Time statistics

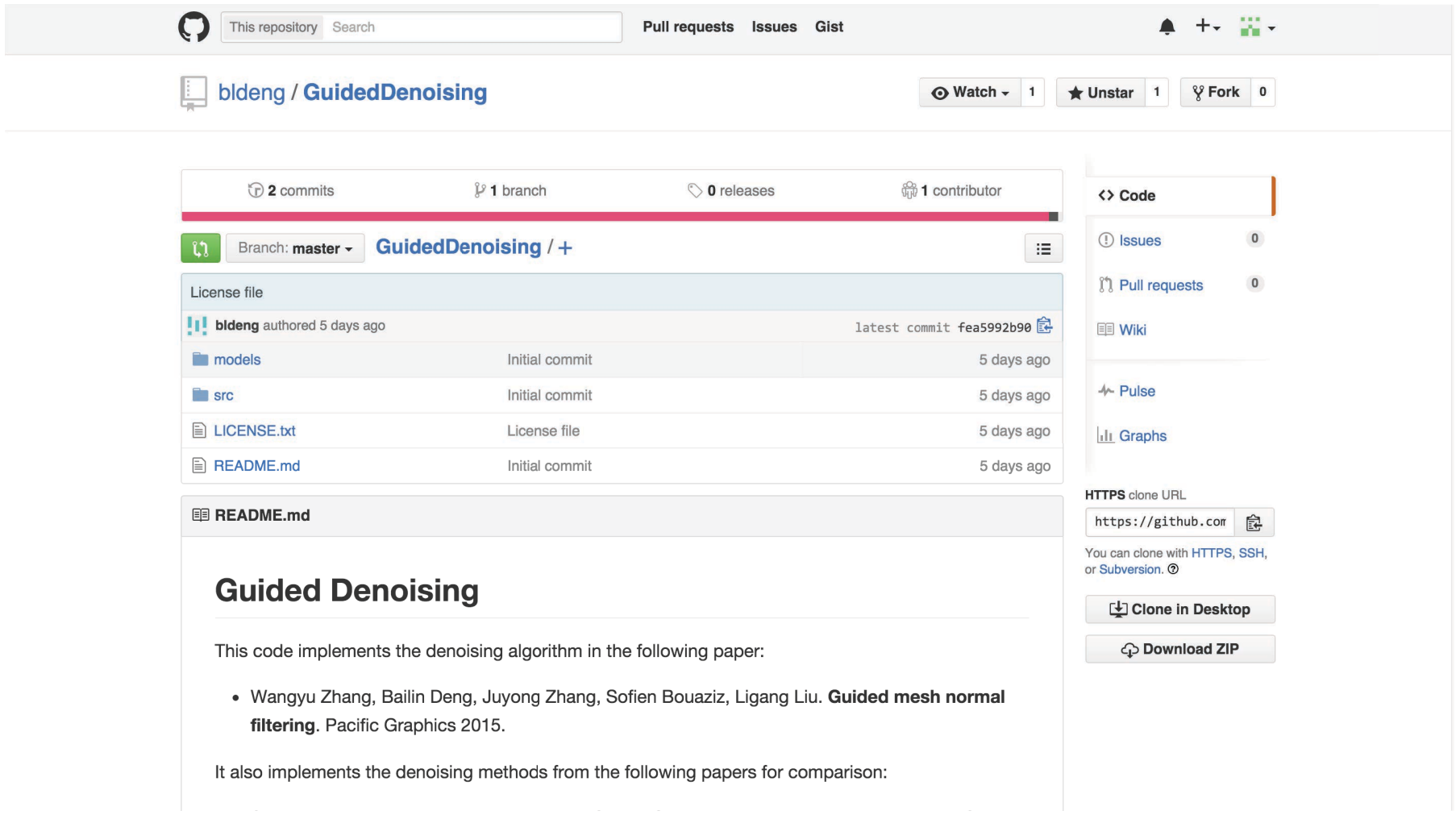
<i>Model</i>	<i>#Vertices</i>	<i>#Faces</i>	<i>Time(s)/Iter</i>
Fandisk	6475	12946	0.076
Block	8771	17550	0.104
Bunny	34834	69451	0.698
Iron	85574	168285	1.571

Conclusion

- A joint bilateral filter for mesh processing
 - ◆ A novel method to construct the guidance signal
 - ◆ Effective and efficient, simple to implement
 - ◆ Much better denoising results than state of the art

The source code is available:

<https://github.com/bldeng/GuidedDenoising>



The screenshot shows the GitHub repository page for `bldeng / GuidedDenoising`. The repository has 2 commits, 1 branch, 0 releases, and 1 contributor. The main branch is `master`. The repository contains a license file, a `models` directory, a `src` directory, `LICENSE.txt`, and `README.md`. The `README.md` file is displayed, containing the title `Guided Denoising` and a description of the code's purpose and references.

2 commits 1 branch 0 releases 1 contributor

Branch: `master` `GuidedDenoising` / +

License file

`bldeng` authored 5 days ago latest commit `fea5992b90`

File	Commit	Time
<code>models</code>	Initial commit	5 days ago
<code>src</code>	Initial commit	5 days ago
<code>LICENSE.txt</code>	License file	5 days ago
<code>README.md</code>	Initial commit	5 days ago

`README.md`

Guided Denoising

This code implements the denoising algorithm in the following paper:

- Wangyu Zhang, Bailin Deng, Juyong Zhang, Sofien Bouaziz, Ligang Liu. **Guided mesh normal filtering**. Pacific Graphics 2015.

It also implements the denoising methods from the following papers for comparison:

Code

- Issues 0
- Pull requests 0
- Wiki
- Pulse
- Graphs

HTTPS clone URL

`https://github.com`

You can clone with HTTPS, SSH, or Subversion.

Clone in Desktop

Download ZIP

Thank you!