Feature Preserving Point Set Surfaces based on Non-Linear Kernel Regression

A. C. Öztireli, G. Guennebaud, and M. Gross

ETH, National Research Council, and INRIA Bordeaux

Introduction

- Reformulating IMLS in terms of Local Kernel Regression (LKR)
- Borrowing ideas from robust statistics
- Advantages:
  - Robust to sparse sampling
  - Preserve sharp features
  - Controllable sharpness
  - Intuitive and easy to Implement
  - Competitive performance

Local Kernel Regression

- Taylor expansion around the evaluation point x

$$f(\mathbf{x}_i) \approx f(\mathbf{x}) + (\mathbf{x}_i - \mathbf{x})^T \nabla f(\mathbf{x}) + \frac{1}{2}(\mathbf{x}_i - \mathbf{x})^T \mathbf{H} f(\mathbf{x})(\mathbf{x}_i - \mathbf{x}) + \ldots$$

- Local best fit by looking for unknowns $\{s_i\}$

$$f(\mathbf{x}_i) \approx s_0 + \mathbf{a}_i^T \mathbf{s}_1 + \mathbf{b}_i^T \mathbf{s}_2 + \ldots$$

where $\mathbf{a}_i = (\mathbf{x}_i - \mathbf{x})$, and $\mathbf{b}_i = [\ldots\ (\mathbf{a}_i)_j(\mathbf{a}_i)_k\ \ldots]^T$

- Weighted least square optimization:

Symmetric & Decreasing!

$$\arg\min_{\mathbf{s}} \sum (y_i - (s_0 + \mathbf{a}_i^T \mathbf{s}_1 + \mathbf{b}_i^T \mathbf{s}_2 + \ldots))^2 \boxed{\phi_i(\mathbf{x})}$$

Implicit MLS [Kolluri et al. 2005]

- Zero-order LKR:

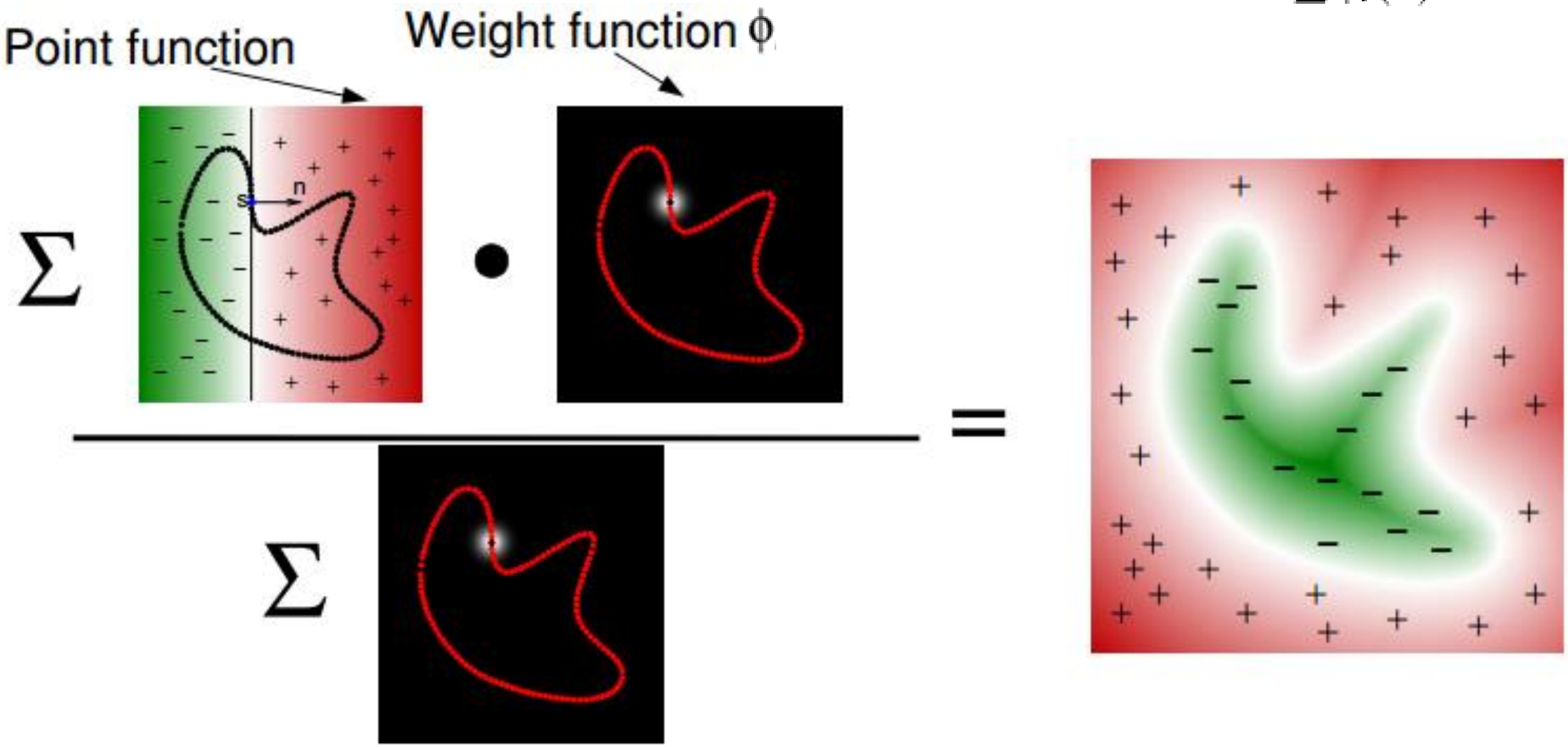
$$\begin{aligned} & \arg\min_{s_0,\mathbf{s}_1} \sum (y_i - (s_0 + \mathbf{a}_i^T \mathbf{s}_1))^2 \phi_i(\mathbf{x}) \\ = \; & \arg\min_{s_0} \sum (s_0 + (\mathbf{x}_i - \mathbf{x})^T \mathbf{n}_i)^2 \phi_i(\mathbf{x}) \end{aligned}$$

- Minimize by

$$f(\mathbf{x}) = s_0 = \frac{\sum \mathbf{n}_i^T (\mathbf{x} - \mathbf{x}_i) \phi_i(\mathbf{x})}{\sum \phi_i(\mathbf{x})}$$

Implicit MLS [Kolluri et al. 2005]

$$f(\mathbf{x}) = s_0 = \frac{\sum \mathbf{n}_i^T(\mathbf{x}-\mathbf{x}_i)\phi_i(\mathbf{x})}{\sum \phi_i(\mathbf{x})}$$



$$w(x) = e^{-\left(\frac{x}{\sigma_r}\right)^2}$$

Robust Implicit MLS (RIMLS)

- Iterative Minimization:

$$f^k(\mathbf{x}) = \arg\min_{s_0} \sum (s_0 + (\mathbf{x}_i - \mathbf{x})^T \mathbf{n}_i)^2 \phi_i(\mathbf{x}) w(r_i^{k-1})$$

with the residuals $r_i^{k-1} = f^{k-1}(\mathbf{x}) - (\mathbf{x} - \mathbf{x}_i)^T \mathbf{n}_i$.

Robust Implicit MLS (RIMLS)

$$w_n(\Delta\mathbf{n}_i^k) = e^{-\frac{(\Delta\mathbf{n}_i^k)^2}{\sigma_n^2}}$$

$$\Delta\mathbf{n}_i^k = \|\nabla f^k(\mathbf{x}) - \mathbf{n}_i\|$$

- Bilateral Filtering
  (position + residual)

$$f^k(\mathbf{x}) = \frac{\sum \mathbf{n}_i^T(\mathbf{x} - \mathbf{x}_i)\phi_i(\mathbf{x})w(r_i^{k-1})}{\sum \phi_i(\mathbf{x})w(r_i^{k-1})}$$

- Trilateral Filtering
  (position + residual + normal difference)

$$f^k(\mathbf{x}) = \frac{\sum \mathbf{n}_i^T(\mathbf{x} - \mathbf{x}_i)\phi_i(\mathbf{x})w(r_i^{k-1})w_n(\Delta\mathbf{n}_i^{k-1})}{\sum \phi_i(\mathbf{x})w(r_i^{k-1})w_n(\Delta\mathbf{n}_i^{k-1})}$$

Pseudo-code

```
repeat
  i = 0;
  repeat
    sumW = sumGw = sumF = sumGF = sumN = 0;
    for p in neighbors(x) do
      px = x - p.position;
      fx = dot(px, p.normal);

      if i>0 then alpha = exp(-((fx-f)/sigma_r)^2)
                        * exp(-(norm(p.normal-grad_f)/sigma_n)^2);
            else alpha = 1;

      w      = alpha * phi(norm(px)^2);
      grad_w = alpha * 2 * px * dphi(norm(px)^2);

      sumW  += w;
      sumGw += grad_w;
      sumF  += w * fx; sumGF += grad_w * fx;
      sumN  += w * p.normal;
    end
    f      = sumF / sumW;
    grad_f = (sumGf - f * sumGw + sumN) / sumW;
  until ++i>max_iters || convergence();
  x = x - f * grad_f;
until norm(f * grad_f) < threshold;
```