PHILIPS

GPU Processing within Philips Healthcare

Danny Ruijters

1 September 2010

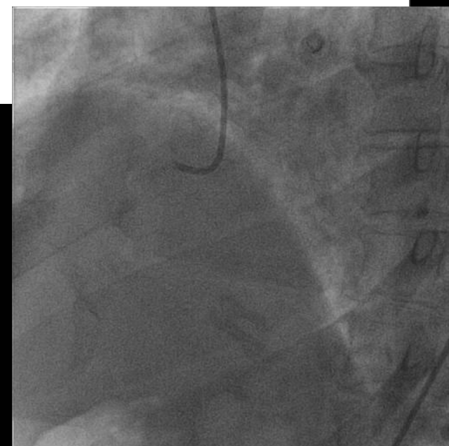
Outline

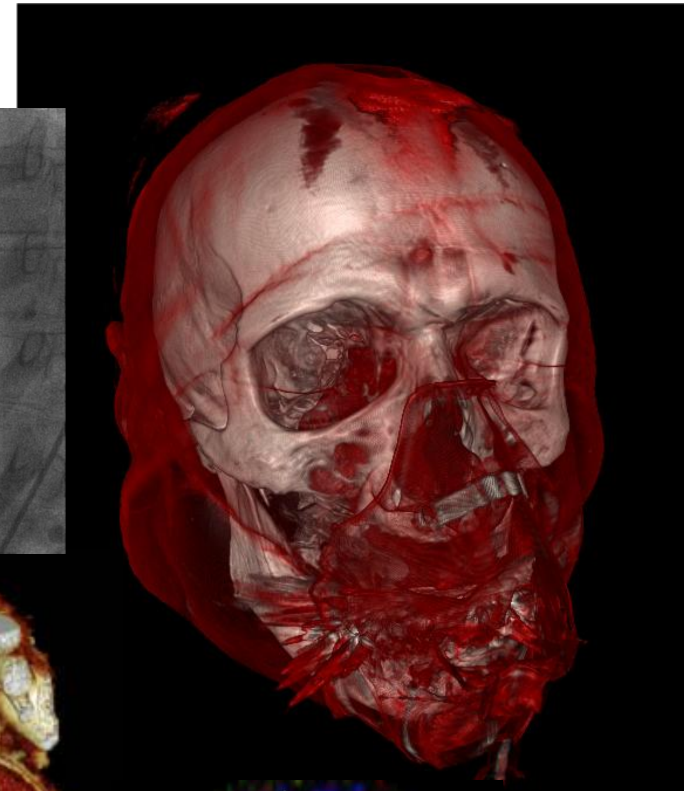
- Philips Healthcare
- The GPU within Philips Healthcare
- Examples
  - Rigid 3D-3D Registration
  - Elastic Registration

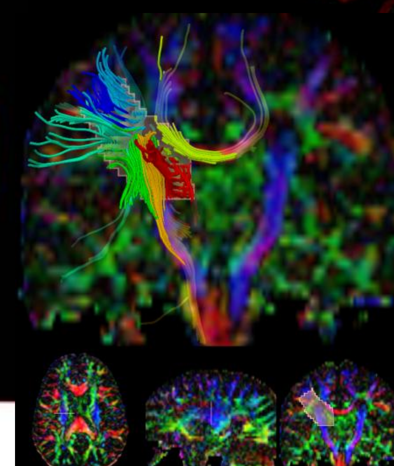
Philips Healthcare

Imaging Systems

Produces image data







The GPU within Philips Healthcare

GPU Usage

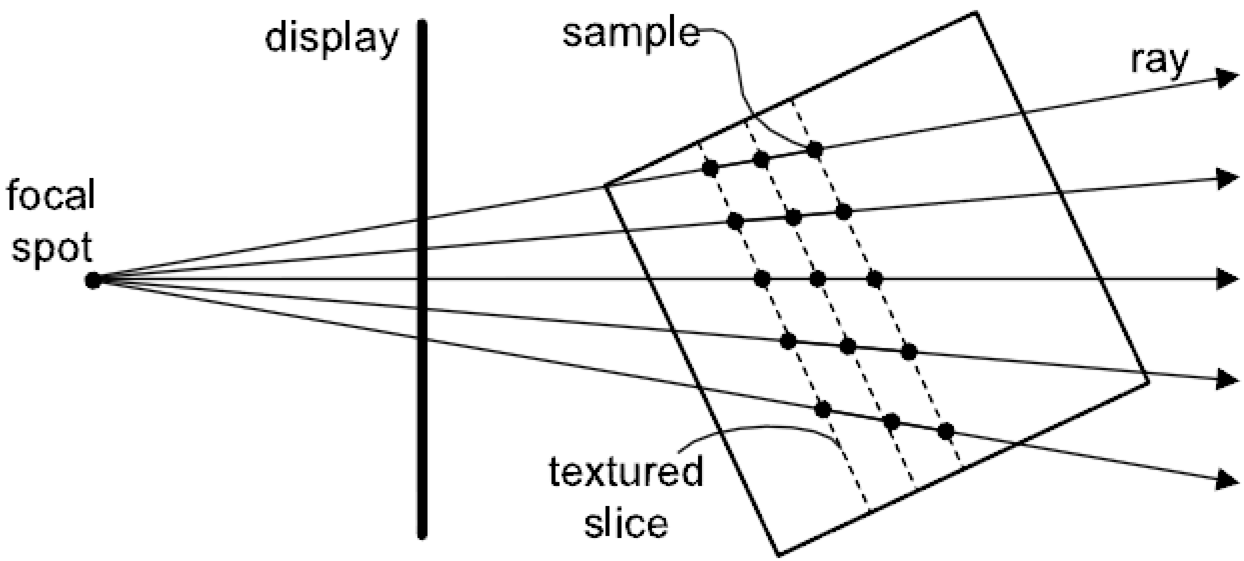
- Visualization
  - 2D (+ time), 3D (+ time), …
- Image Processing
  - 3D Reconstruction (CT, MR)
  - Segmentation
  - Registration (matching)
  - …

API's

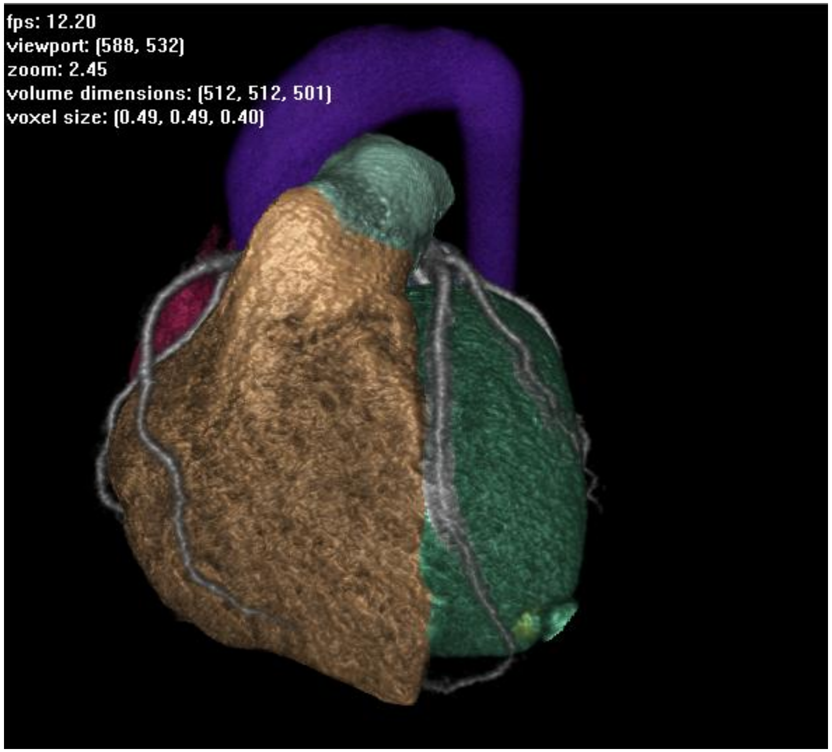
- OpenGL
- DirectX
- CUDA
- OpenCL
- DirectCompute

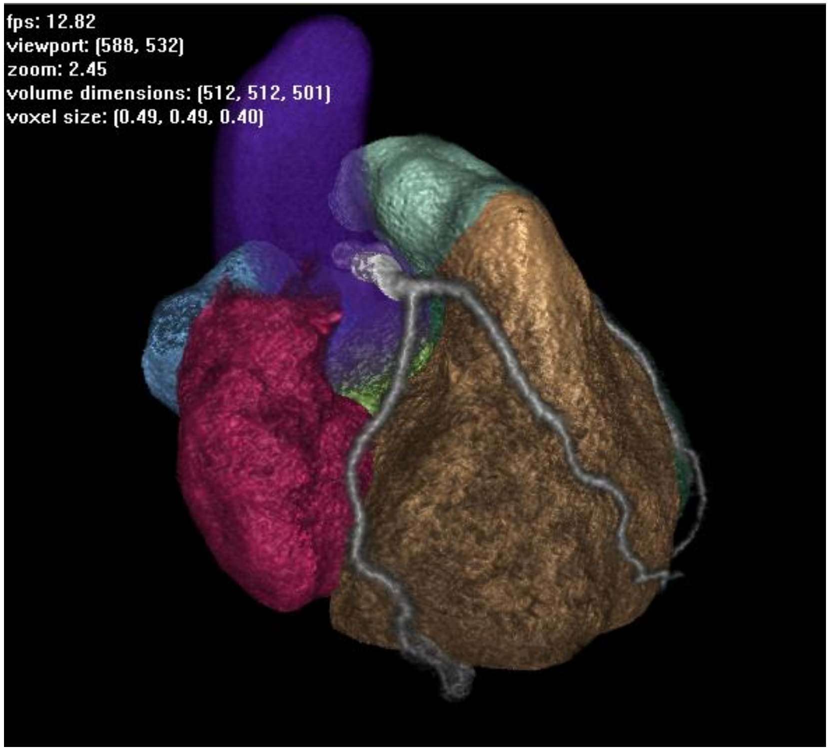
Reconstruction

$$f(\vec{x}) = \frac{1}{4\pi} \int_0^{2\pi} p^*(\alpha, (\cos\alpha, \sin\alpha) \cdot \vec{x})\, d\alpha$$



Segmentation





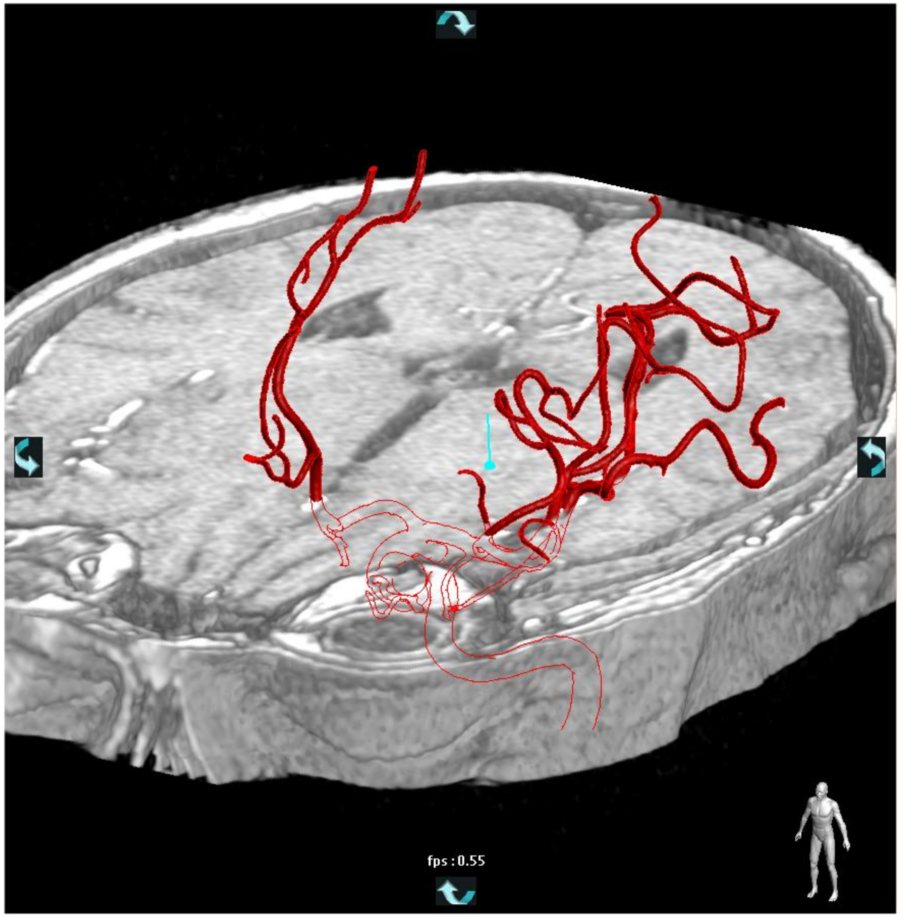
Examples

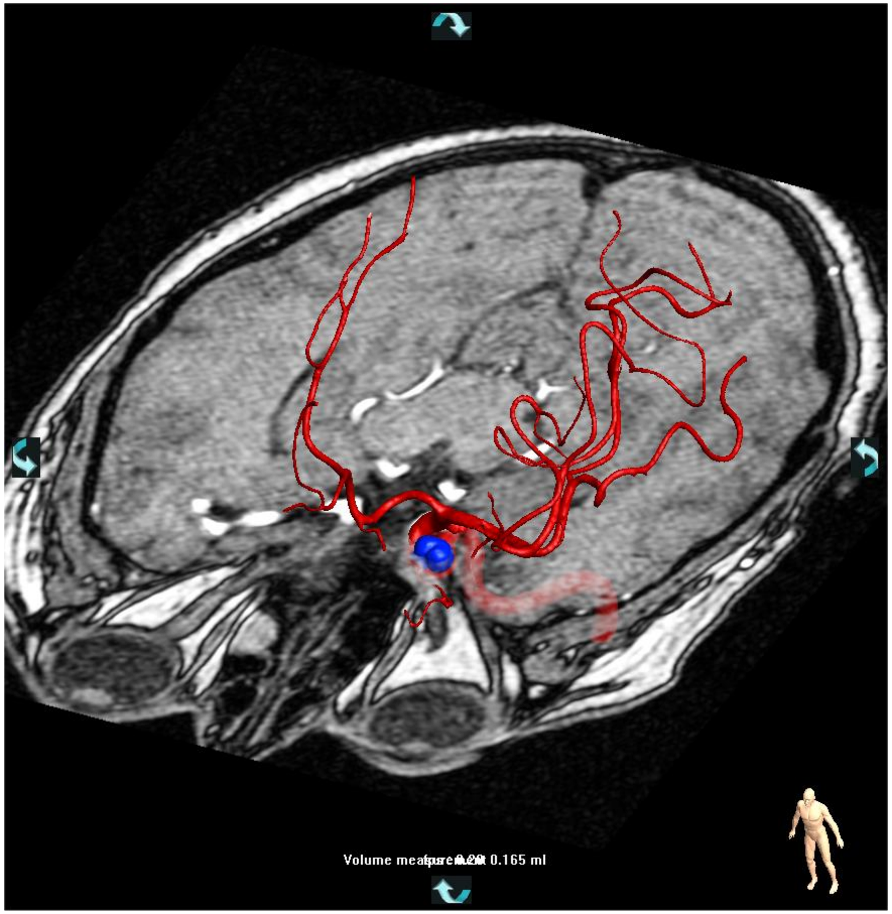
Rigid 3D-3D Registration

Elastic Registration

Rigid 3D-3D Registration

3DRA – MR registration





Mutual information

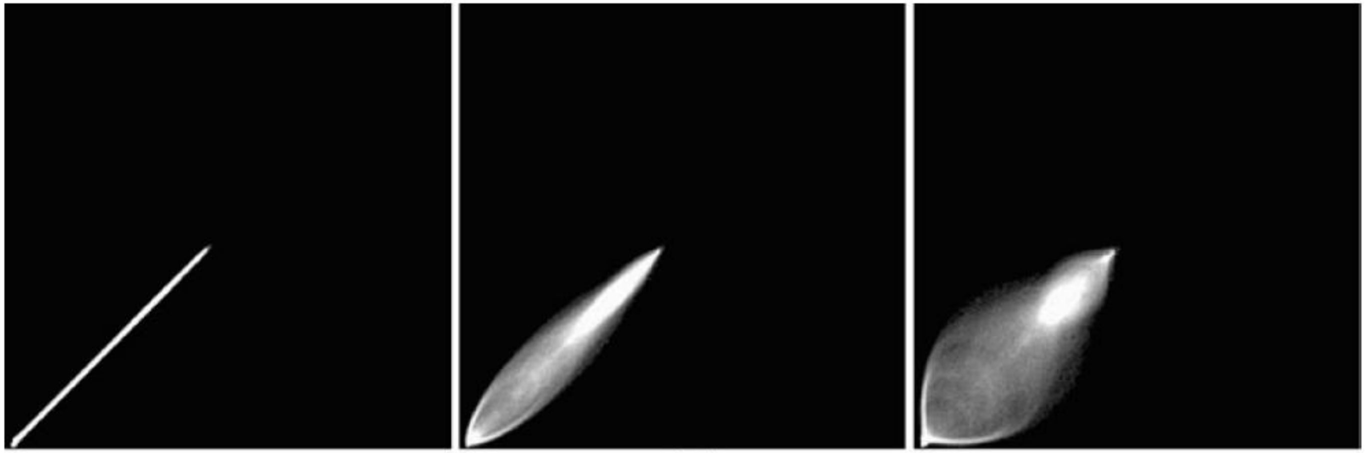
$$I(A,B) = \sum_{a,b} p_{AB}(a,b) \log \frac{p_{AB}(a,b)}{p_A(a).p_B(b)}$$

MI is related to entropy by the equations

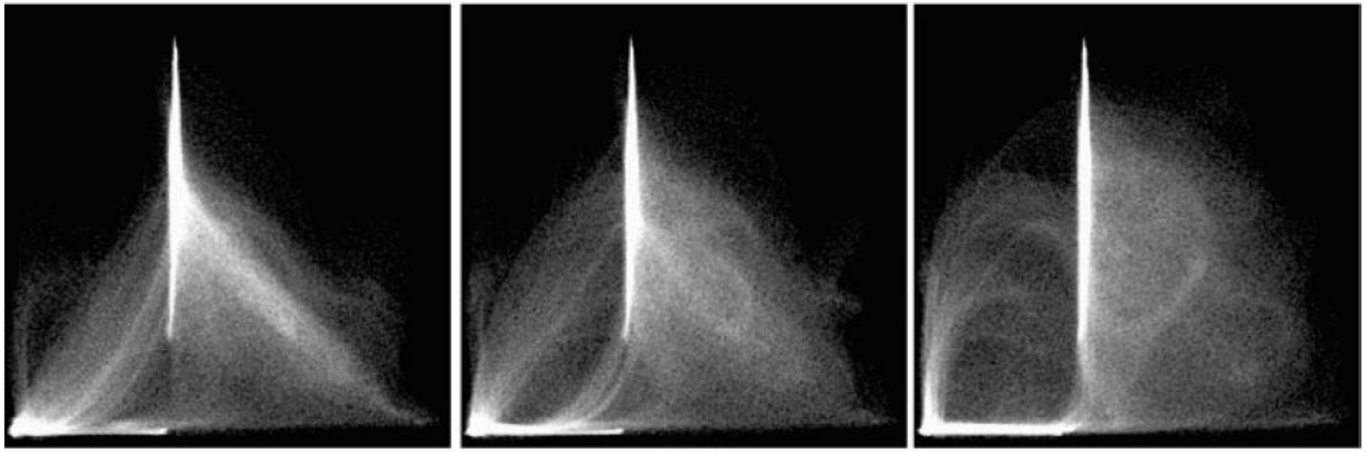
$$\begin{aligned} I(A,B) &= H(A) + H(B) - H(A,B) \\ &= H(A) - H(A \mid B) \\ &= H(B) - H(B \mid A) \end{aligned}$$

F. Maes et al., "Multimodality Image Registration by Maximization of Mutual Information,“ IEEE Transactions on Medical Imaging 16(2), pp. 187-198, April 1997

Joint histogram

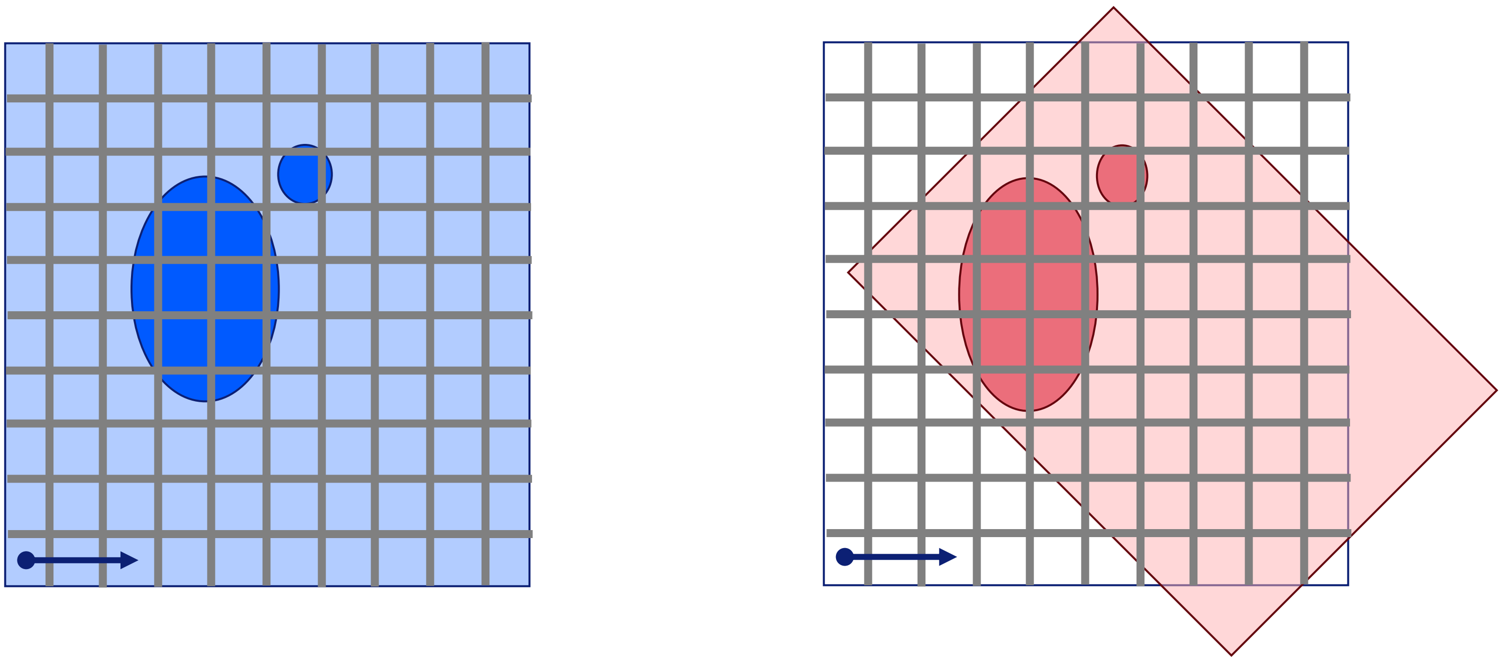


(a)



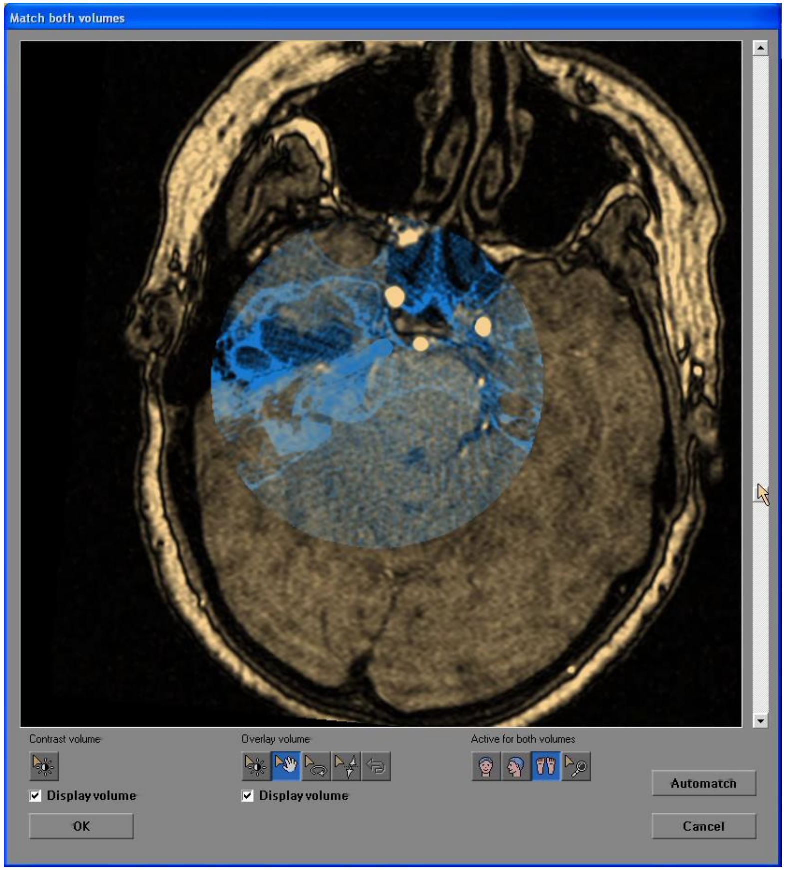
(b)

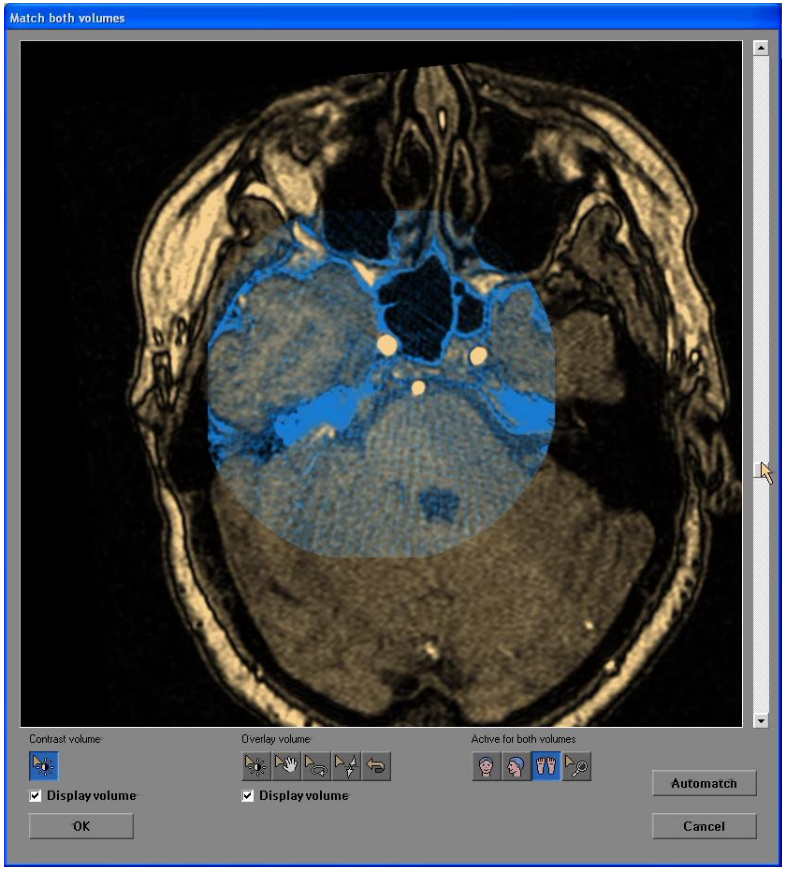
Resampling



Joint histogram: increment(g,g)

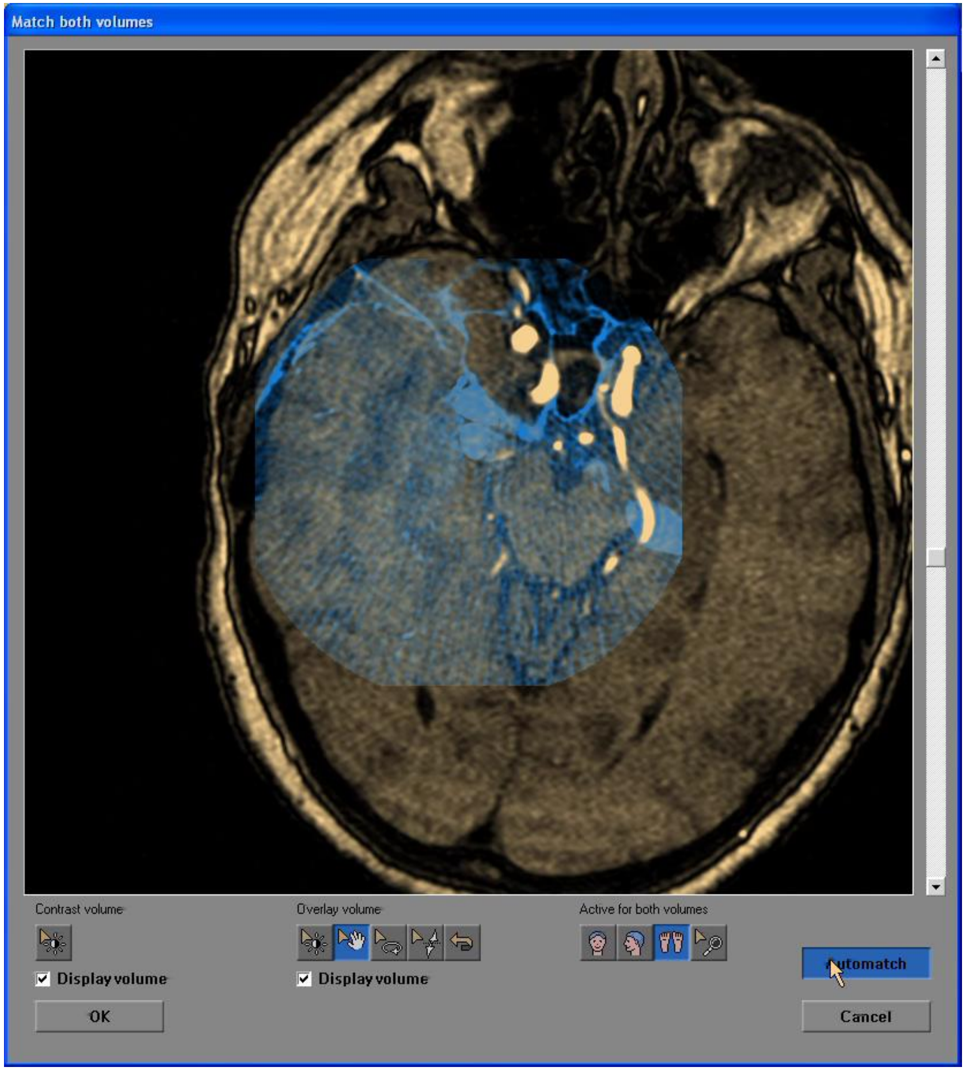
3DRA – MR, before, after



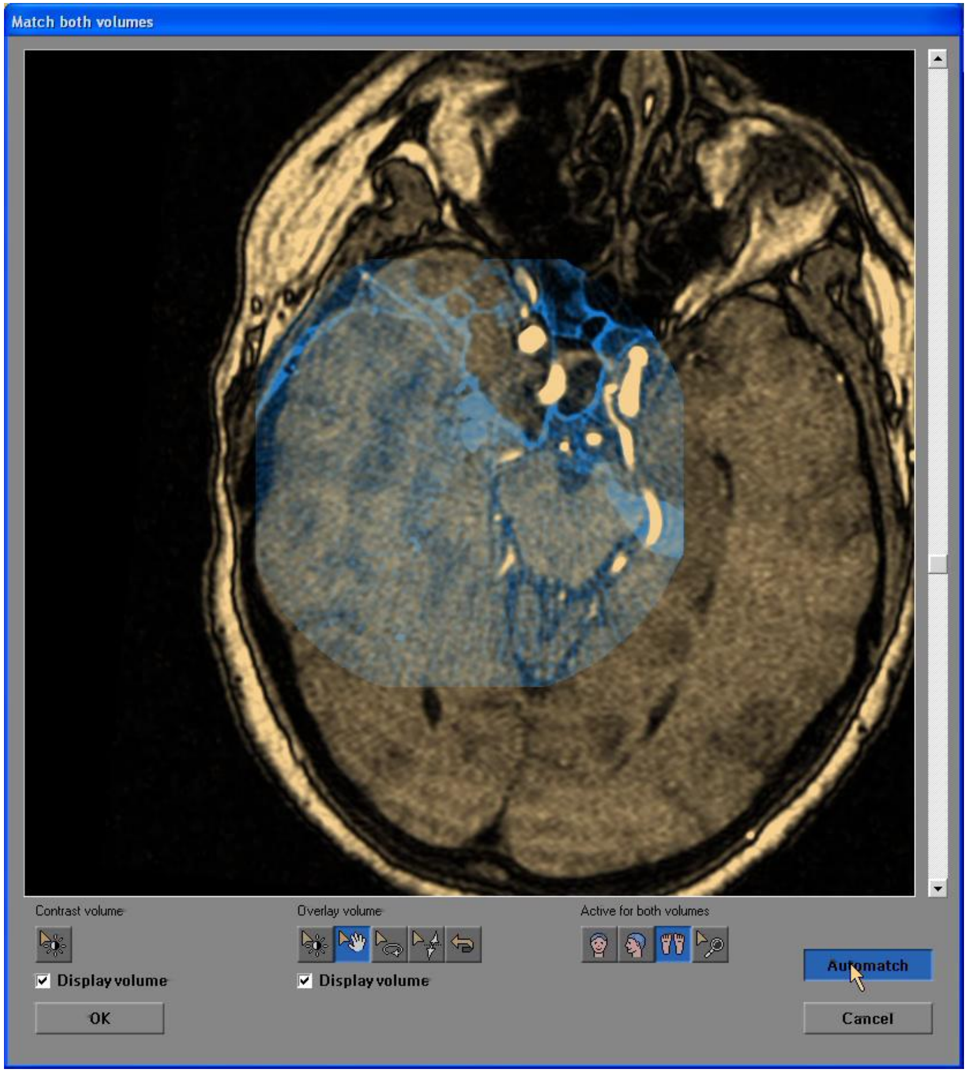


PHILIPS

3DRA – MR: CPU interpolation



3DRA – MR: GPU interpolation



PHILIPS

Elastic Registration

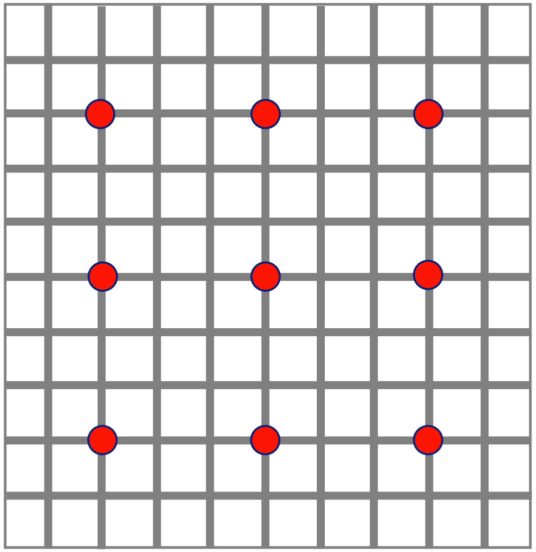
Elastic deformation

- Parameterized deformation:

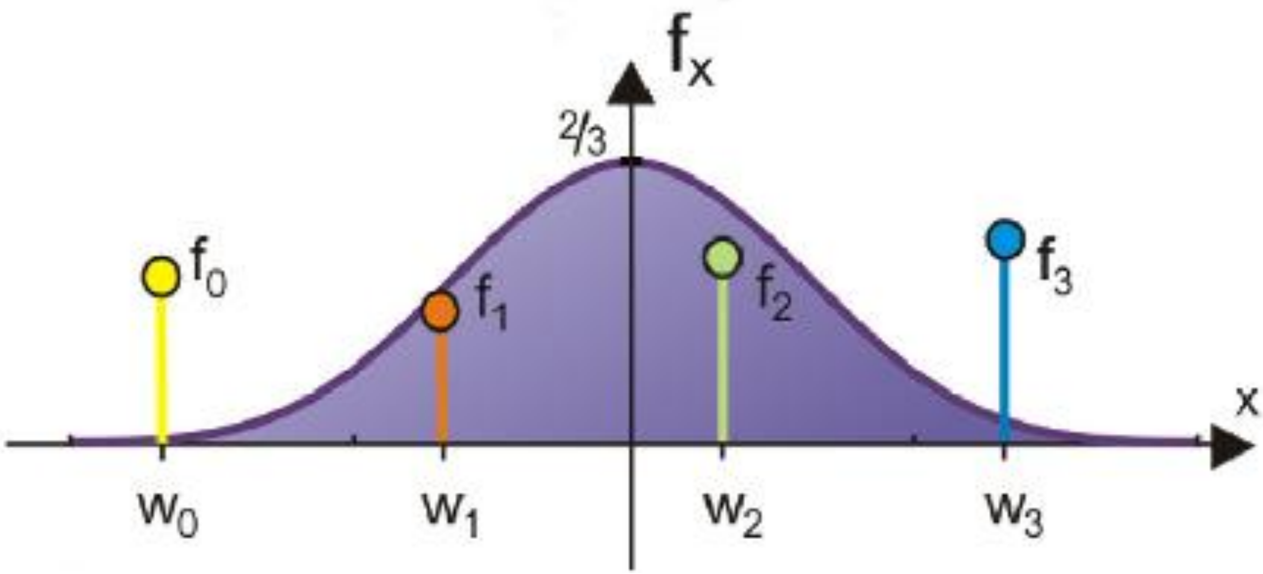
$$\mathbf{g}(\mathbf{x}) = \mathbf{x} + \sum_{\mathbf{j} \in J} \mathbf{c_j} \varphi_{\mathbf{j}}(\mathbf{x})$$

- B-spline deformation:

$$\mathbf{g}(\mathbf{x}) = \mathbf{x} + \sum_{\mathbf{j} \in I_c \subset \mathbb{Z}^N} \mathbf{c_j} \beta_{n_m}(\mathbf{x}/\mathbf{h} - \mathbf{j})$$



Cubic B-spline



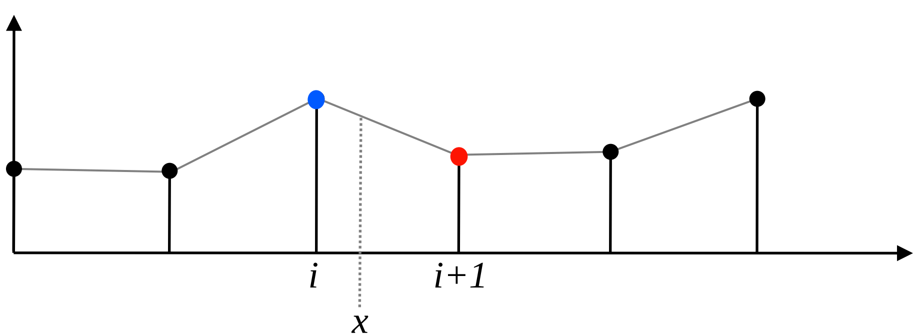
$$w_0(x) \cdot f_{i-1} + w_1(x) \cdot f_i + w_2(x) \cdot f_{i+1} + w_3(x) \cdot f_{i+2}$$

GPU linear interpolation

- Hardwired: linear interpolation is much faster than separate lookups

$$f_x = (1 - \alpha) \cdot f_i + \alpha \cdot f_{i+1}$$

$$\alpha = x - i$$



GPU Cubic Interpolation

- Compose cubic interpolation from weighted sum of linear interpolations:

$$w_0(x) \cdot f_{i-1} + w_1(x) \cdot f_i + w_2(x) \cdot f_{i+1} + w_3(x) \cdot f_{i+2}$$

$$a \cdot f_i + b \cdot f_{i+1} = (a+b) \cdot f_{i+b/(a+b)}$$

$$f_x = (1-\alpha) \cdot f_i + \alpha \cdot f_{i+1}$$

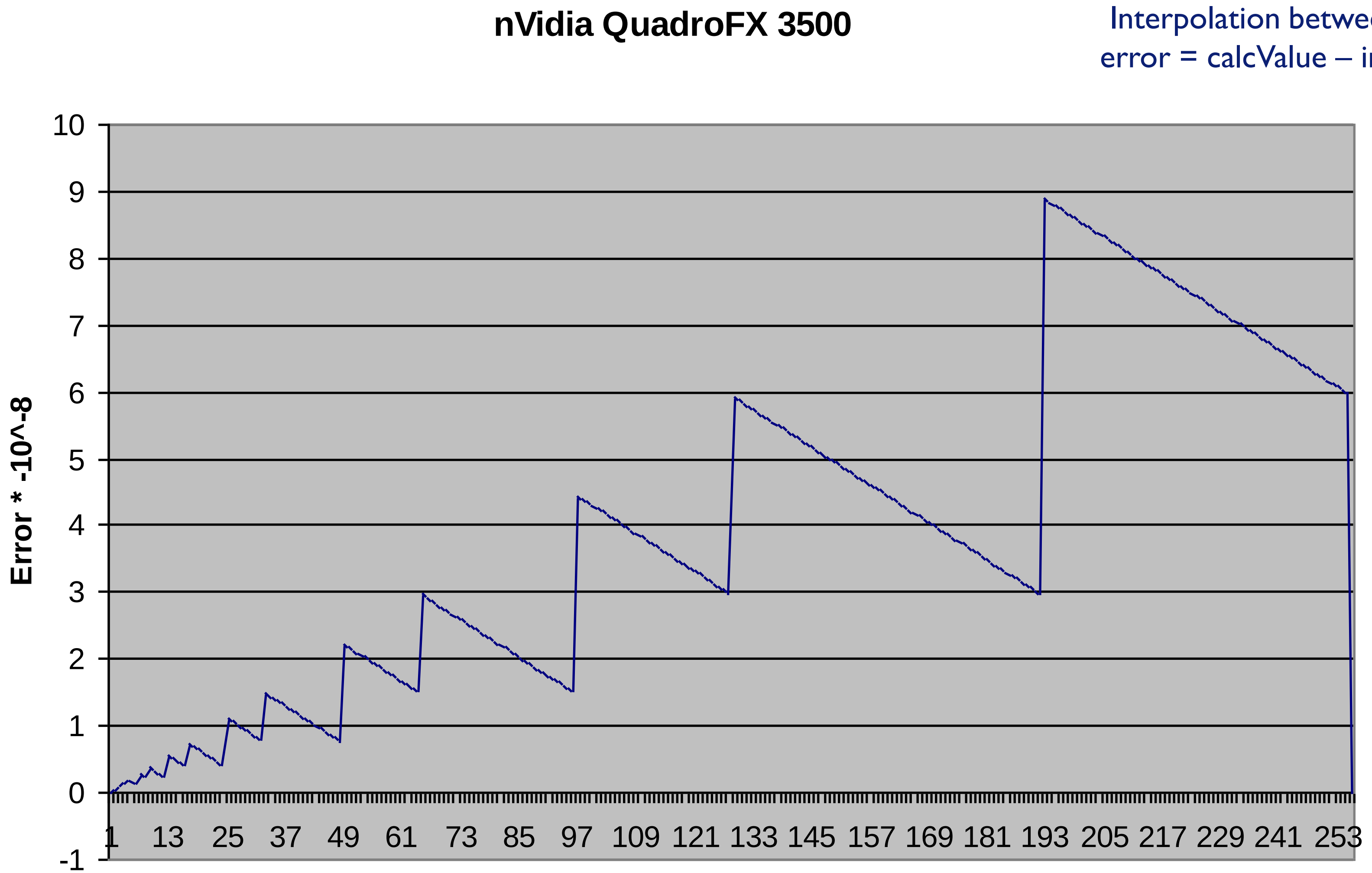
C. Sigg, M. Hadwiger, "Fast Third-Order Texture Filtering", GPU Gems 2

GPU Cubic Interpolation

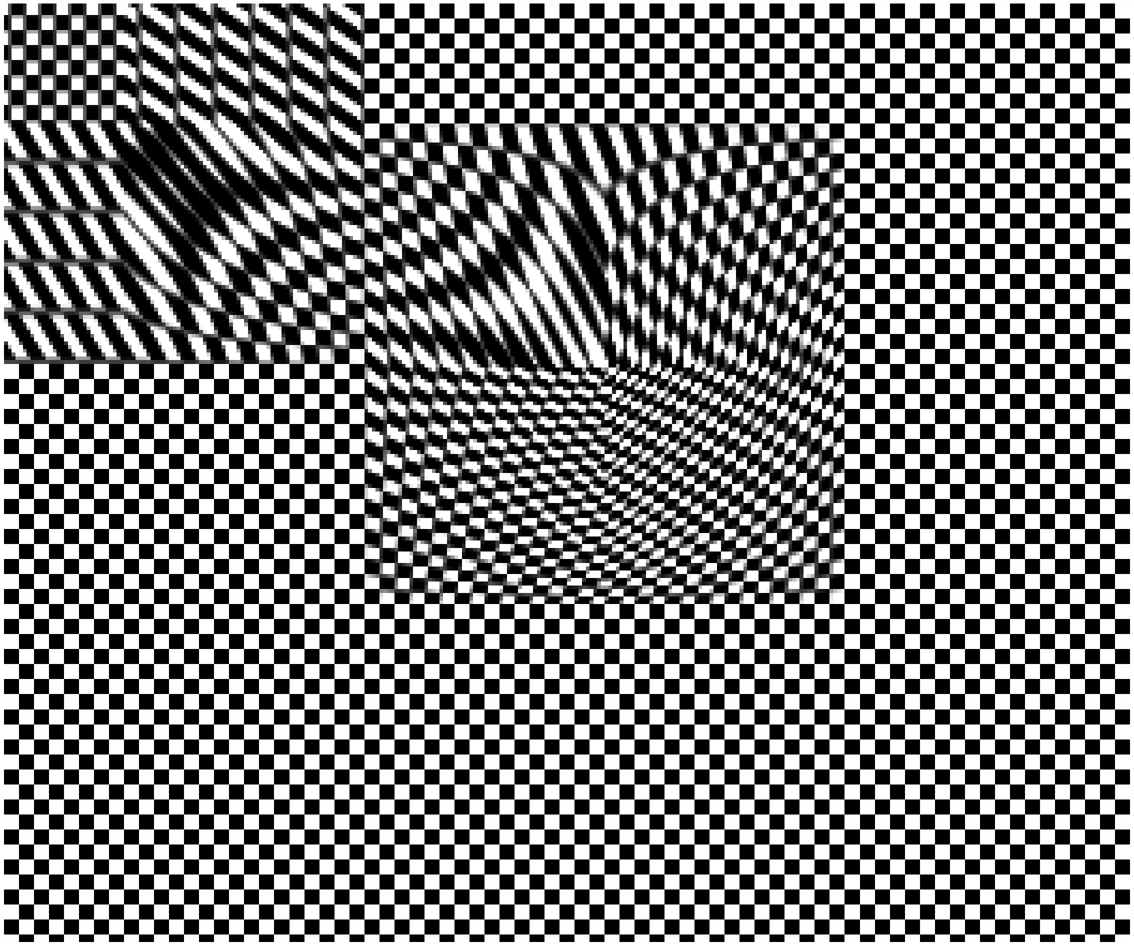
- 2D: 4 linear-interpolated lookups, instead of 16 direct lookups
- 3D: 8 linear-interpolated lookups, instead of 64 direct lookups

GPU Linear Interpolation Accuracy

Interpolation between 0 and 2^{-16}
error = calcValue – interpolValue

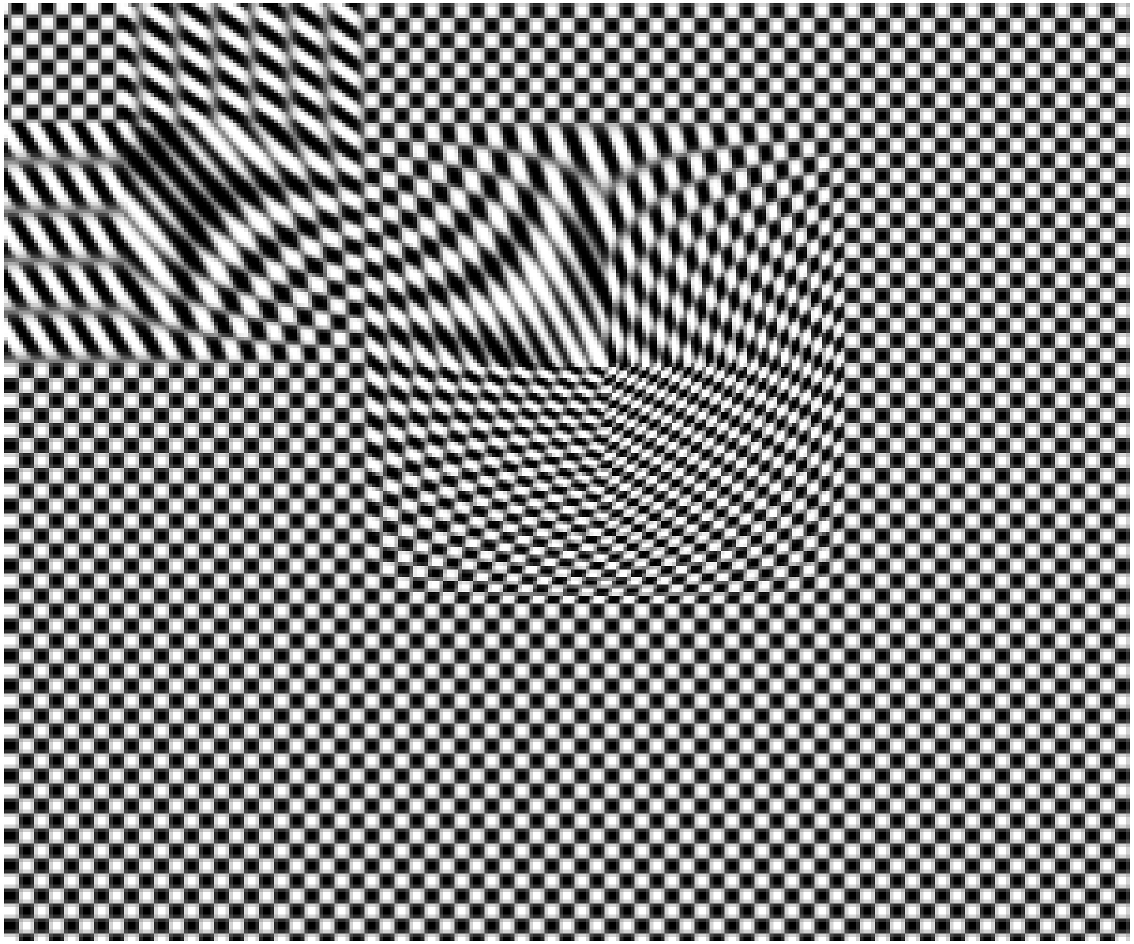


Linear deformation, linear interpolation

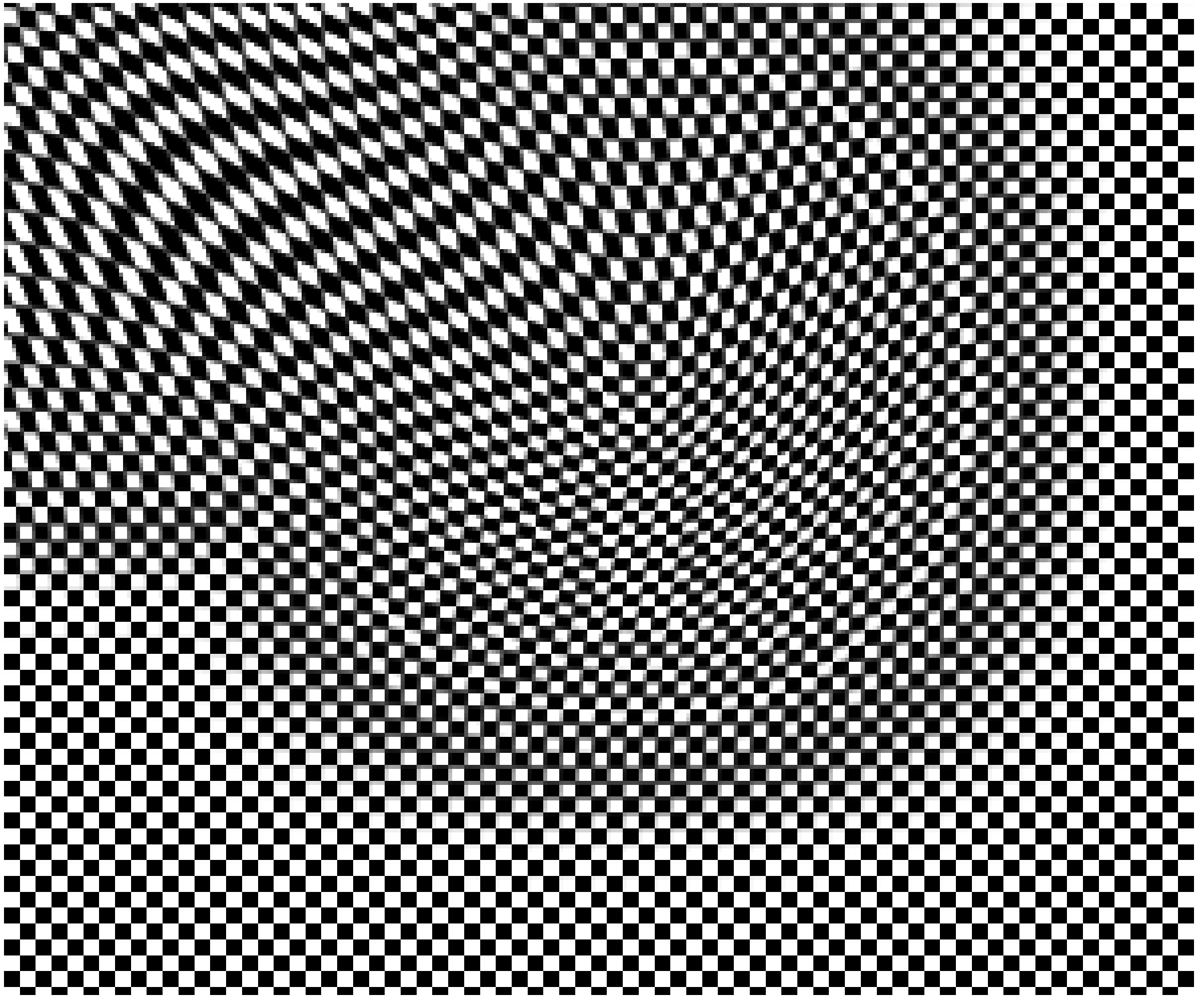


PHILIPS

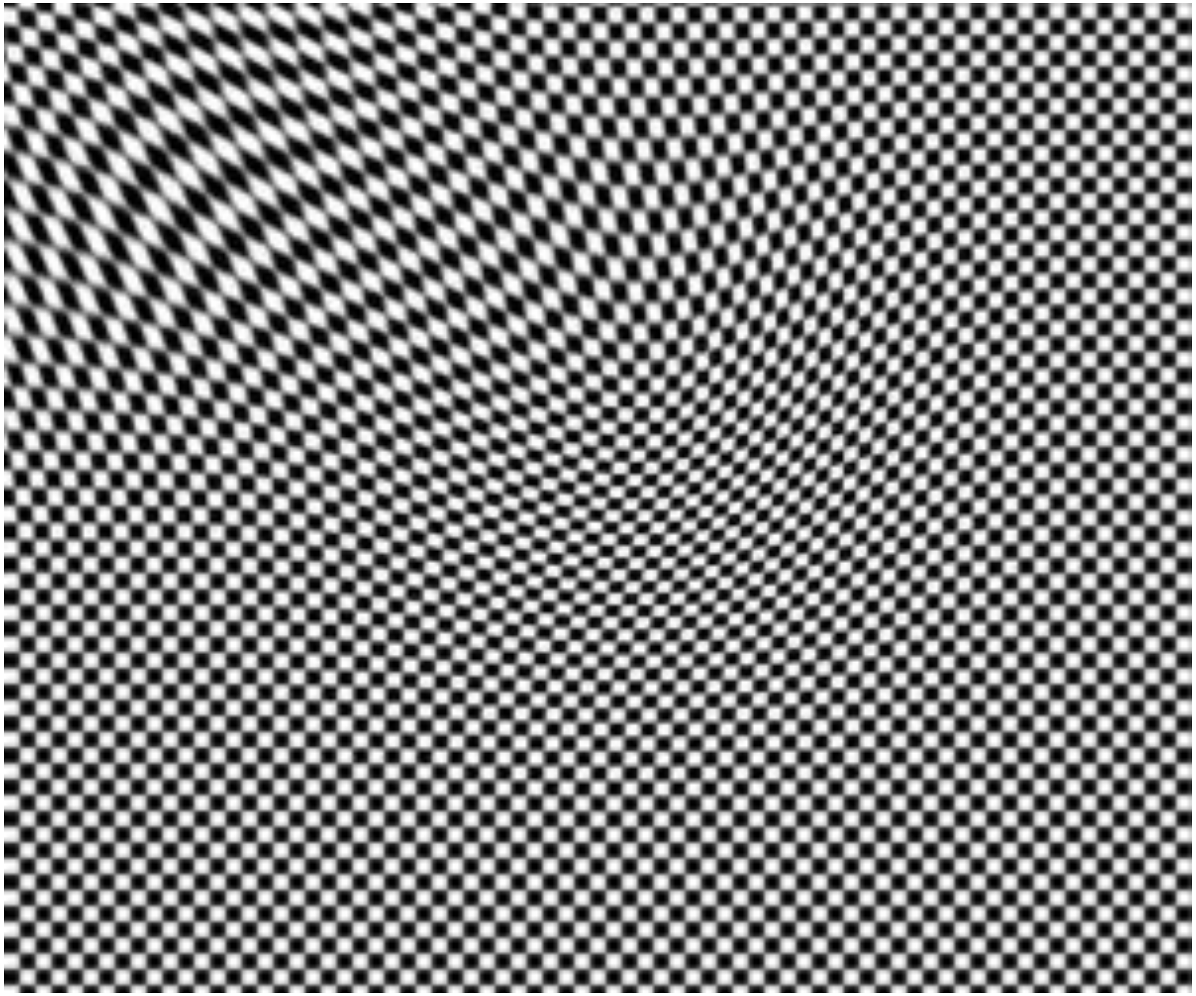
Linear deformation, cubic interpolation



Cubic deformation, linear interpolation



Cubic deformation, cubic interpolation



Optimization

- Many parameters: huge parameter space
- Solution: use derivatives like Jacobian, Hessian
- Examples: Gradient Descent, Quasi-Newton, Levenberg-Marquardt

GPU Elastic Registration Iteration

1. Generate deformed image on GPU & store to texture
2. Calculate Similarity Measure & First-Order Derivative on GPU
   - Texture with reference image
   - Texture with deformed image

First-Order Derivative of Sim. Measure

$$\frac{\partial E}{\partial c_{\mathbf{j},m}} = \frac{1}{\|I\|} \sum_{\mathbf{i} \in I_b} \frac{\partial e_{\mathbf{i}}}{\partial f_w(\mathbf{i})} \left. \frac{\partial f_t^c(\mathbf{x})}{\partial x_m} \right|_{\mathbf{x}=\mathbf{g}(\mathbf{i})} \frac{\partial g_m(\mathbf{i})}{\partial c_{\mathbf{j},m}}$$

J. Kybic, M. Unser, "Fast Parametric Elastic Image Registration"

Derivative of the Similarity Measure $\frac{\partial e_{\mathbf{i}}}{\partial f_w(\mathbf{i})}$

SSD:

$$E = \frac{1}{\|I\|} \sum_{\mathbf{i} \in I} e_{\mathbf{i}}^2 = \frac{1}{\|I\|} \sum_{\mathbf{i} \in I} \left(f_w(\mathbf{i}) - f_r(\mathbf{i})\right)^2$$

$$= \frac{1}{\|I\|} \sum_{\mathbf{i} \in I} \left(f_t^c(\mathbf{g}(\mathbf{i})) - f_r(\mathbf{i})\right)^2$$

$$\frac{\partial e_{\mathbf{i}}}{\partial f_w(\mathbf{i})} = 2\left(f_w(\mathbf{i}) - f_r(\mathbf{i})\right)$$

Derivative of the Deformed Image

$$\left.\frac{\partial f_t^c(\mathbf{x})}{\partial x_m}\right|_{\mathbf{x}=\mathbf{g}(\mathbf{i})}$$

- Sobel operator to calculate gradients:

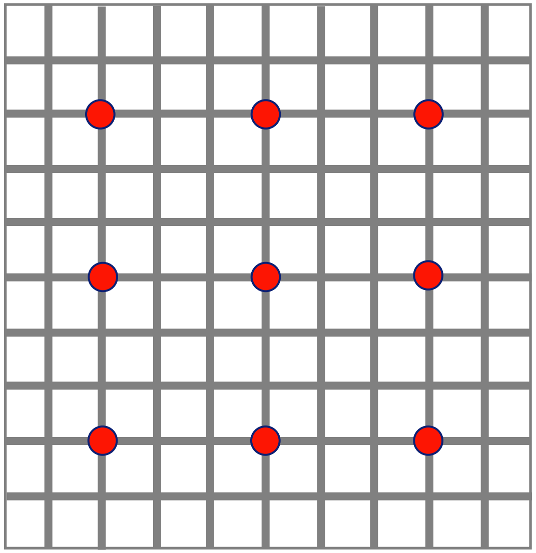
-1	0	1
-4	0	4
-1	0	1

1	4	1
0	0	0
-1	-4	-1

Derivative of the Control Points $\frac{\partial g_m(\mathbf{i})}{\partial c_{\mathbf{j},m}}$

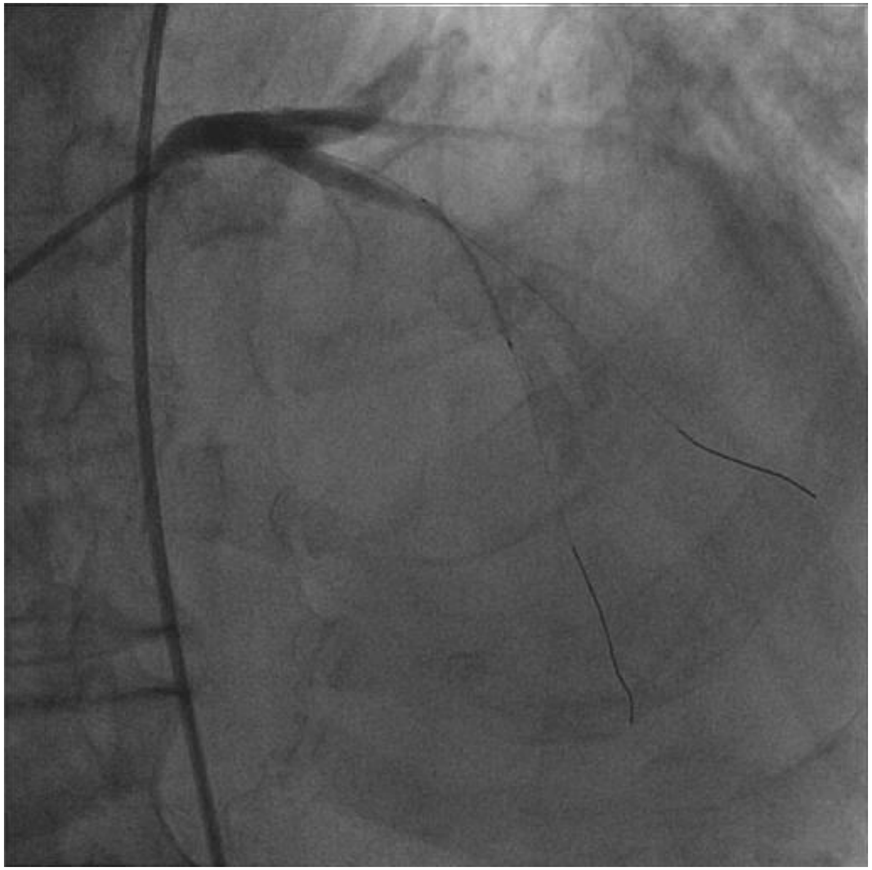
$$\mathbf{g}(\mathbf{x}) = \mathbf{x} + \sum_{\mathbf{j} \in I_c \subset \mathbb{Z}^N} \mathbf{c}_{\mathbf{j}} \beta_{n_m}(\mathbf{x}/\mathbf{h} - \mathbf{j})$$

$$\frac{\partial g_m}{\partial c_{\mathbf{j},m}} = \beta_{n_m}(\mathbf{x}/\mathbf{h} - \mathbf{j})$$



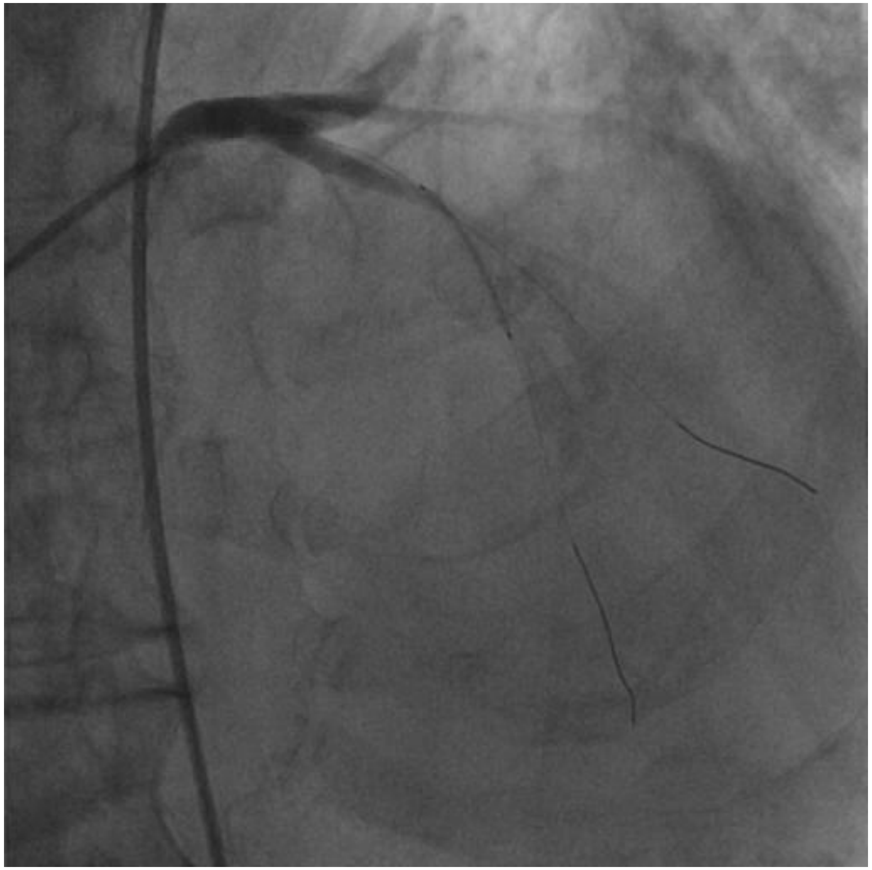
- Constant
- B-spline: separatable kernel of fixed size

Original Fluoroscopy Sequence

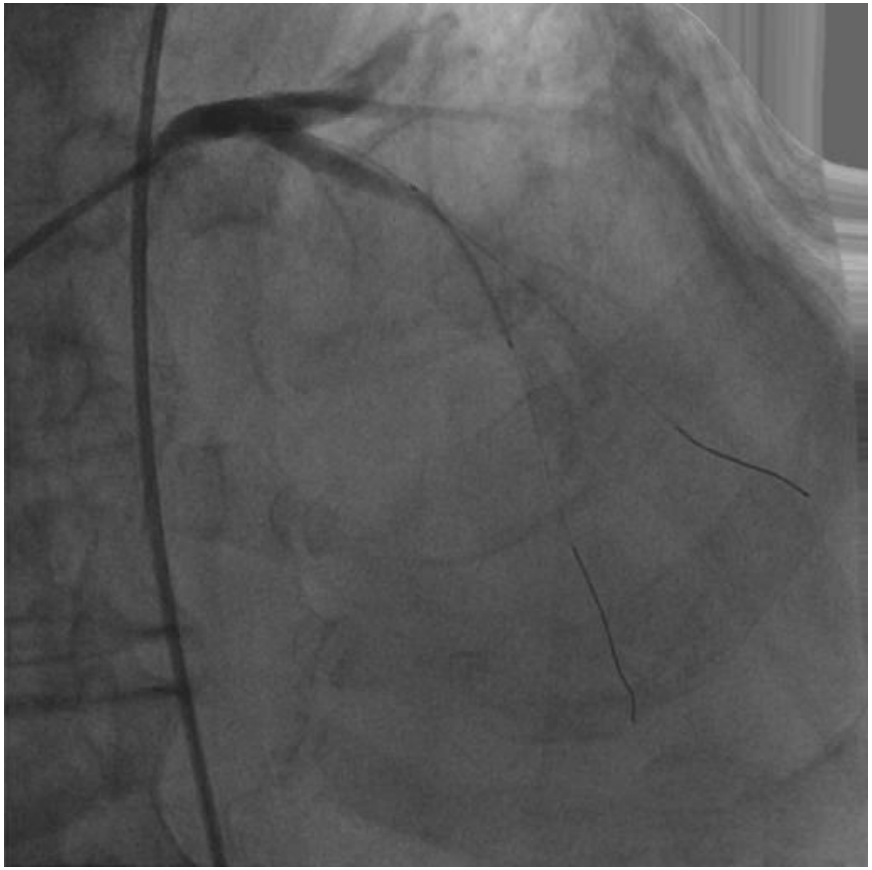


PHILIPS

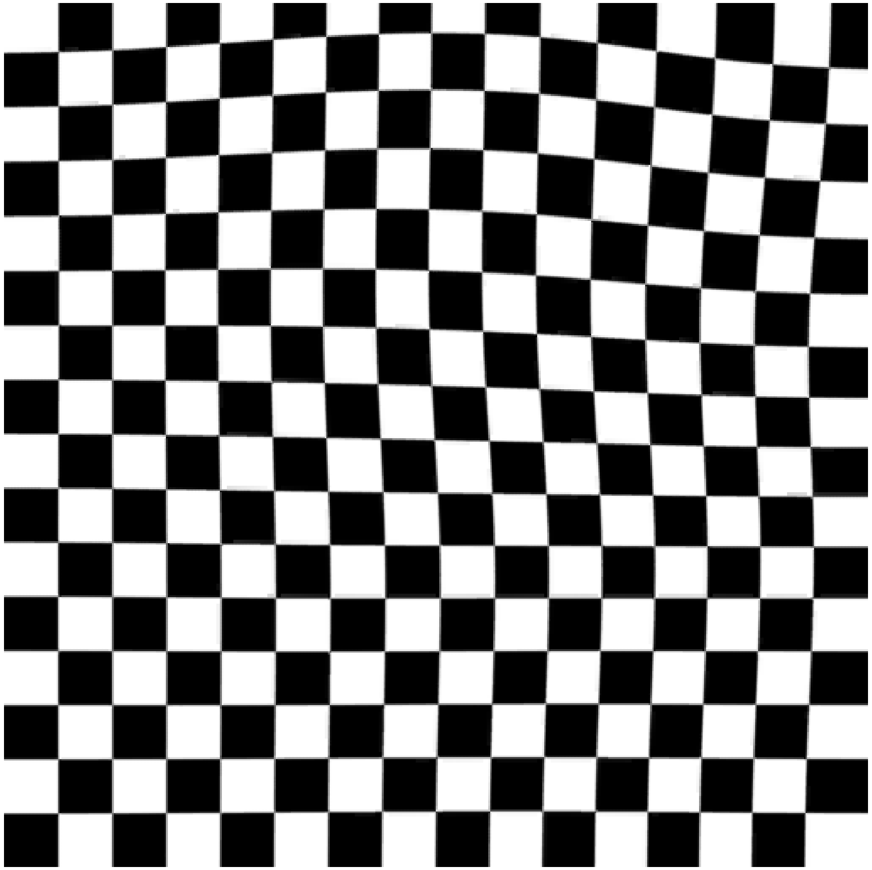
2 * 2 Control Points



8 * 8 Control Points



Deformation Field



GPU Elastic Registration

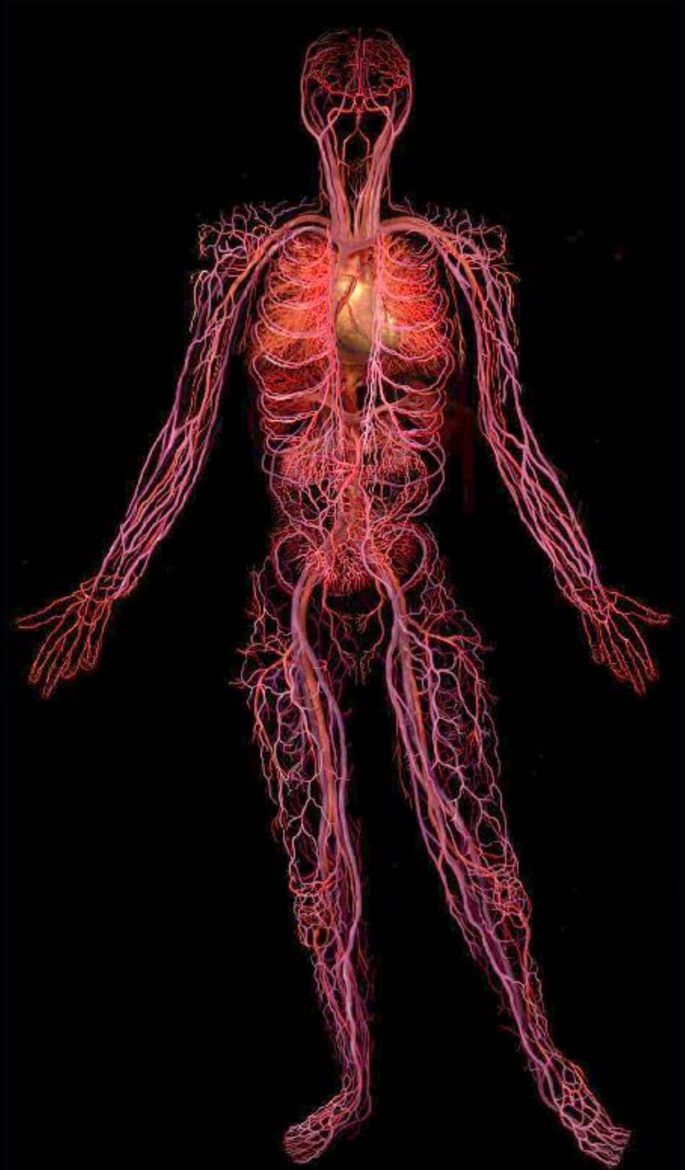
- 40 images: Quasi Newton: 16 seconds
- Gradient Descent: 63 seconds

PHILIPS

More information

- http://www.healthcare.philips.com
- http://dannyruijters.nl/phd.html

Questions?



PHILIPS