Seminar on Statistical Anatomic Models,
Registration, and Reconstruction
2011/1/27

GPU-accelerated Registration/Reconstruction Toolkit and its Application to 2D/3D registration

Yoshito Otake

Acknowledgements

Advisers

- Russell H. Taylor
- Mehran Armand
- Peter Kazanzides
- Jeffrey H. Siewerdsen

Collaborators

JHU/CISST

- Ehsan Basafa
- Blake Lucas
- Wen Liu
- Ben X. Kang

APL

- Robert Armiger
- Michael D. Kutzer
- Ryan Murphy

Bayview

- Simon C Mears
- Stephen Belkoff

I-STAR lab

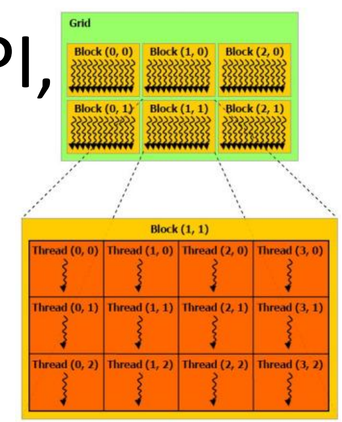
- J. Web Stayman
- Wojtek Zbijewski
- Sebastian Schafer
- Junghoon Lee
- Ali Uneri
- Sureerat Reaungamornrat
- Sajendra Nithiananthan
- Daniel Mirota

Outline

1. Development of the GPU reg./recon. toolkit
   - Motivation
   - Purpose
   - Implementation detail
   - Example applications
   - Future work
2. An application of the toolkit: Intensity-based rigid 2D/3D registration
   - Implementation and experimental evaluation

Motivations

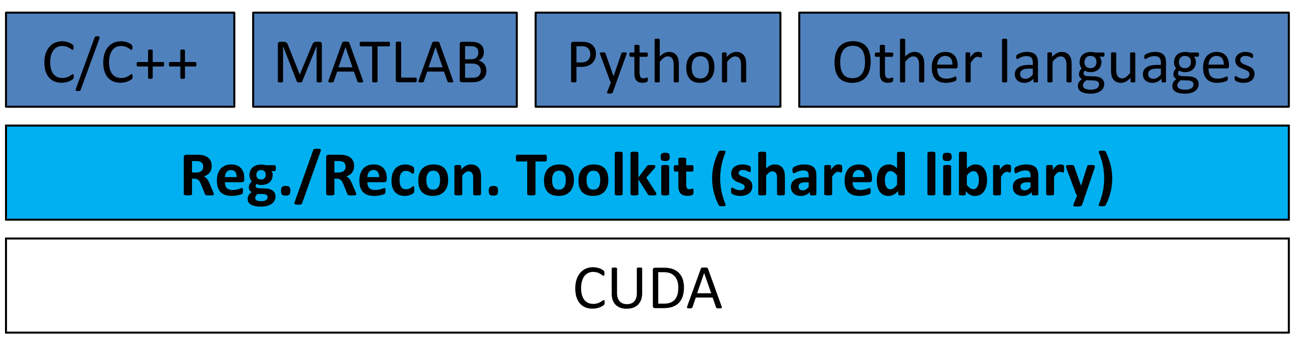
- Many components of registration/reconstruction researches overlap and usually computationally intensive.
  - e.g. DRR generation (forward projection), back projection, resampling, etc.
- Why don't we implement those components on GPU with an unified interface?
- Matlab-friendliness is important for research
- Shared library, cross-platform, intuitive API, modular design, multi-GPU support are preferable...



Cited from
NVIDIA CUDA Programming Guide

Purpose

- To develop a cross-platform shared toolkit with GPU-acceleration, which helps registration/reconstruction researches.



Software architecture

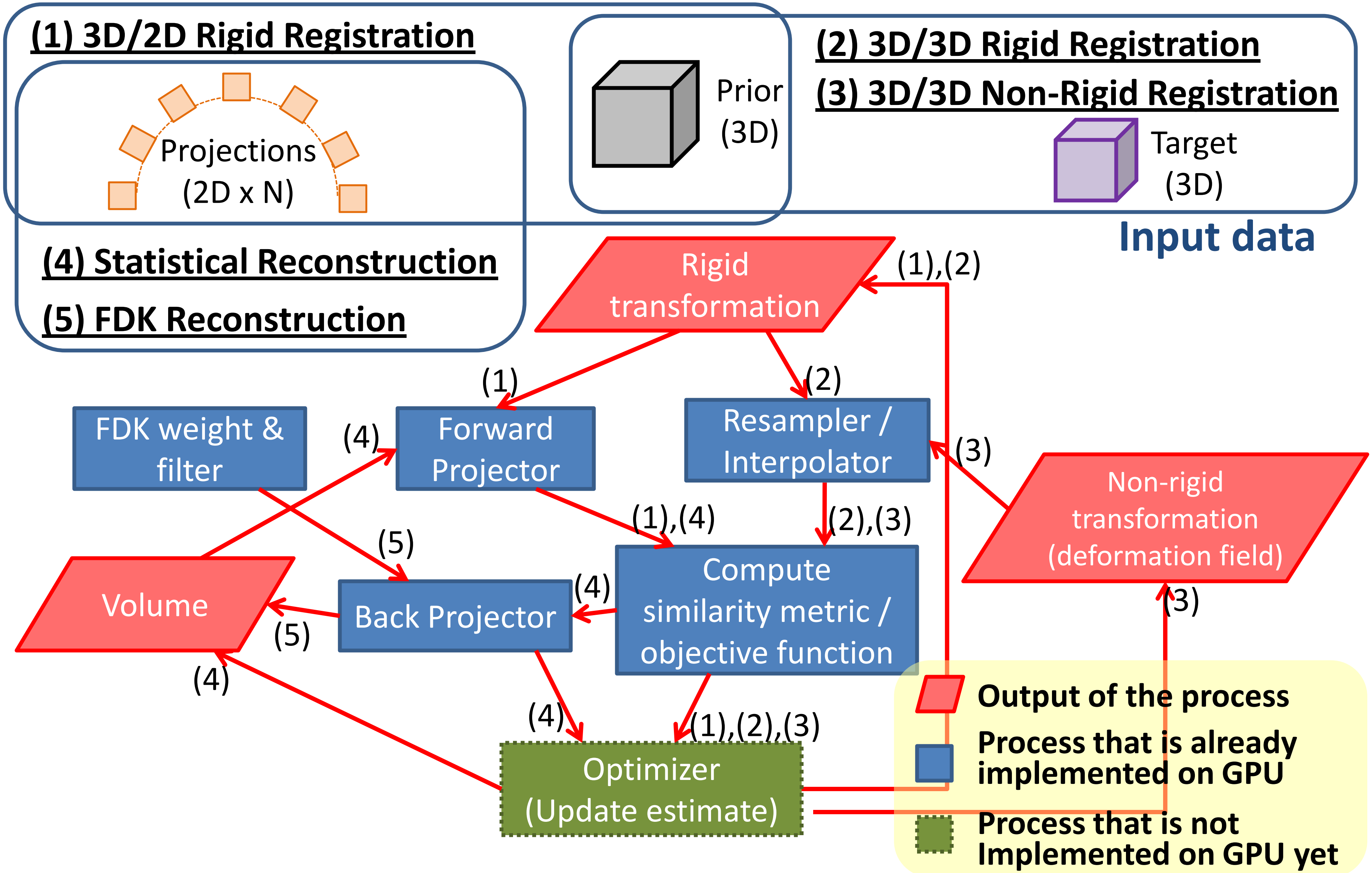
Overview of the Toolkit

- One shared library (named "FBProjector.dll")
- Almost all code in the library are implemented from scratch
  - Exceptions
    - FDK filtering – collaboration with Dr. Sebastian Schafer
    - MI computation on GPU – open source (written by Ramtin Shams)
- Source files and sample code
  - https://svn.lcsr.jhu.edu/yotake2/C++/CUDA_Programs/FBProjectorDll
- Dependency
  - CUDA Toolkit (http://developer.nvidia.com/object/cuda_3_2_toolkit_rc.html)
- Supported platform
  - Developed and tested mainly on Windows, but it should work on other platform, probably…
  - A little more details are on I-STAR wiki
    - https://trac.lcsr.jhu.edu/istar/wiki/Projectors

Key components in the Toolkit

- Already implemented on the GPU
  - Forward projector (DRR generator)
  - Back projector (Voxel-driven & Ray-driven)
  - Resampler / Interpolator
  - Similarity measure computation (not optimized yet)
    - Mutual information (MI), Normalized mutual information (NMI), Gradient information (GI), Gradient correlation (GC), Normalized cross correlation (NCC), Mean square distance (MSD).
  - FDK filtering
- To be implemented… (currently Matlab is used)
  - Optimizer

Application scenarios



Application scenarios

	2D/3D rigid reg.	3D/3D rigid reg.	3D/3D non-rigid reg.	Stat. recon.	FDK recon.
Input	2D x N (fixed) + 3D (moving)	3D (fixed) + 3D (moving)	3D (fixed) + 3D (moving)	2D x N	2D x N
Optimization process	Forward project	Resample	Apply D (resample)	Forward project	
	Compute similarity measure	Compute similarity measure	Compute gradient	Compute (curvature?)	FDK weighting & filtering
				Back project	Back project
	Update T	Update T	Update D	Update V	
Output	T	T	D	V	V

N: Number of projection images
T: Rigid transformation
V: Volume (3D)
D: Deformation field

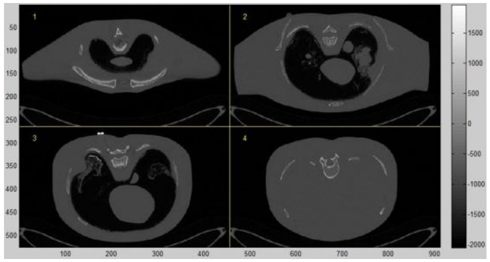
Example MATLAB interface

```
% Initialize dll instance
dllHandle = FBProjectorDll_wrapper( 'FBProjectorDll' );
dllHandle.CreateFBProjectorInstance(imageSize);
dllHandle.InitializeProjectorParameters(1, ... % intensity window
      0,  ... % intensity level
      1.0, ... % step size (ignored in siddon mode)
      1 )     % 1: siddon mode, 0: trilinear interpolation mode
```

Initialization

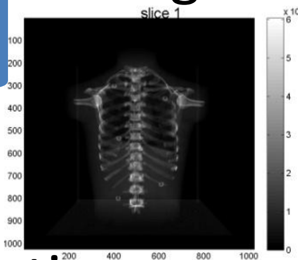
```
% set volume data
dllHandle.InitializeInputData_CT( volume );
dllHandle.SetVolumeInfo(volumeSize, voxelSize);
```

Volume data setting



```
% X-ray projection geometry setting
dllHandle.InitializeProjectionParametersArray(numProj);
for i=1:100
      projectionGeometry = struct('DetectorFrame', DetectorFrame(:,:,i) , ...
                        'SourcePosition', SourcePosition(:,i), ...
                        'IsPerspective', 1, 'FOV', fov);
      dllHandle.SetProjectionParameter_objectOriented(i-1, projectionGeometry);
end
```

Geometry setting

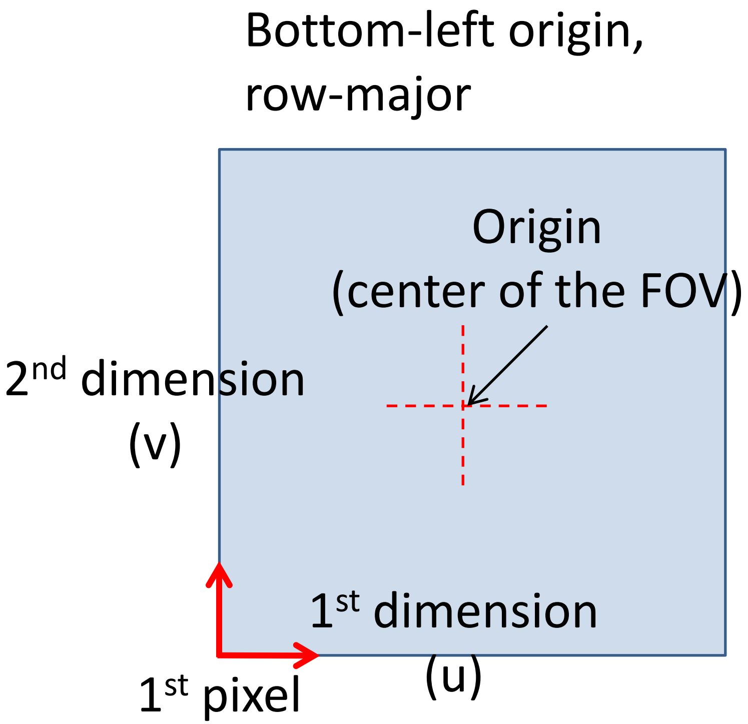



```
% Compute DRR
dllHandle.ForwardProjection( imageBufferPtr );
DRRs = reshape( get(imageBuffer.Data, 'value'), [imageSize numProj] ) ;
```

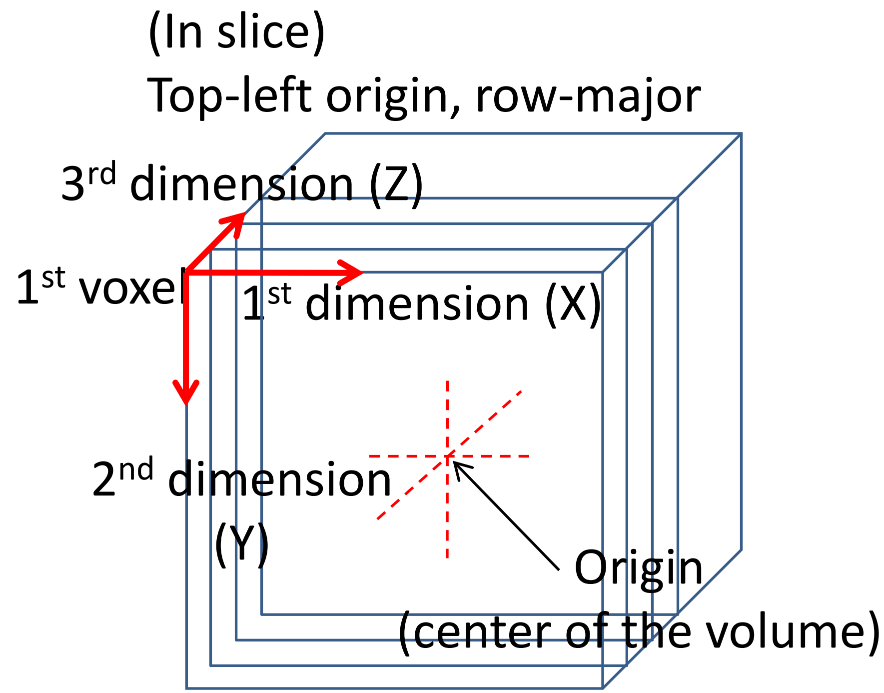
DRR generation

Pixel/voxel order used in the library

For projection (2D)



For volume (3D)



Example C/C++ interface

```
// Initialize dll instance
FBProjectorInstance fbProjector;
CreateFBProjectorInstance(&fbProjector, WIDTH, HEIGHT, false, err_str);
SetWindowLevel(fbProjector, WINDOW, LEVEL, err_str);
SetStepSize(fbProjector, 1.0, err_str);
SetIsSiddon(fbProjector, true, err_str);
```

Initialization

```
// set volume data
float *volume; // see the previous slide for the voxel order
InitializeInputData_CT(fbProjector, volume, w, h, d, 1, err_str);
SetVolumeInfo(fbProjector, w, h, d, vox_w, vox_h, vox_d, err_str);
```

Volume data setting

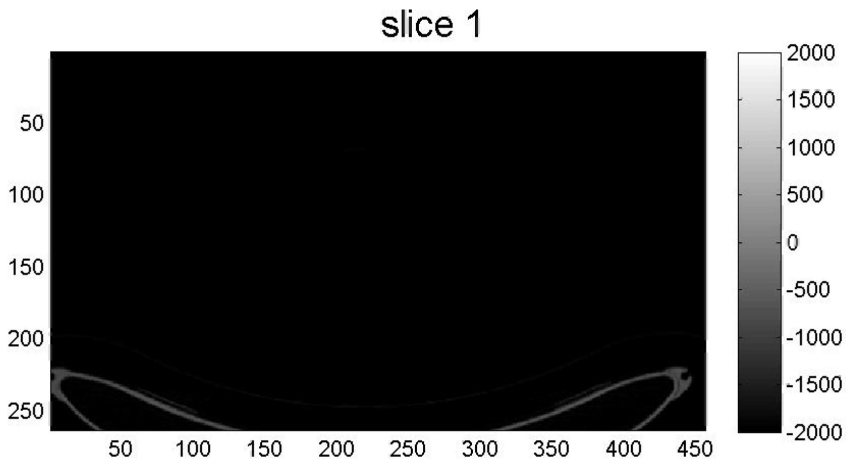
```
// X-ray projection geometry setting
InitializeProjectionParametersArray(fbProjector, NUM_PROJECTION, err_str);
ProjectionParameters_cameraOriented projectionGeometry;
for(int i=0;i<NUM_PROJECTION;i++){
   // Geometry setting on projectionGeometry
   SetProjectionParameter_cameraOriented(fbProjector, i, projectionGeometry, err_str);
}
```

Geometry setting

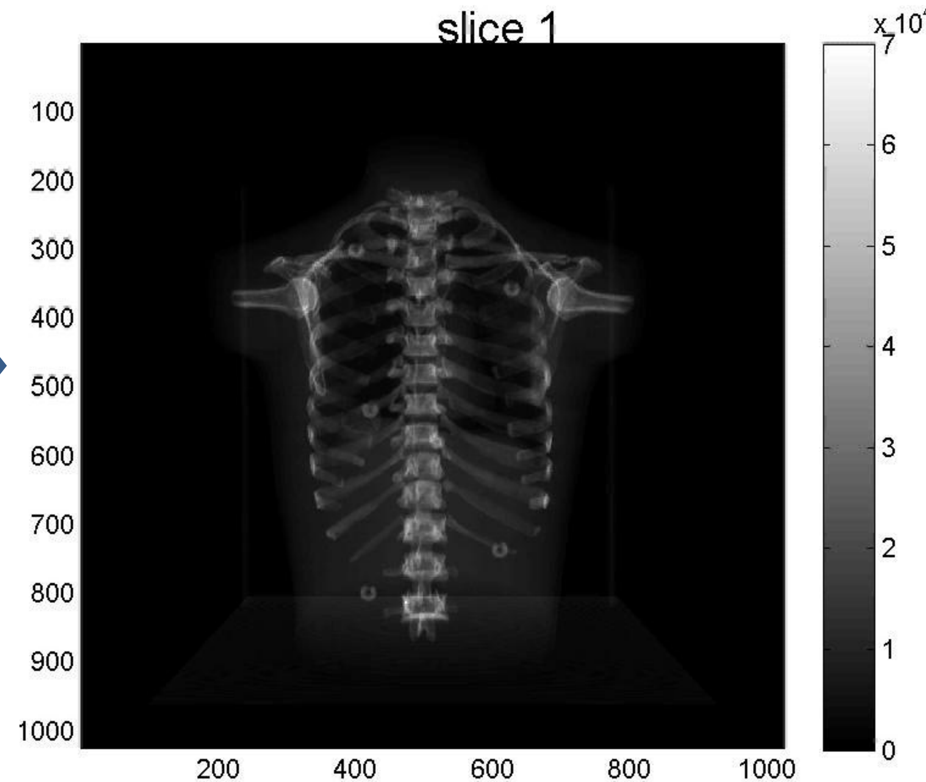
```
// Compute DRR
PackedDynamicFloatMatrix DRRs;
 InitializePackedDynamicFloatMatrix(&DRRs);
ForwardProjection(fbProjector, &DRRs, err_str);
```

DRR generation

Performance example



456×263×494 (1.0mm³)



1024×1024×360

12.17 sec (~30fps)

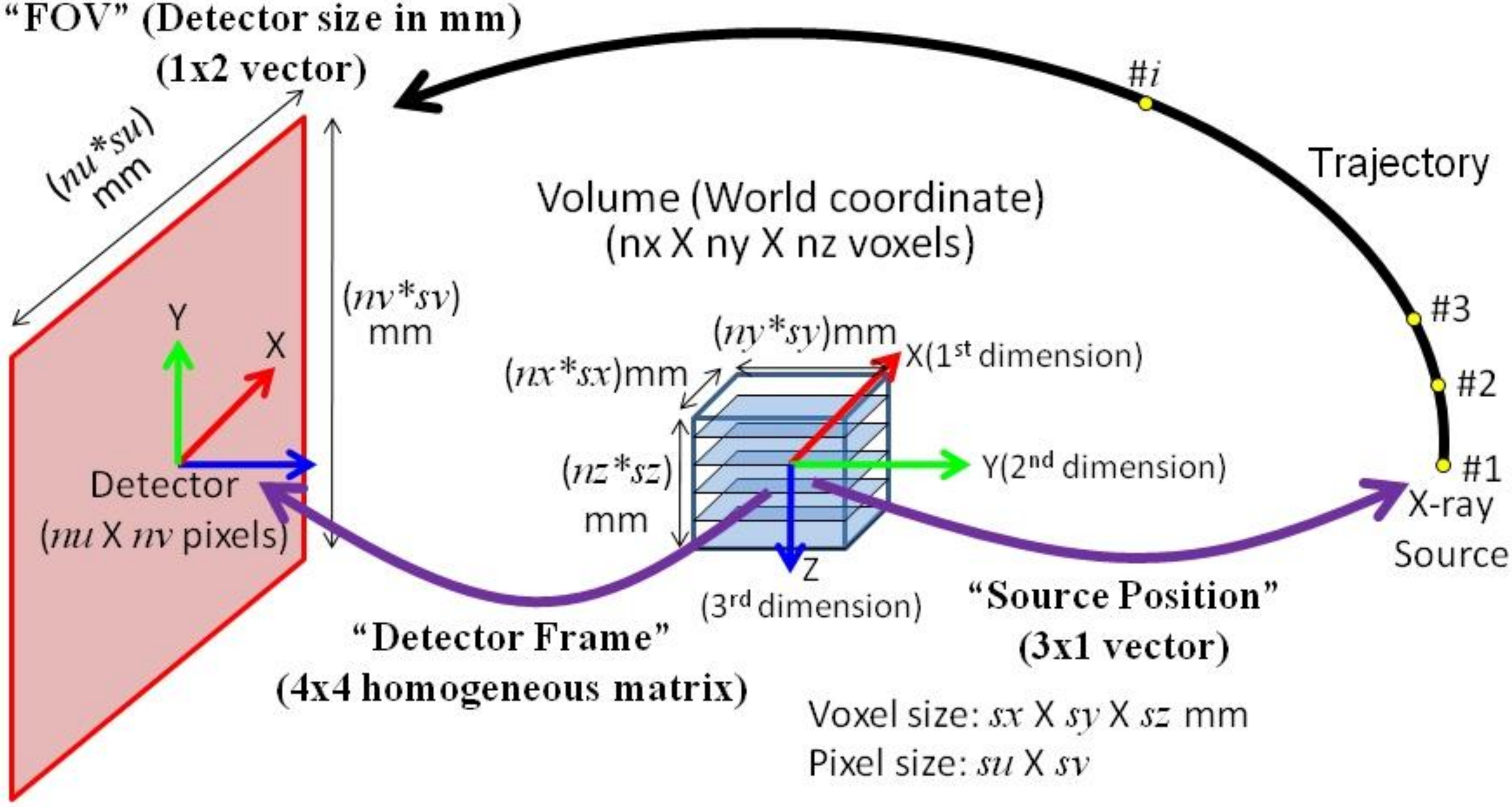
(1.0mm step length, non-Siddon mode)

Hardware specifications

Operating System	Windows Vista 64 bit
Processor type	Intel® Core™ 2 Duo
CPU clock frequency (GHz)	2.66
Graphics card type	NVIDIA® Quadro® FX3700M
No. processors core	128
Memory bandwidth (GB/s)	51.2
Graphics memory (MB)	1024

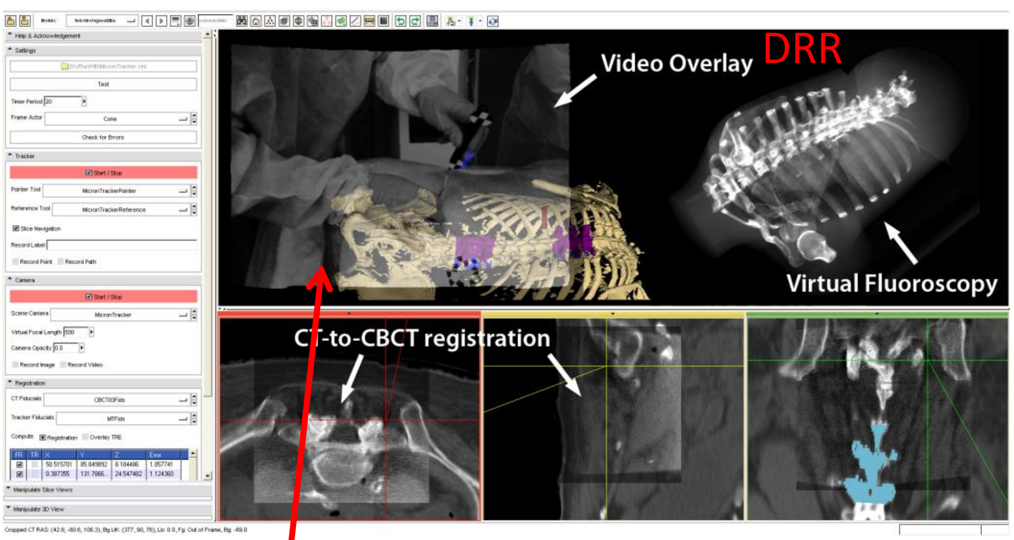
Demo

Definition of the coordinate systems

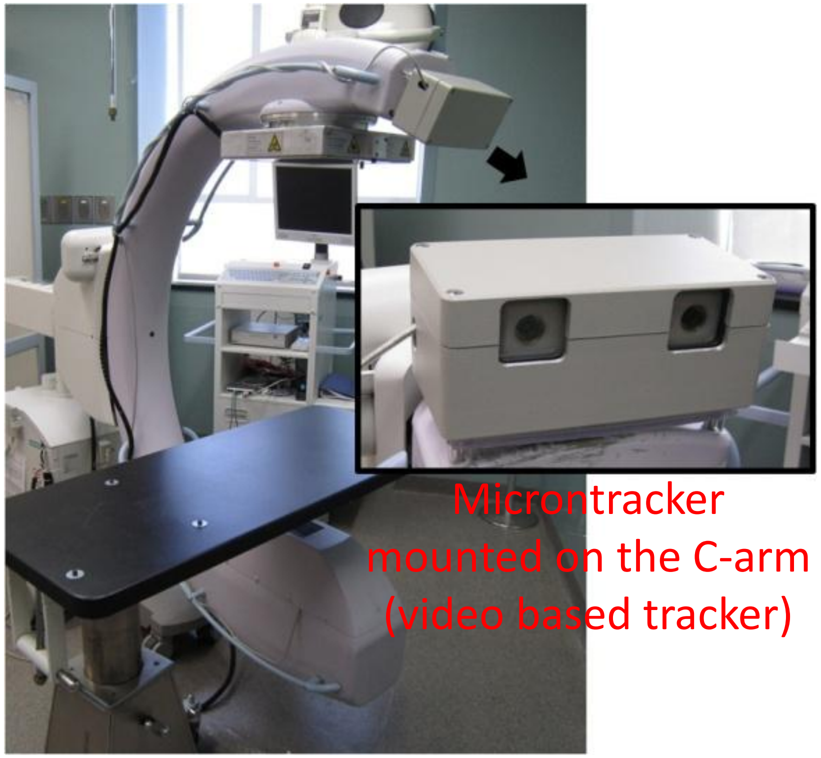


Example application (1): DRR generation as an image-guidance tool

- Functions used
  - Forward projector
- Integrated into Slicer 3
- Real-time (~30fps)



Video image captured by Microntracker

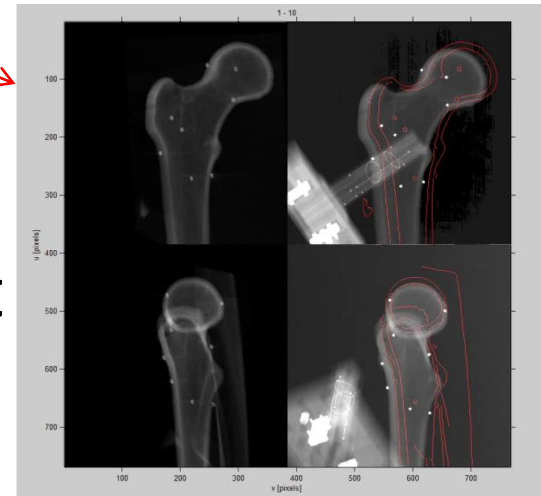


Tracker-on-C

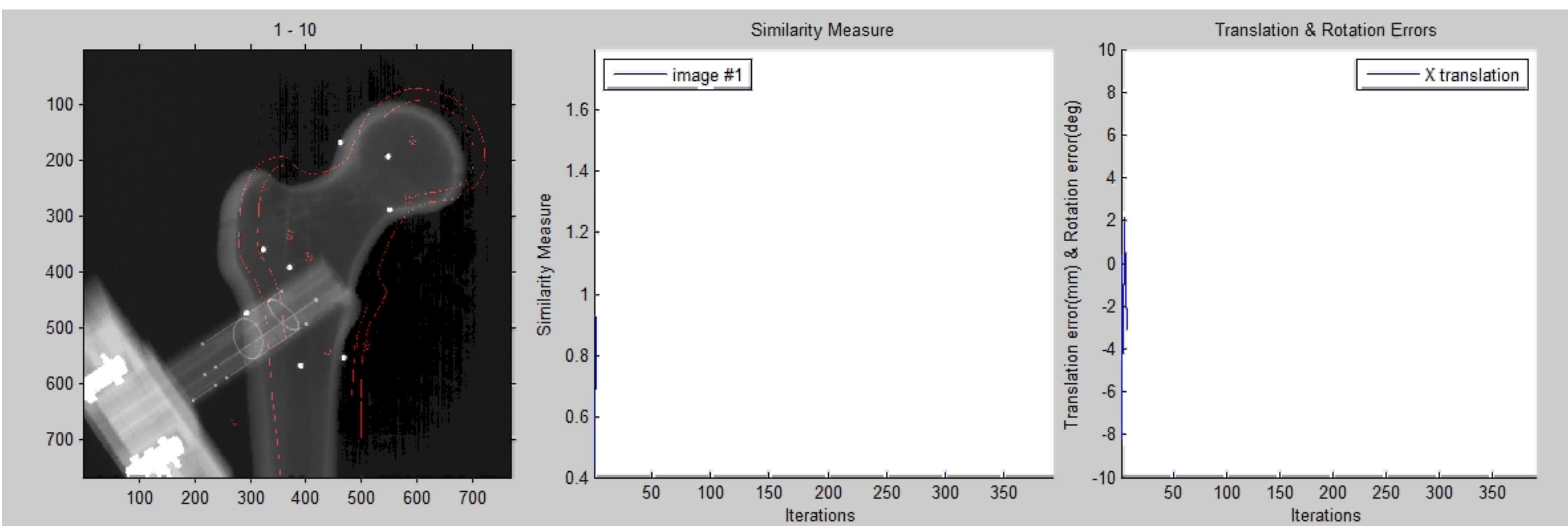
Example application (2): Intensity-based 2D/3D rigid registration

- Functions used
  - Forward projector, similarity measure computation
- Function other than the toolkit
  - Optimizer (in Matlab): downhill simplex

Floating image



Fixed image with edges of the floating image

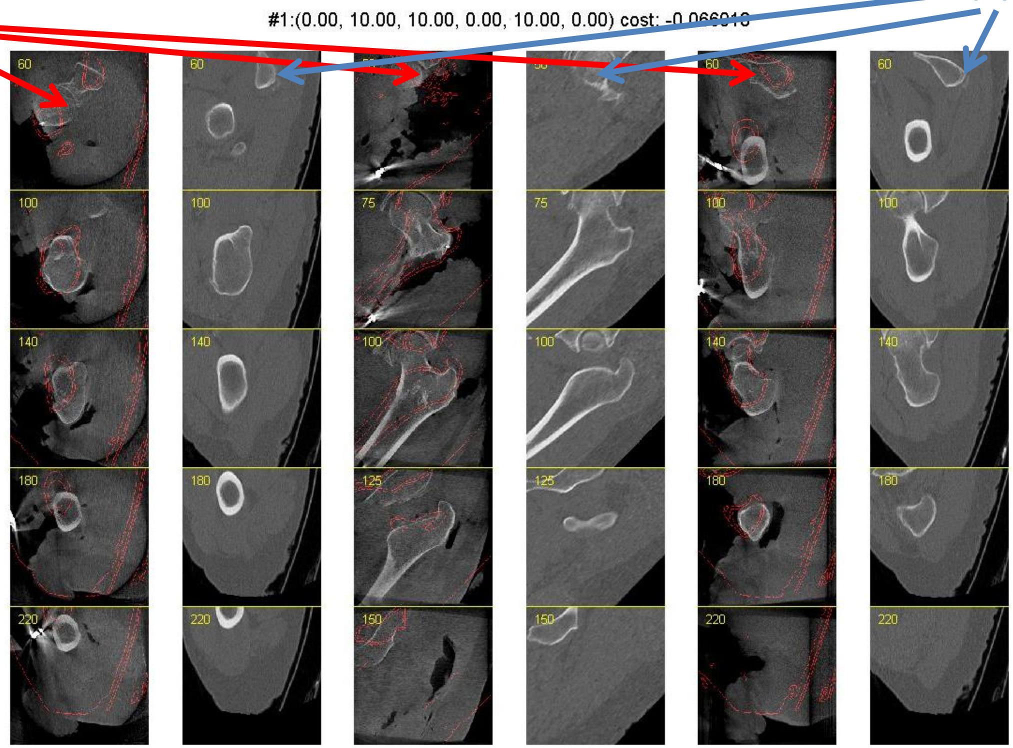


Example application (3):
Intensity-based 3D/3D rigid registration

- Functions used
  - Resampler, similarity measure computation (MI)
- Function other than the toolkit
  - Optimizer (in Matlab): downhill simplex

Fixed volume

Floating volume

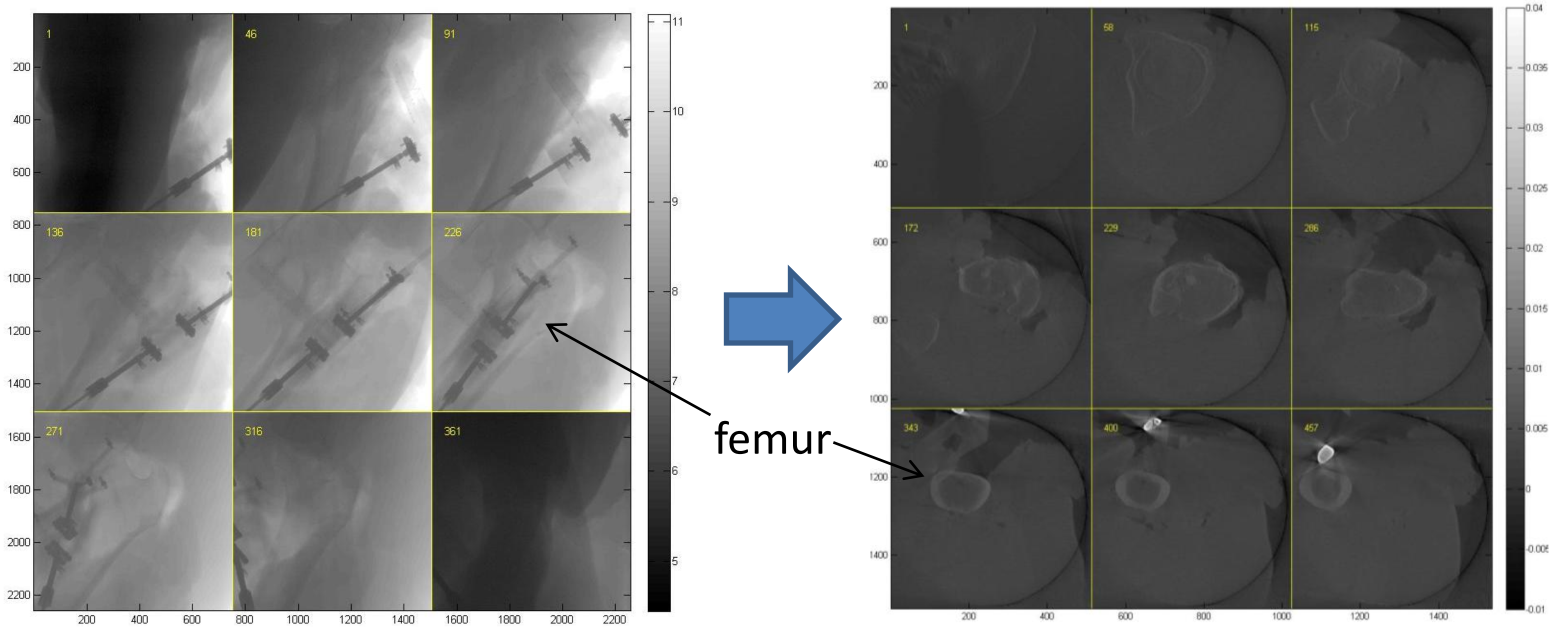


Axial slices Coronal slices Sagittal slices

Example application (4): FDK reconstruction

- Functions used
  - FDK filtering, voxel-driven back projector
- Integrated into istar3D (cone-beam reconstruction platform developed in I-STAR lab)

Performance example
Input: 768x768, 360 projections
Output: 512x512x512 volume
GPU: nVidia Quadro FX3700
About 62 seconds



Input X-ray projections (Log corrected)

Output volume

Example application (5): Statistical reconstruction

- Functions used
  - Siddon-based forward projector, ray-driven back projector
- Integrated into Matlab software (written by Dr. Stayman)

Input projection images (6 images) Reconstructed volume

An example of sparse sample (6 samples) reconstruction

Current status

- Implementation of the basic functions has been completed and almost ready for "version 1.0" release.
- Some code are not clean and need to reorganize to make things consistent.
- Need to organize test datasets (ground truth reconstruction) to check the functionalities.

Future works

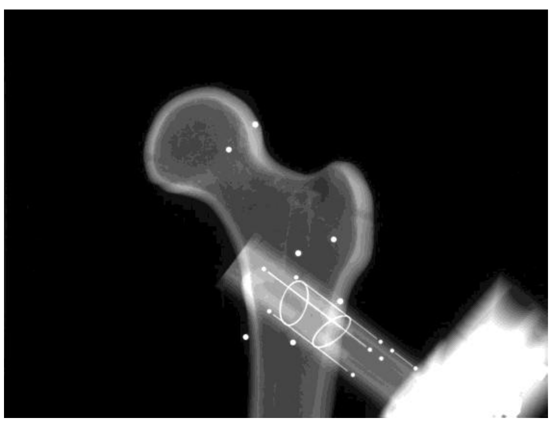
- Unit test & debug
- Multi-GPU support
  - partly supported in the current version
- Re-organize (clean up) API
- New functions/applications
  - Polyenergetic projector (‘segmented volume’ projector)
  - Depth map computation for video/CT registration
  - Statistical atlas using voxel-based statistics with GPU-acceleration (fast instance generation, 2D/3D reg., etc.)
  - Connect the ‘real-time’ X-ray imaging with robot (da Vinci, ROBODOC)?
  - etc. (any suggestions are very welcomed.)

Additional features of the toolkit

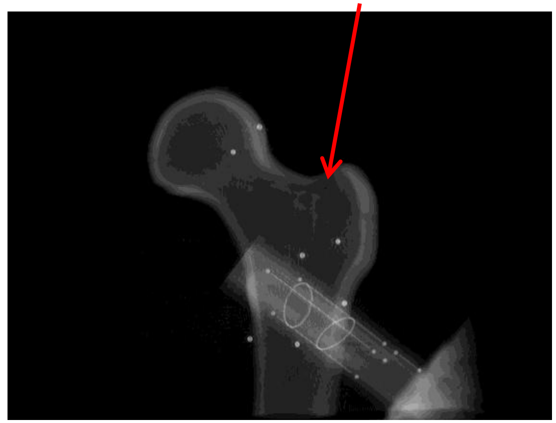
- Surface Projector
  - Generate projection images of surface (polygonal mesh) model using VTK
  - Generate multiple images at the same time
  - Can be used from MATLAB, C/C++, etc. along with DRR generator

CT volume data

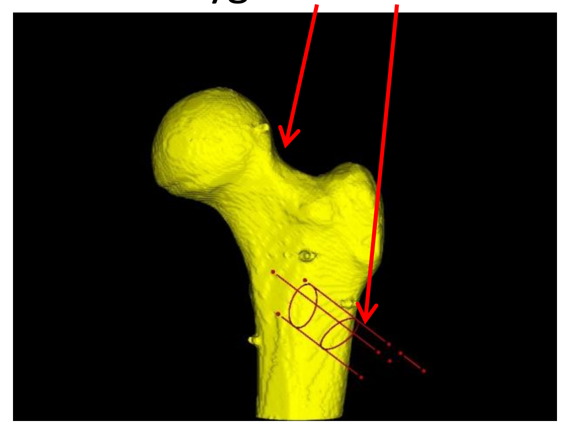
Polygon mesh model



X-ray image



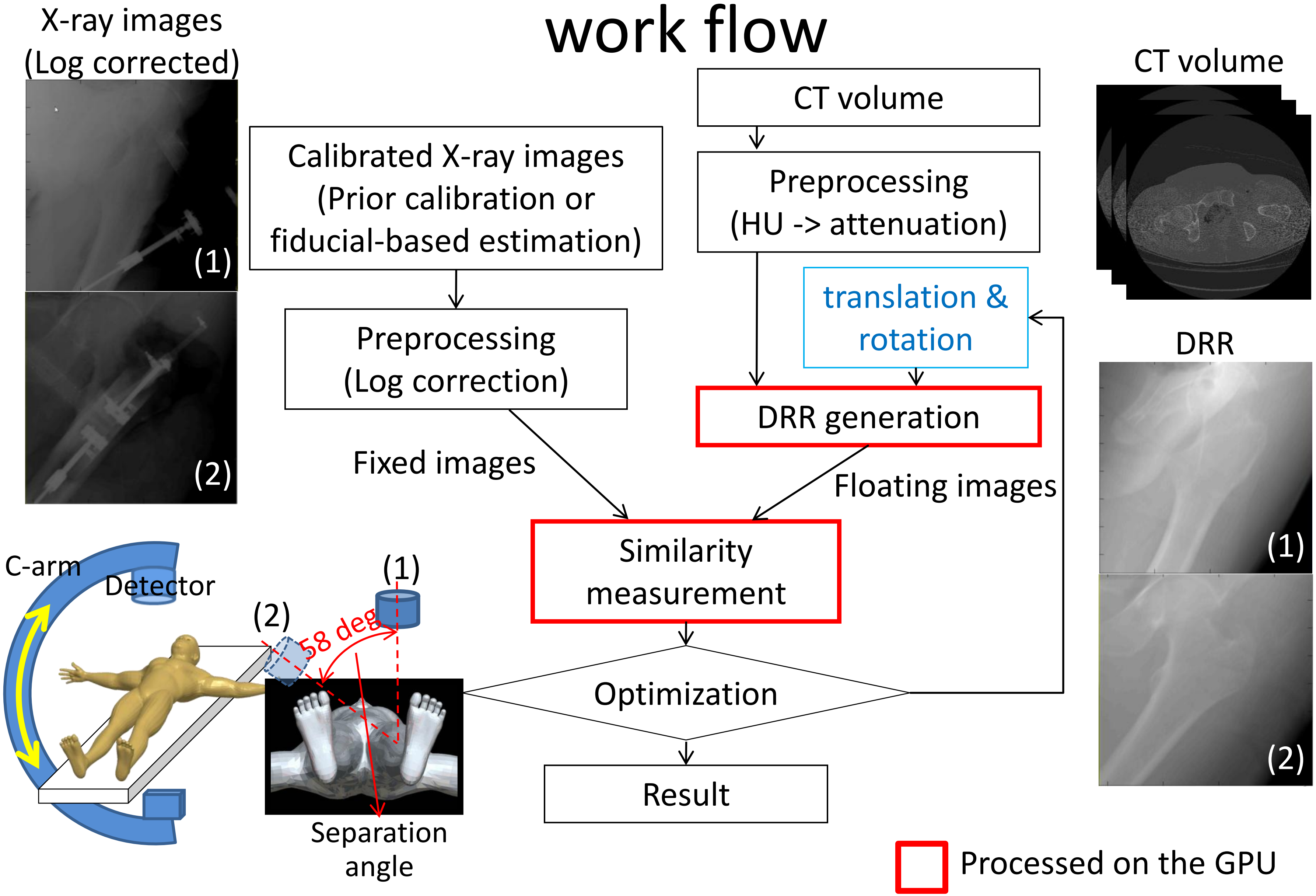
DRR



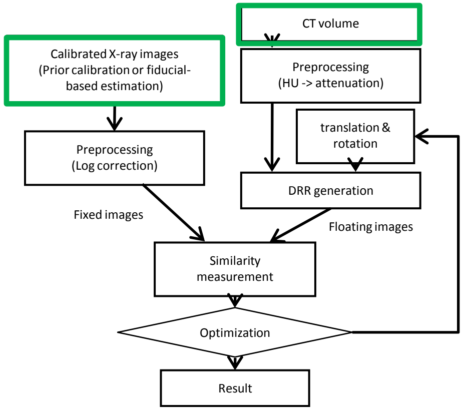
Surface model projection (rendering)

Implementation and experimental validation of an intensity-based rigid 2D/3D registration

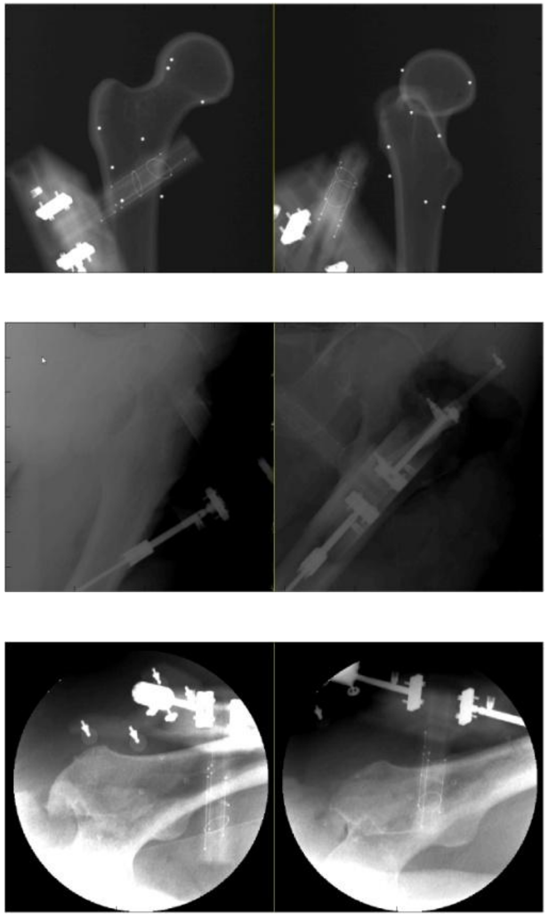
Intensity-based rigid 2D/3D registration: work flow



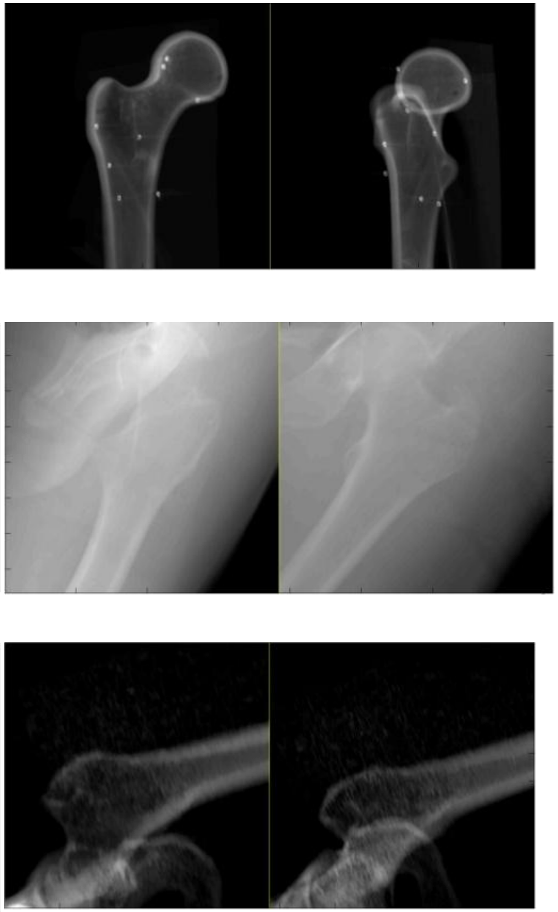
Three Datasets for evaluation



1. Sawbone
   - Flat-panel
   - without soft tissues
2. Cadaver #1
   - Flat-panel C-arm
   - with soft tissues
3. Cadaver #2
   - Image intensifier
   - with soft tissues



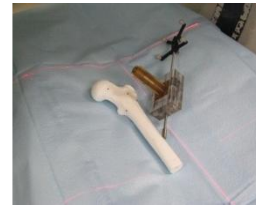
Fixed images (Log corrected)

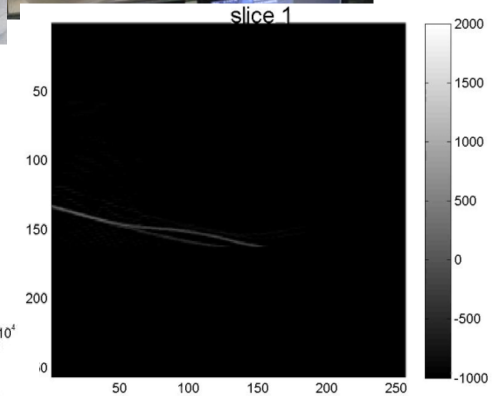


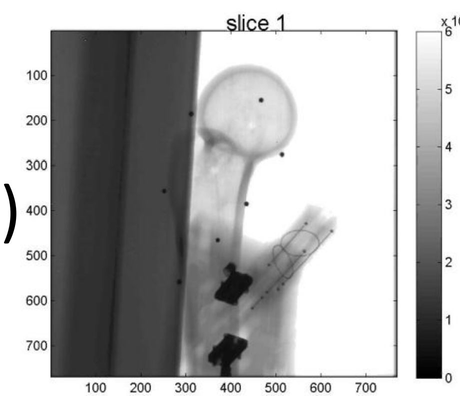
Floating images

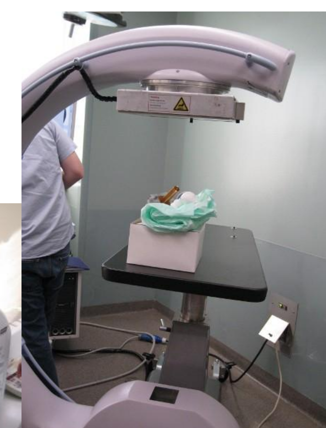
Dataset 1: Sawbone phantom

- Preop CT
  - Diagnostic CT at Bayview Medical Center (2009/3/3)
  - 256x256x256 voxel (cropped)
  - 0.564x0.564x0.6 mm/voxel
  - 135 kVp, 250 mAs
- Intraop X-ray
  - C14 at MISTIC (2010/3/8)
  - 768x768 pixel
  - 0.388x0.388 mm/pixel
  - 100 kVp, 5.8 mA

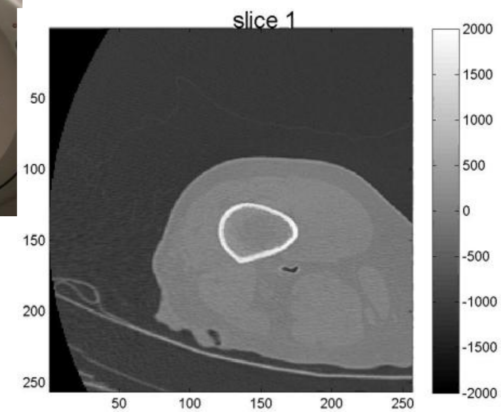




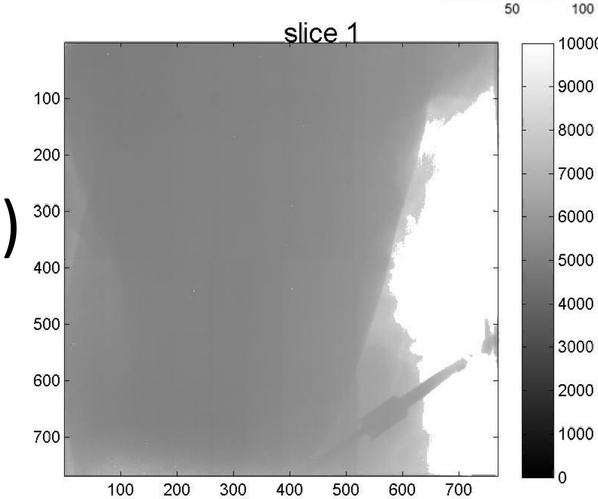


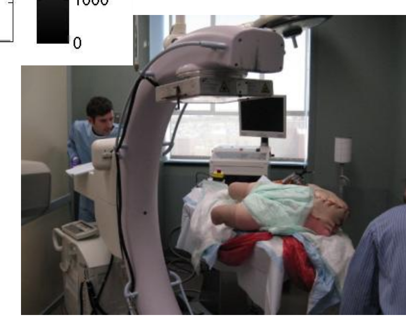
Dataset 2: Cadaver #1



- Preop CT
  - Diagnostic CT at Bayview Medical Center (2010/4/29)
  - 256x256x256 voxel (cropped)
  - 0.782x0.782x2 mm/voxel
  - 135 kVp, 250 mAs

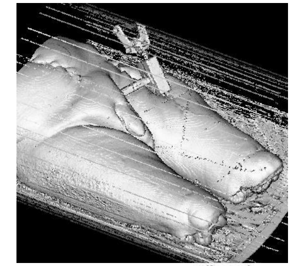


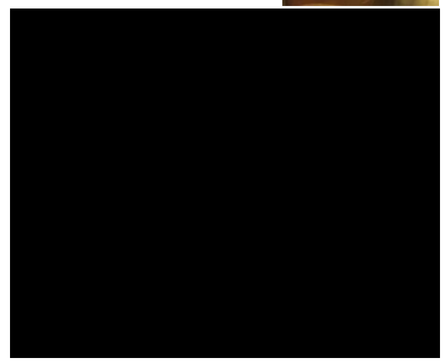
- Intraop X-ray
  - C14 at MISTIC (2010/4/30)
  - 768x768 pixel
  - 0.388x0.388 mm/pixel
  - 120 kVp, 5.2 mA
  - After cement injection

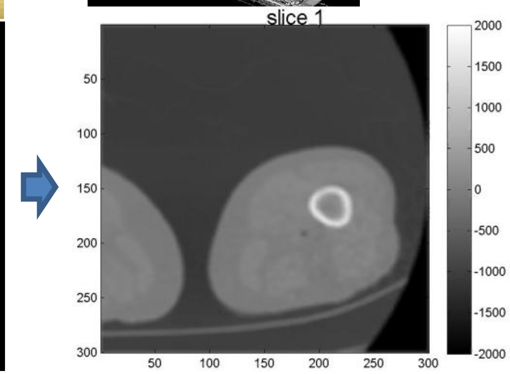


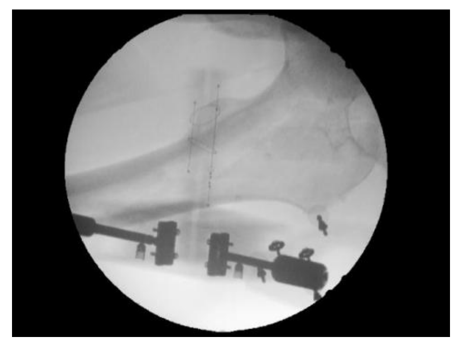
Dataset 3: Cadaver #2

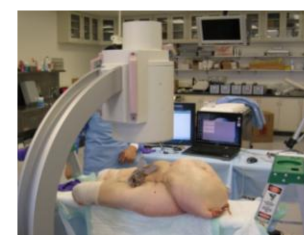
- Preop CT
  - Diagnostic CT at Bayview Medical Center (2009/8/7)
  - 300x300x700 voxel (cropped)
  - 0.835x0.835x0.602 mm/voxel
  - 135 kVp, 250 mAs
  - Gaussian filter to reduce streak artifact
- Intraop X-ray
  - Philips at Bayview (2009/8/10)
  - 480x480 pixel
  - 0.45x0.45 mm/pixel
  - Before cement injection
  - With distortion correction
  - Only 7 images with narrow separation angle

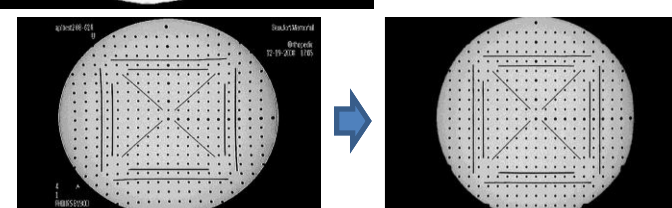




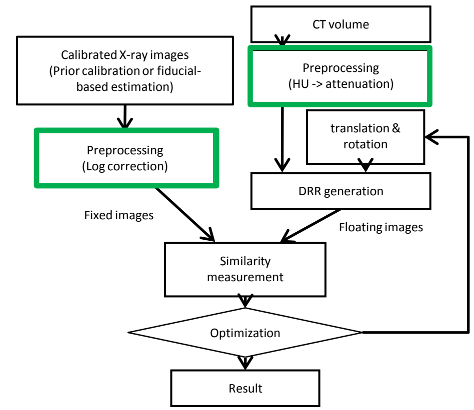




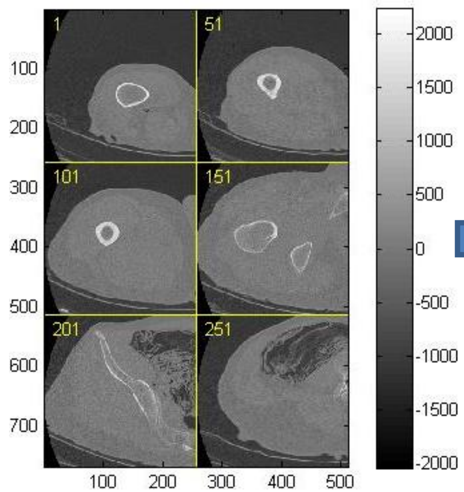




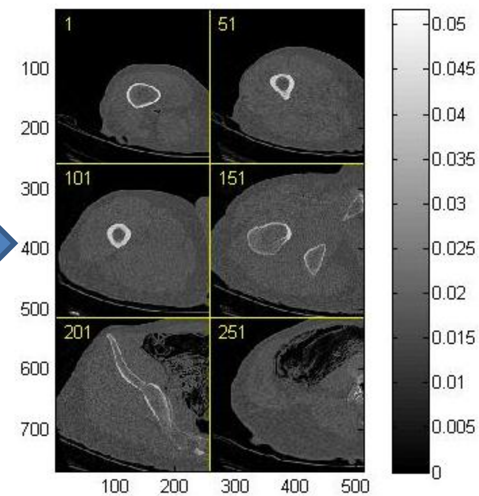
Preprocessing of input images



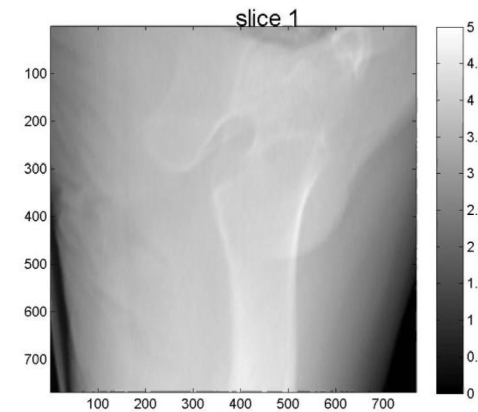
Prior CT (135kVP, 250mAs)



$$HU = 1000 \times \frac{\mu - \mu_{water}}{\mu_{water}}$$

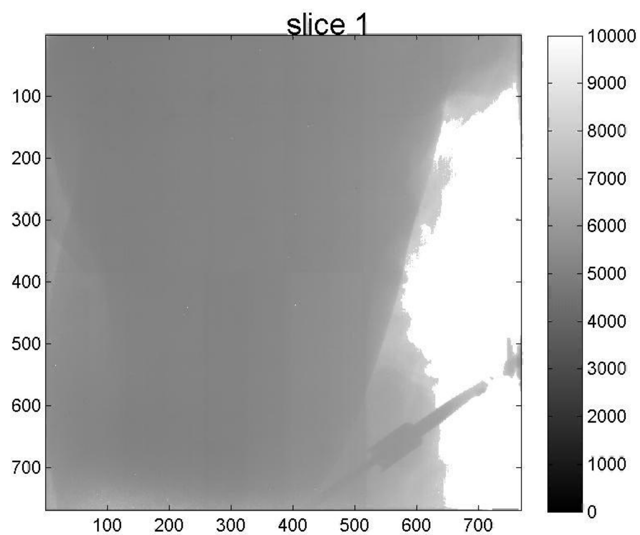


$$\mu$$

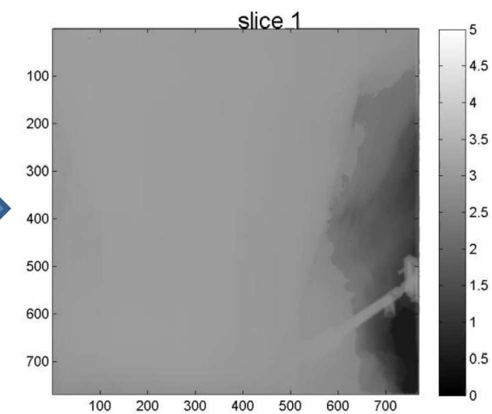


$$DRR = \int_0^d \mu(s;\overline{E})\,ds$$

X-ray projections (120kVp, 5.2mA)

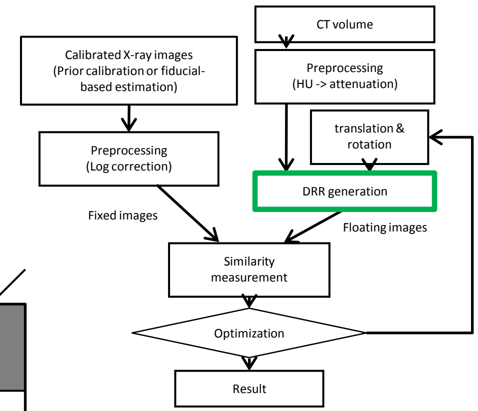


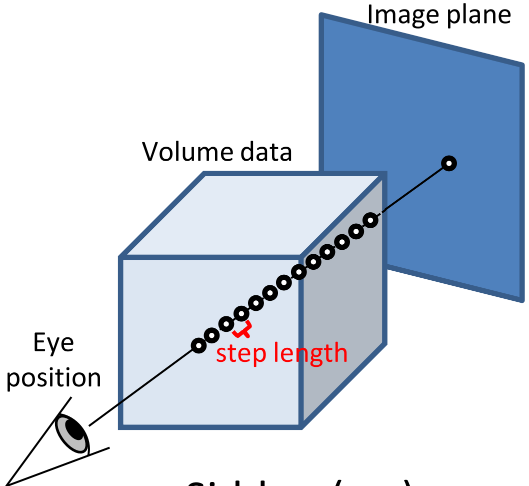
$$I_d = I_0 \exp\left[-\int_0^d \mu(s;\overline{E})\,ds\right]$$

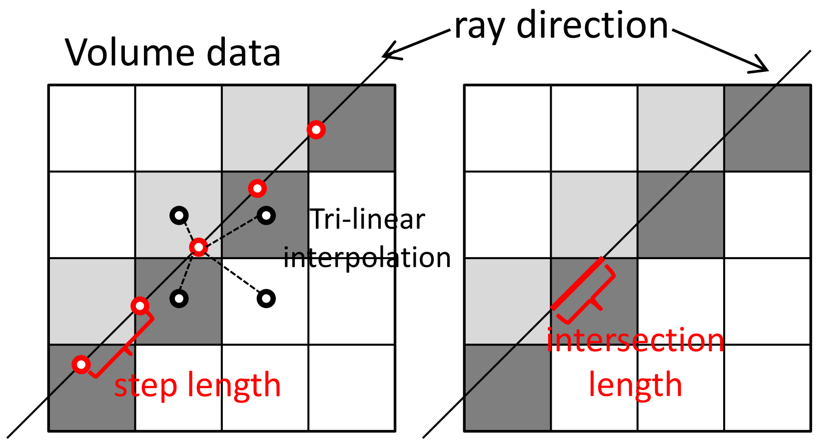


$$g_d = -\ln\left(\frac{I_d}{I_0}\right) = \int_0^d \mu(s;\overline{E})\,ds$$

DRR generation







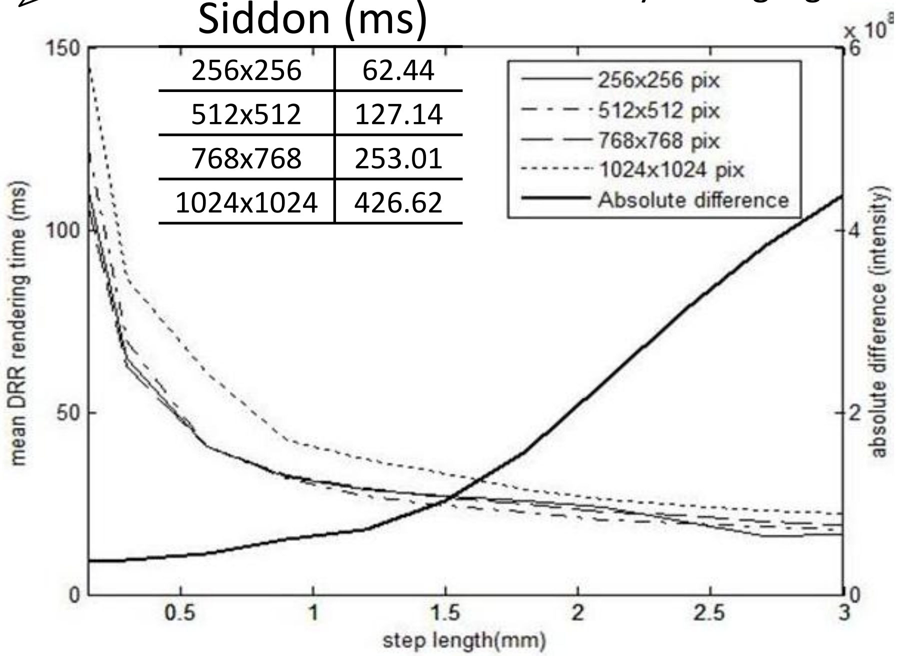
Tri-linear interpolation ray-tracing algorithm

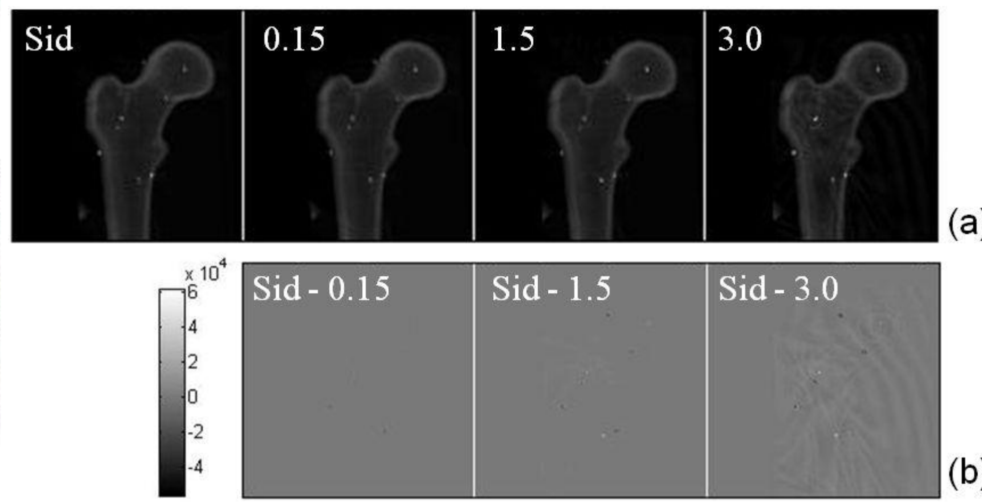
Siddon's ray-casting algorithm

Siddon RL. , "Fast calculation of the exact radiological path for a three-dimensional CT array," Med.Phys. Mar-Apr 12(2), 252-255 (1985)

Siddon (ms)

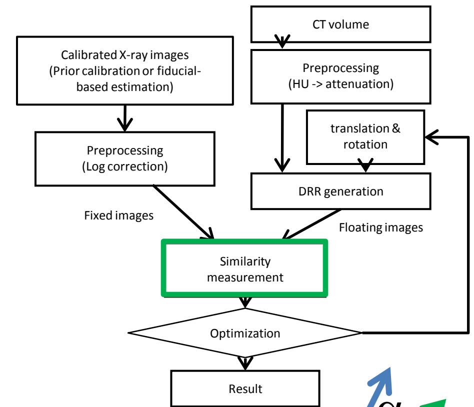
256x256	62.44
512x512	127.14
768x768	253.01
1024x1024	426.62





512×512×512 volume (0.3mm^3/voxel)

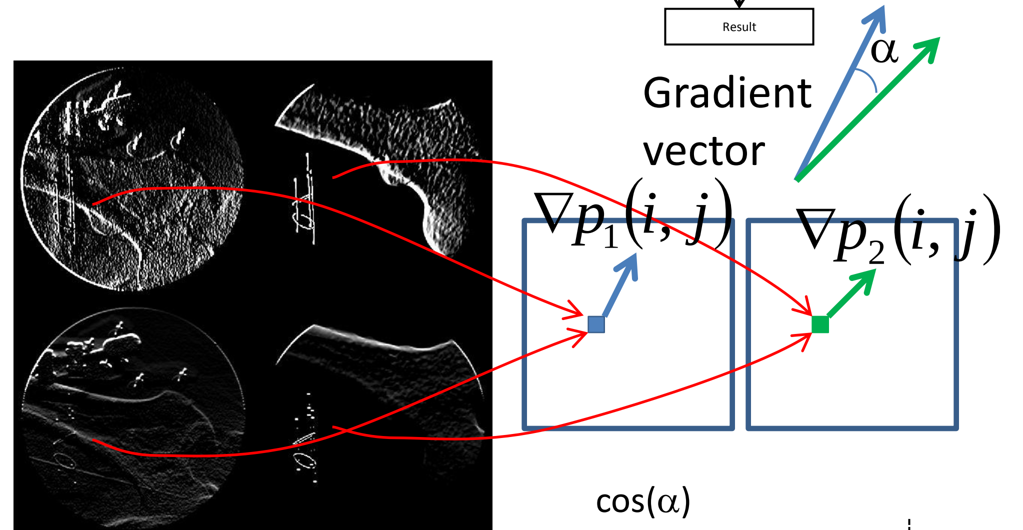
Gradient Information (GI) Similarity Measure



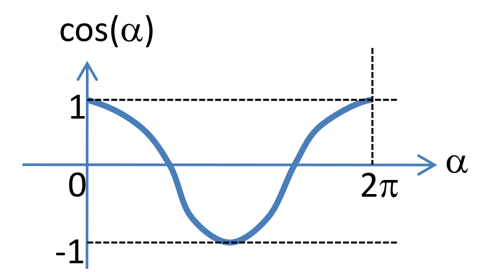
$$\alpha_{i,j} = \arccos \frac{\nabla p_1(i,j) \cdot \nabla p_2(i,j)}{\left|\nabla p_1(i,j)\right| \left|\nabla p_2(i,j)\right|}$$

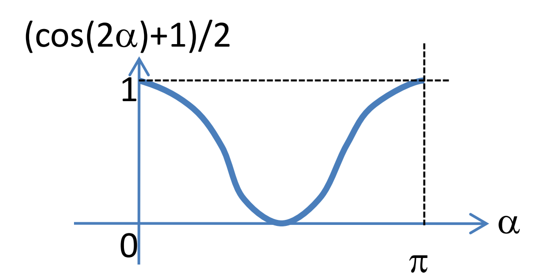
(angle between two gradient vectors)

$$w(i,j) = \frac{\cos(2\alpha_{i,j}) + 1}{2}$$



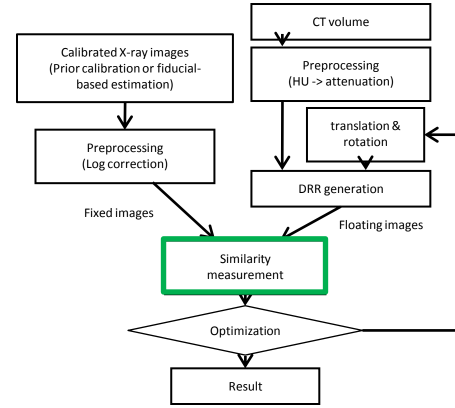
$$G(p_1, p_2) = \sum_{i,j} w(i,j) \min\left(\left|\nabla p_1(i,j)\right|, \left|\nabla p_2(i,j)\right|\right)$$



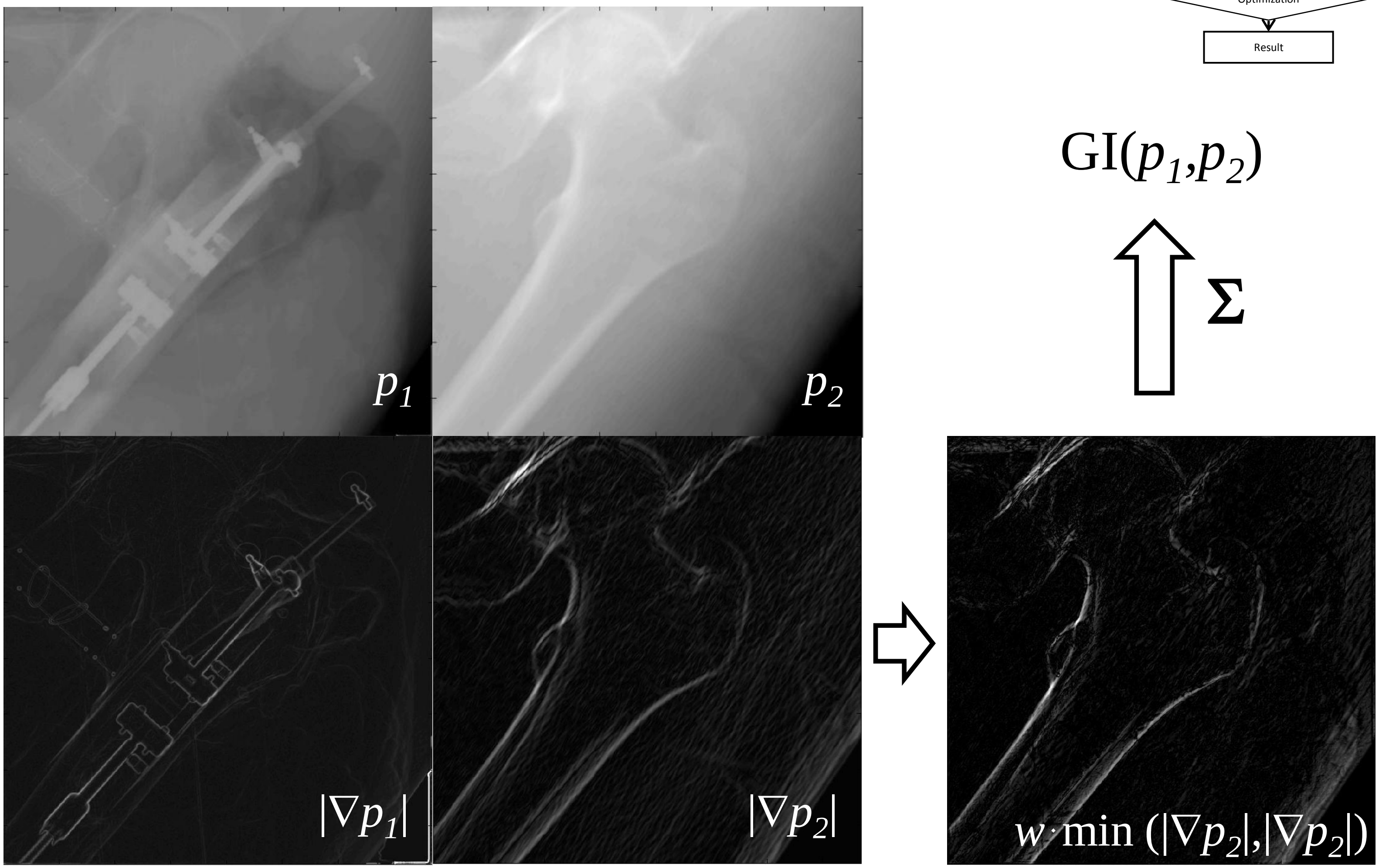


Pluim, J.P., Maintz, J.B. and Viergever, M.A., 2000. Image registration by maximization of combined mutual information and gradient information. IEEE Transactions on Medical Imaging, 19(8), 809-814.

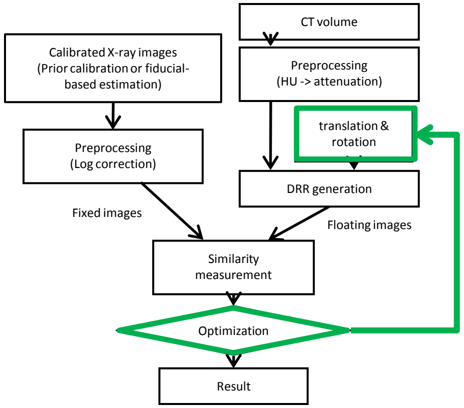
Gradient Information (GI) Similarity Measure



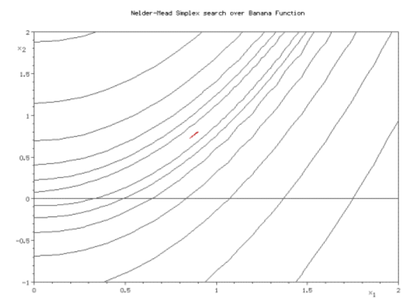
$$\mathrm{GI}(p_1,p_2)$$

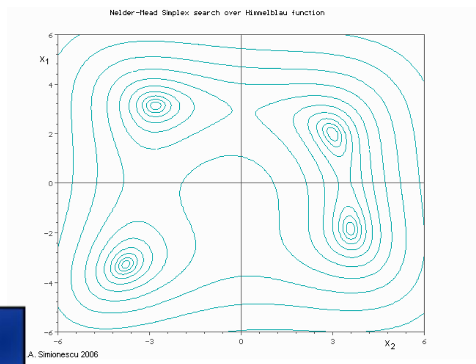


Optimizer

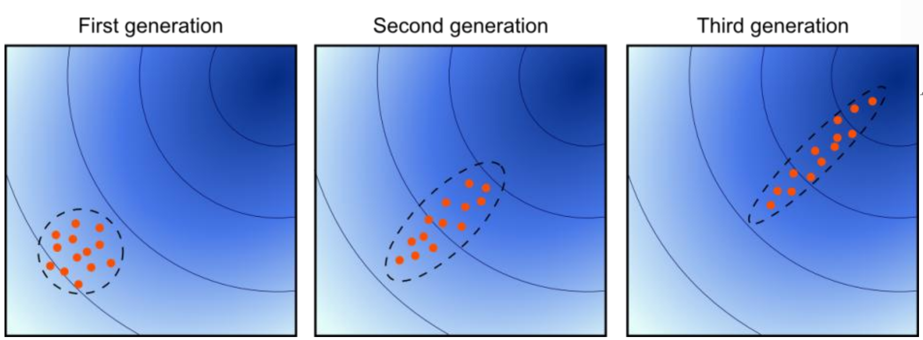


1. Nelder-Mead Downhill Simplex
   - Heuristic optimization algorithm
   - No derivative computation
   - Matlab implementation - fminsearch()
2. CMA-ES (Covariance Matrix Adaptation Evolution Strategy)
   - No derivative computation
   - Known for rubostness and efficiency in a rugged search landscape
   - Matlab implementation by Hansen*1



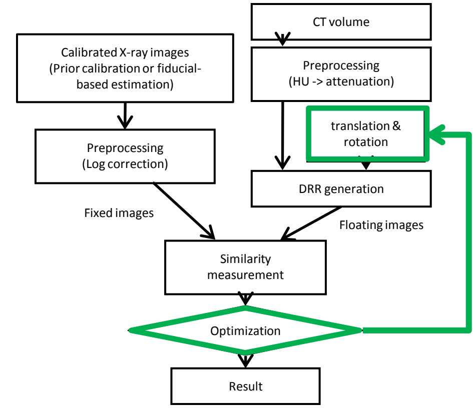


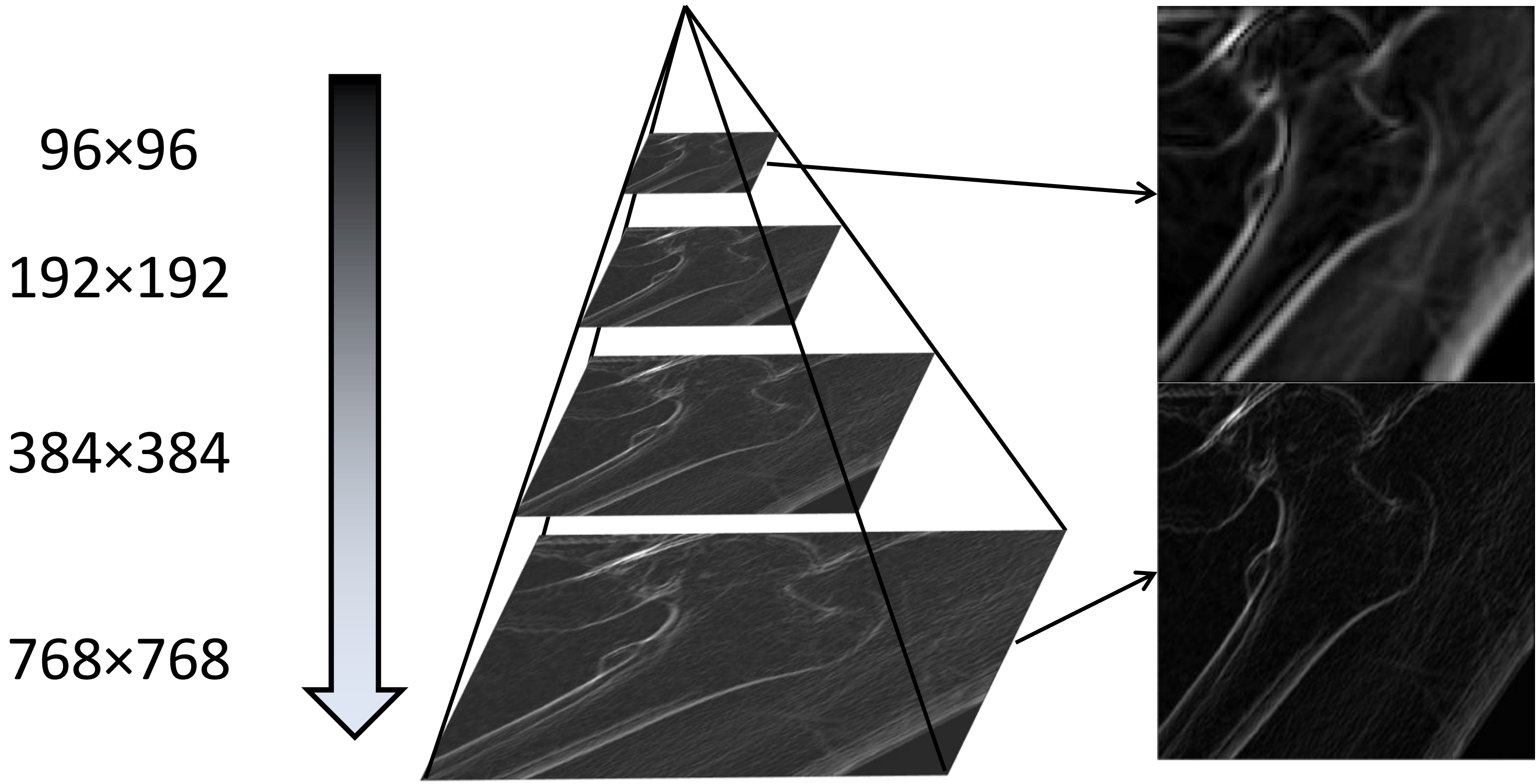
*1: Hansen N. The CMA evolution strategy: a comparing review. In: Lozano JA, Larranaga P, Inza I, Bengoetxea E, editors. Towards a new evolutionary computation. Advances on estimation of distribution algorithms: Springer; 2006. p. 75-102.



from Wikipedia

Coarse-to-fine multi-resolution optimization strategy

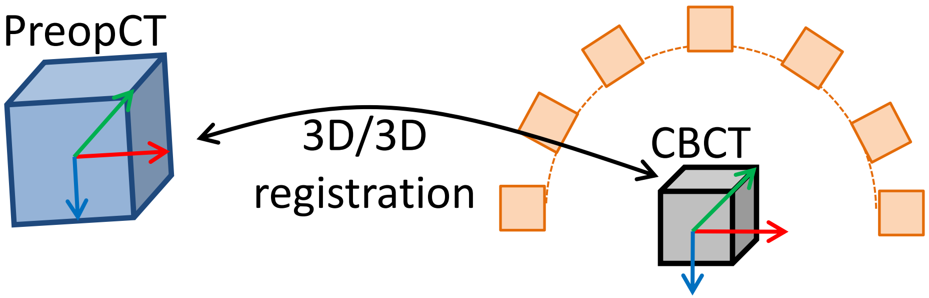




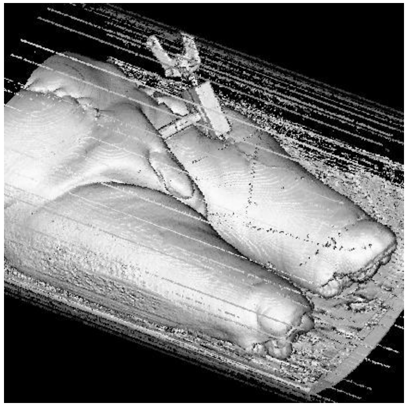
- Optimization was repeated 4 times using different resolution

Ground truth registration

1. Flat-panel C-arm (Siemens C14)
   - Geometric calibration using helix BB phantom
   - 3D/3D registration between Preop CT and CBCT



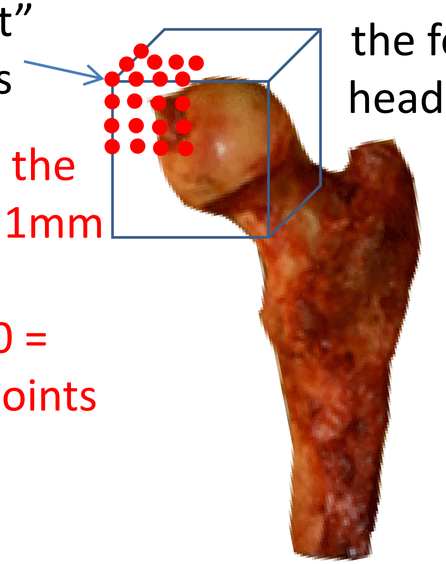
2. Conventional C-arm
   - Pose estimation using FTRAC
   - Diagnostic CT of the cadaver <u>with the FTRAC</u>



Evaluation method: Error measure

- mTRE (mean TRE)
  - An error measure proposed in [1] to determine 3D error of a registration.
  - While it is widely used in literature, definition of the "target" points differ from study to study

[1] van de Kraats EB, Penney GP, Tomazevic D, van Walsum T, Niessen WJ. , "Standardized evaluation methodology for 2-D-3-D registration," IEEE Trans.Med.Imaging Sep 24(9), 1177-1189 (2005).



"Target" points

$$mTRE = \frac{1}{k}\sum_{i=1}^{k}\left\| T_{reg}\, p_i - T_{gold}\, p_i \right\|$$

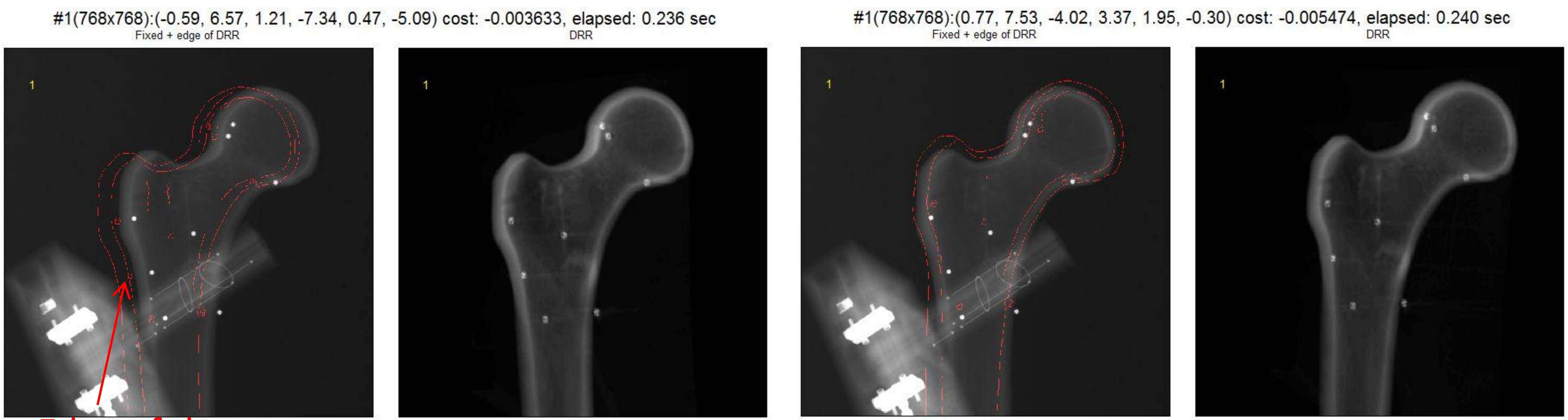
Estimated   Ground truth

Evaluation method: Initial guesses

- 50 registration trials from different initial pose were conducted
- Initial poses were randomly selected by perturbing the ground truth registration by [-10 +10] mm, [-10 +10] degrees.
- The same initial poses were used for all experiments

Fixed image

Floating image

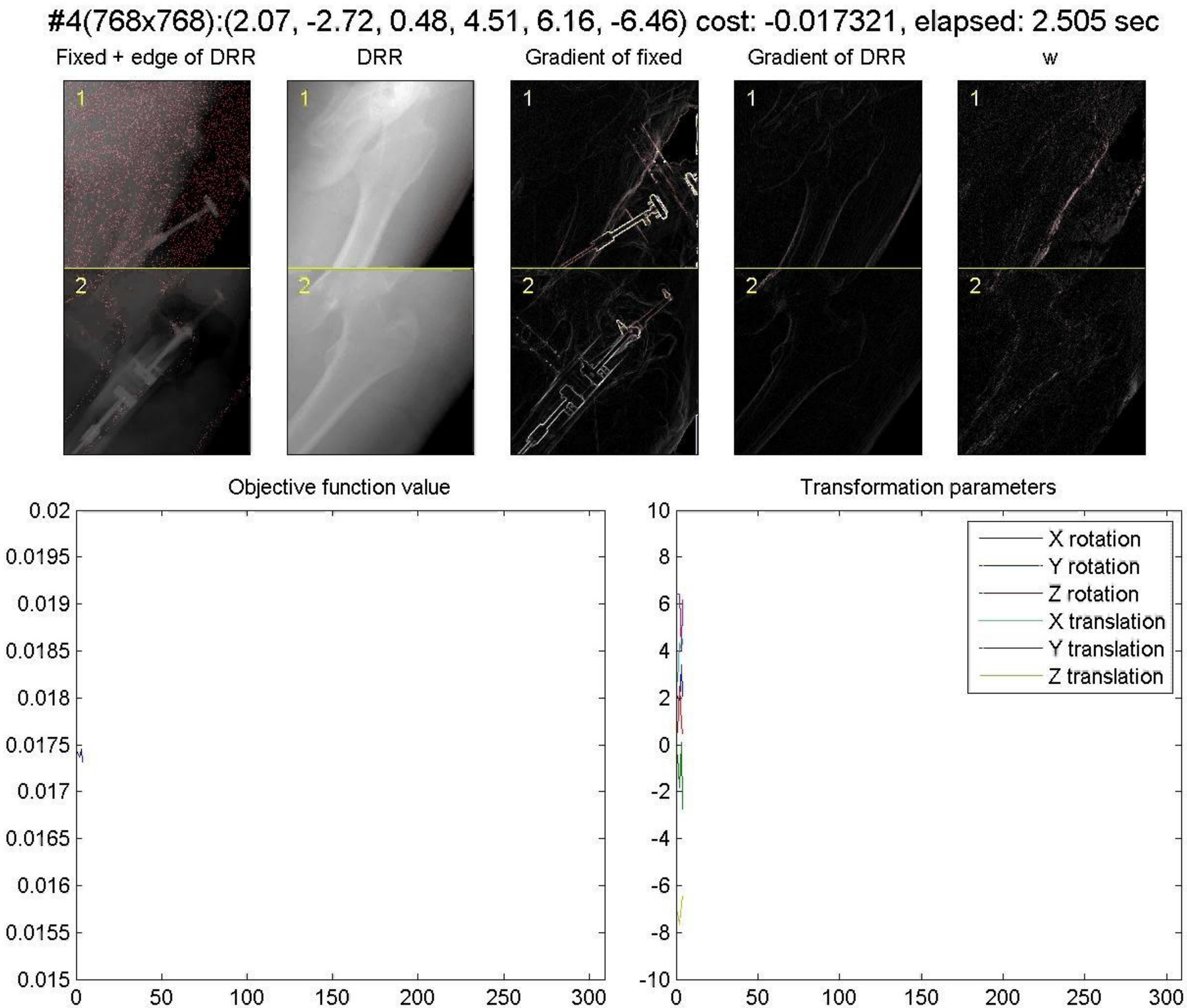


Edge of the floating image

Randomly generated initial guess (2 examples out of 50)

Results

One typical trial

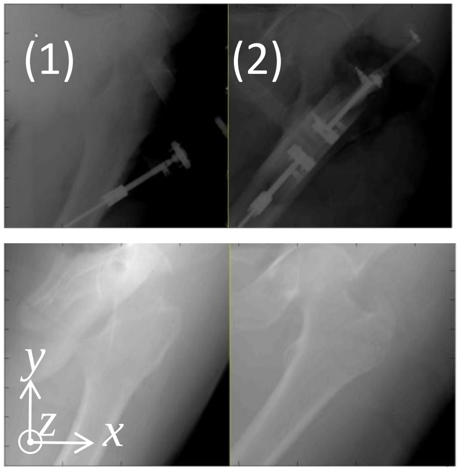


- GI similarity measure
- CMA-ES optimizer
- Siddon-based DRR

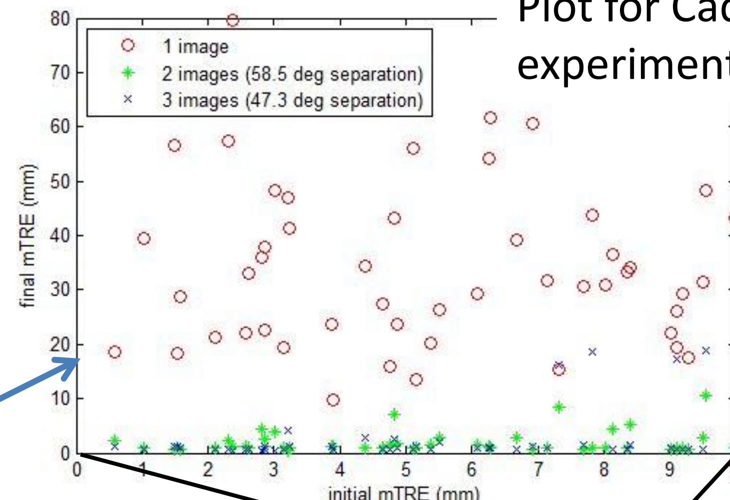
Result of 50 trials: comparison of number of images

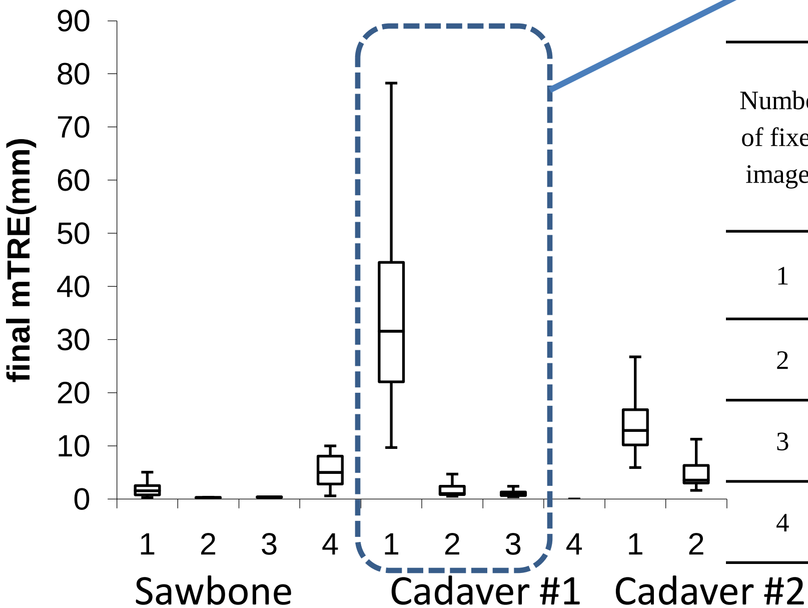
Fixed images

Floating images



Plot for Cadaver #1 experiment

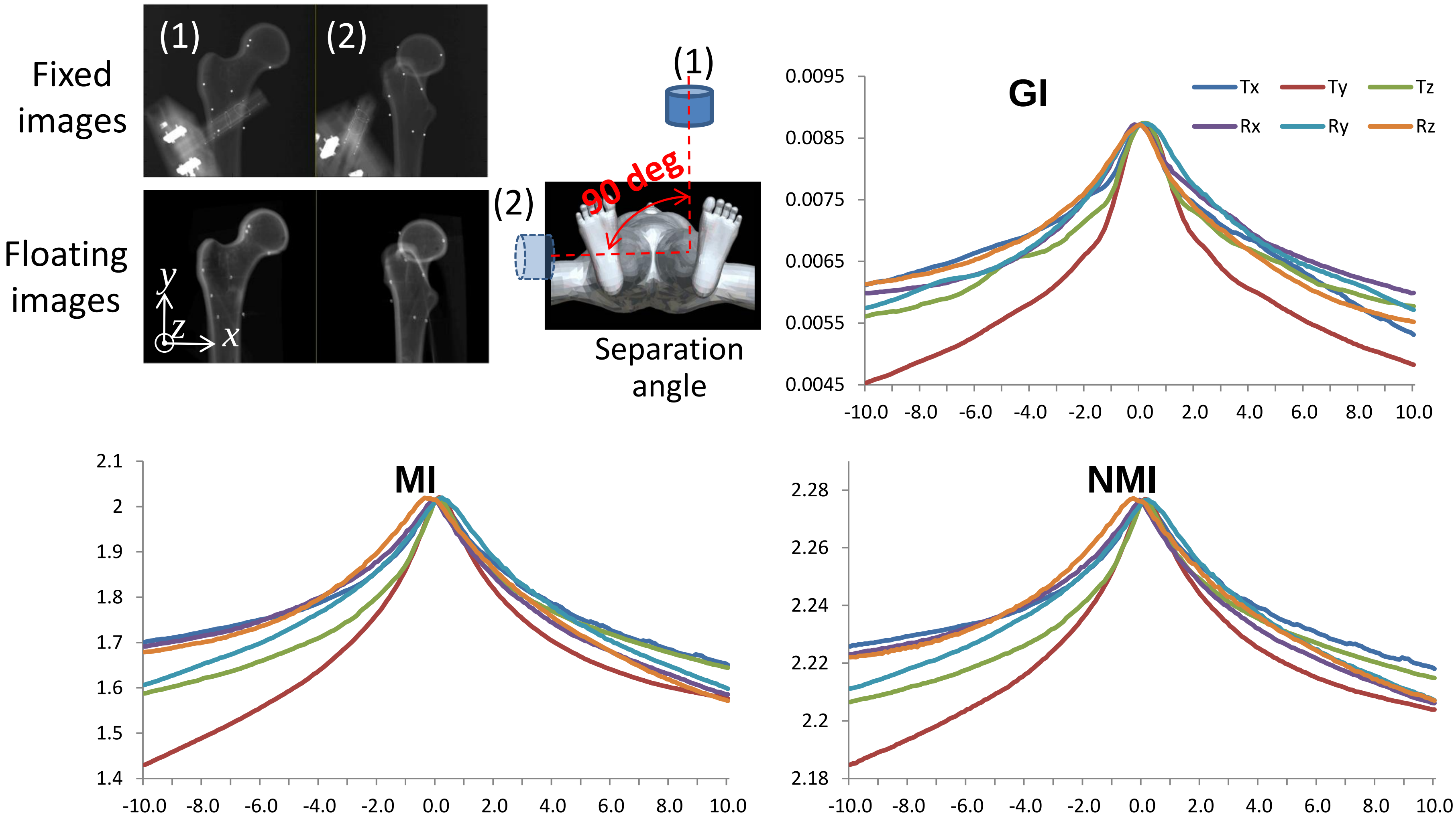




Number of fixed images	Sawbone mTRE (mm) for success	Sawbone T (s)	Sawbone R (%)	Sawbone Δ (deg)	Cadaver #1 mTRE (mm) for success	Cadaver #1 T (s)	Cadaver #1 R (%)	Cadaver #1 Δ (deg)	Cadaver #2 mTRE (mm) for success	Cadaver #2 T (s)	Cadaver #2 R (%)	Cadaver #2 Δ (deg)
1	1.560 ± 1.013	142.8 ± 3.5	94	n/a	*	159.3 ± 3.8	0	n/a	*	104.3 ± 8.0	0	n/a
2	0.229 ± 0.042	280.1 ± 5.6	100	90	1.404 ± 0.967	323.5 ± 5.7	92	58.5	3.241 ± 0.756	184.5 ± 6.1	68	31.5
3	0.325 ± 0.027	410.6 ± 16.1	100	60	1.021 ± 0.682	450.5 ± 5.4	90	47.3	n/a	n/a	n/a	n/a
4	0.314 ± 0.025	684.9 ± 40.2	100	45	----	----	--	31.5	n/a	n/a	n/a	n/a

T - registration time, R - success rate, D - angle between consecutive images used in the experiment

Similarity Measures: Sawbone Images

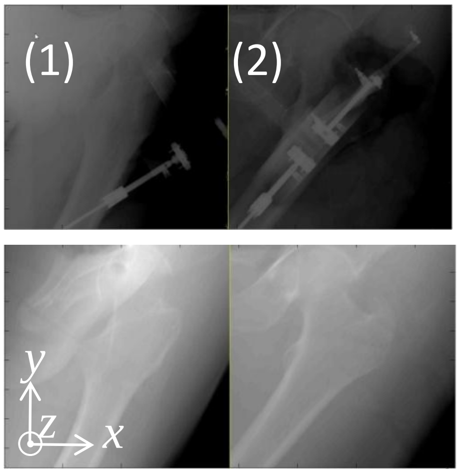


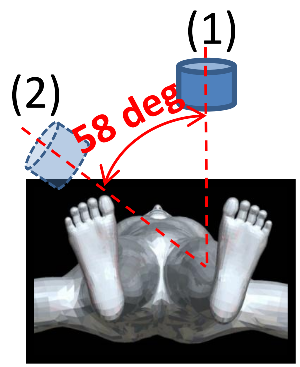
*Sum of the similarity measures of the 2 image pairs

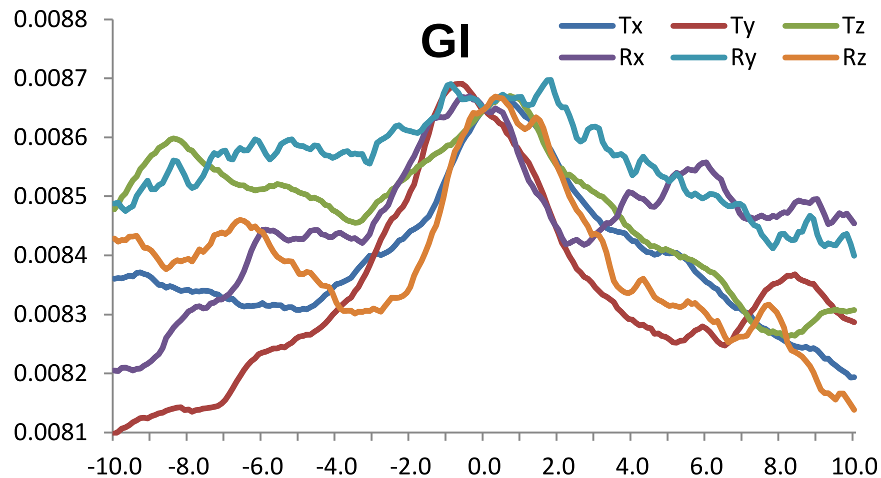
Similarity Measures: Cadaver #1 (flat-panel)

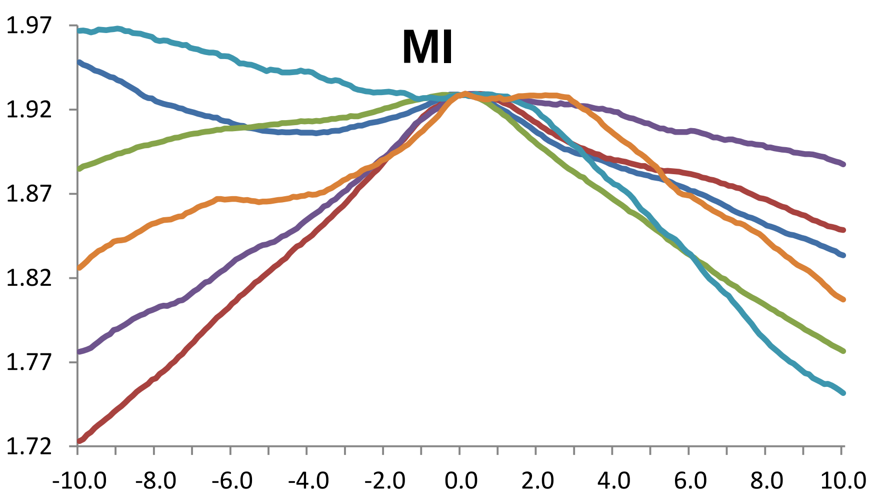
Fixed images

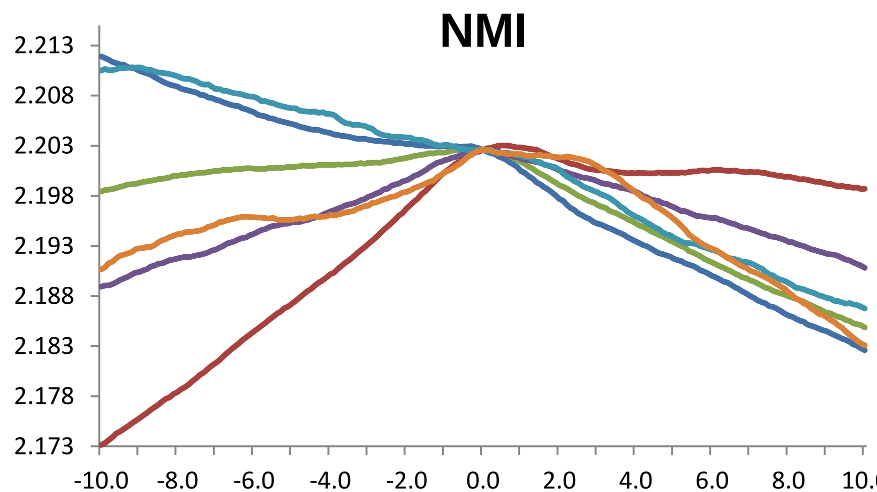
Floating images









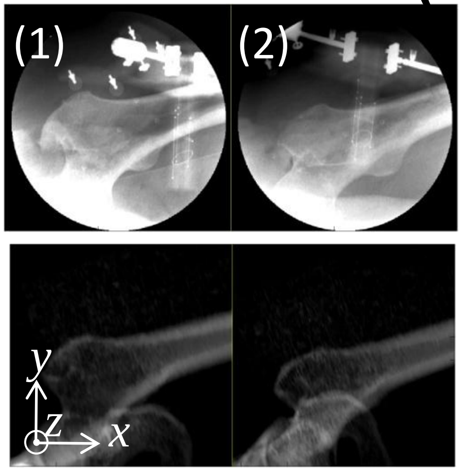


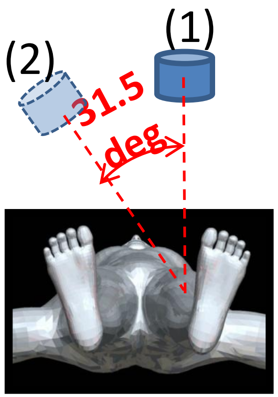
Challenges: Soft tissue and metallic instruments

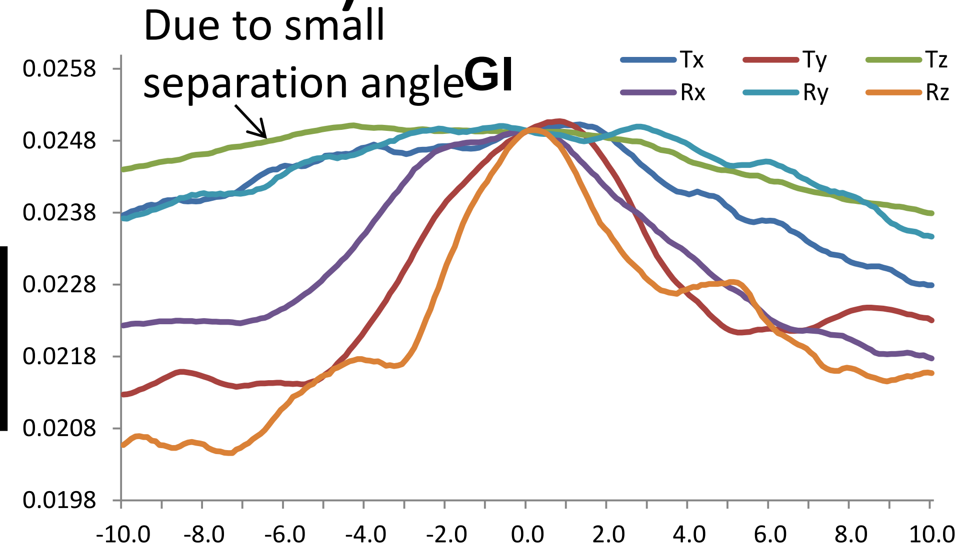
Similarity Measures: Cadaver #2 (image intensifier)

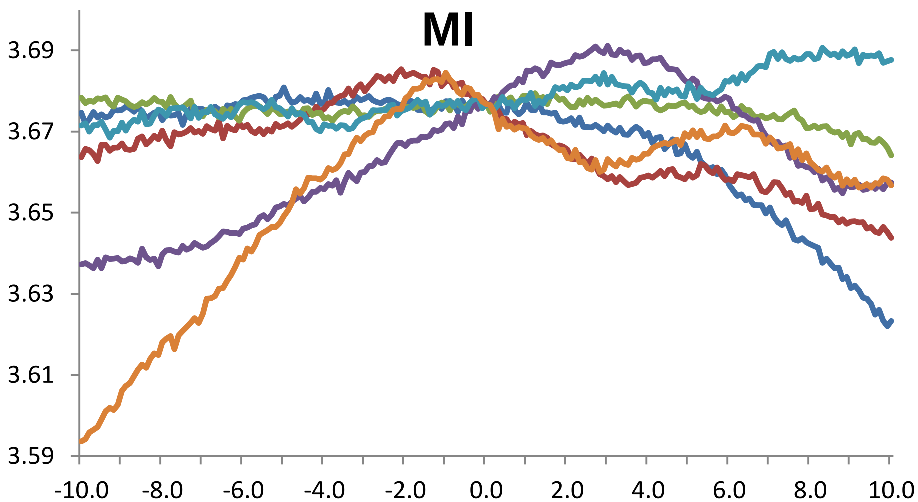
Fixed images

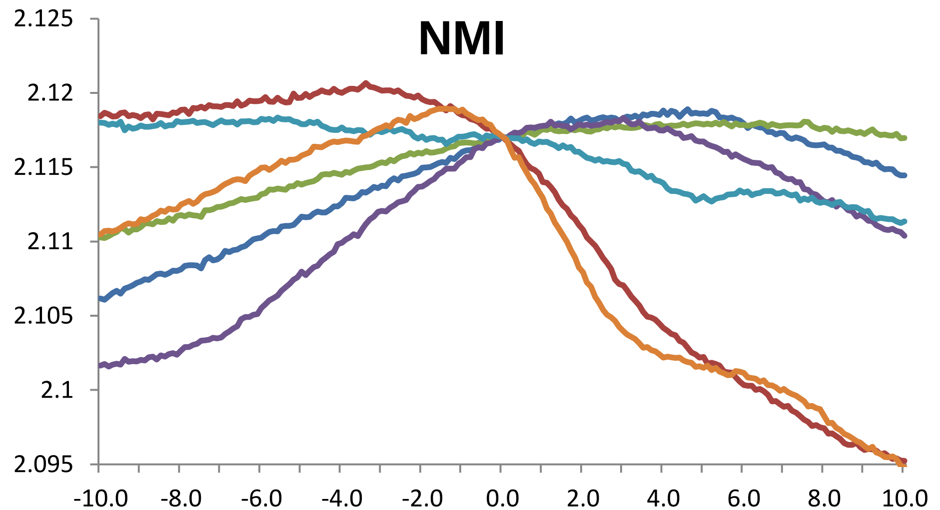
Floating images







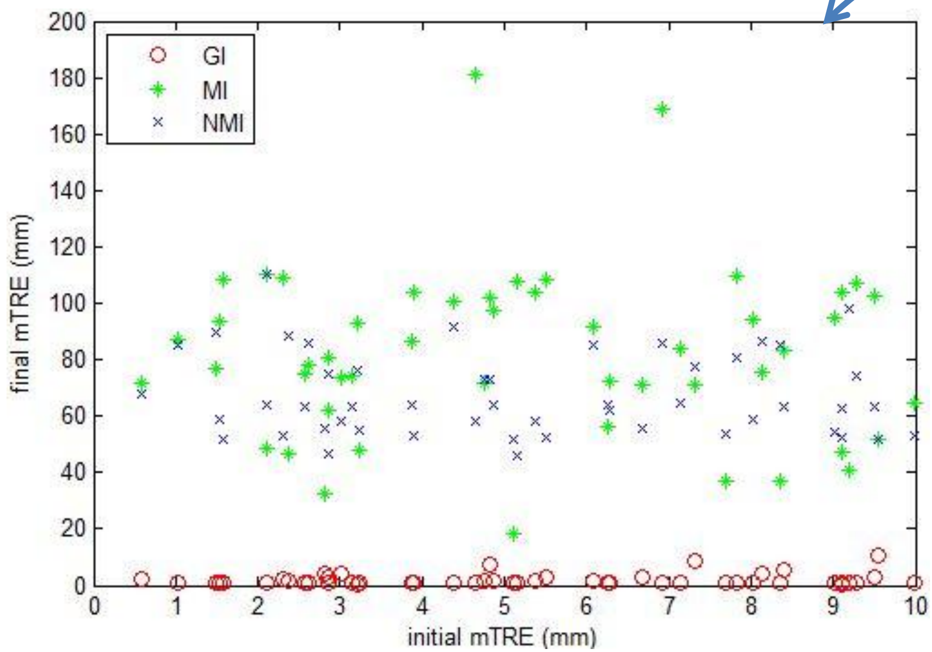


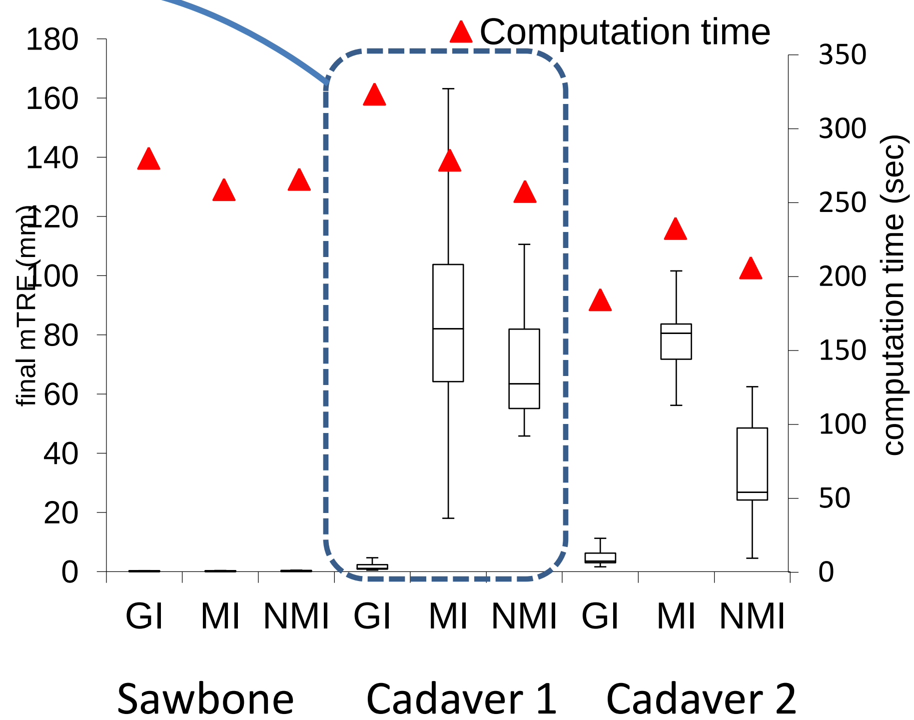


Challenges: Soft tissue, metallic instruments, small separation angle

Result of 50 trials: comparison of similarity measures

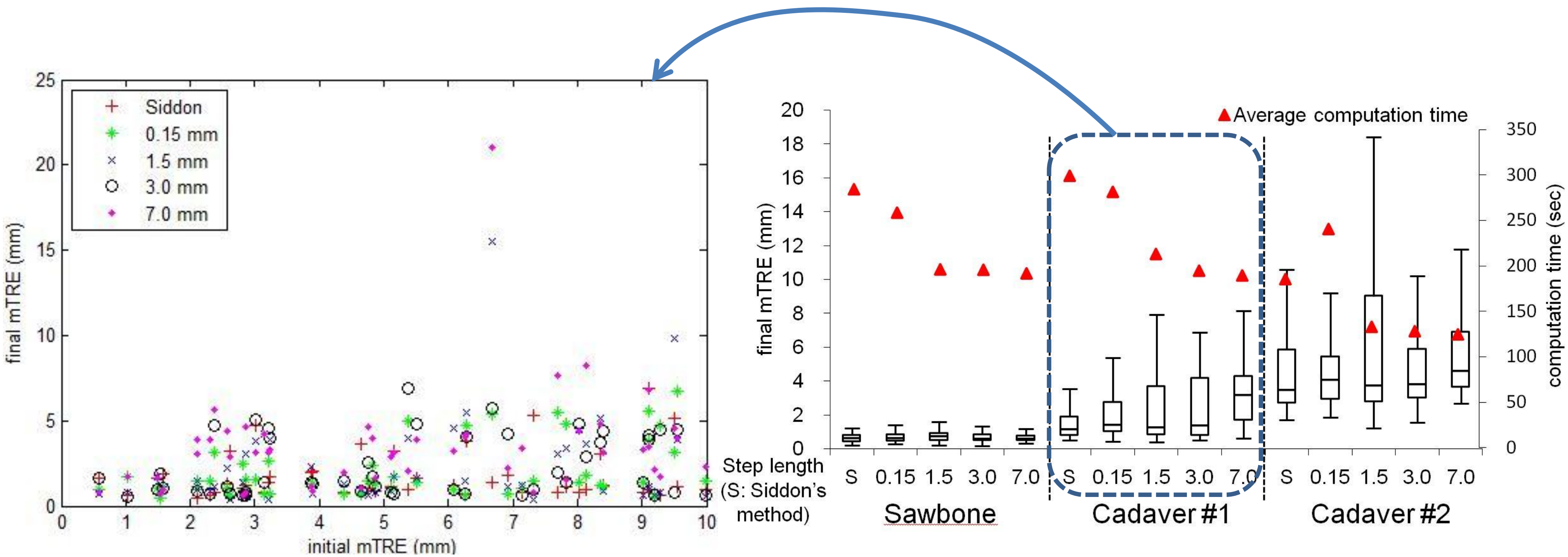
Cadaver #1 (2 images)

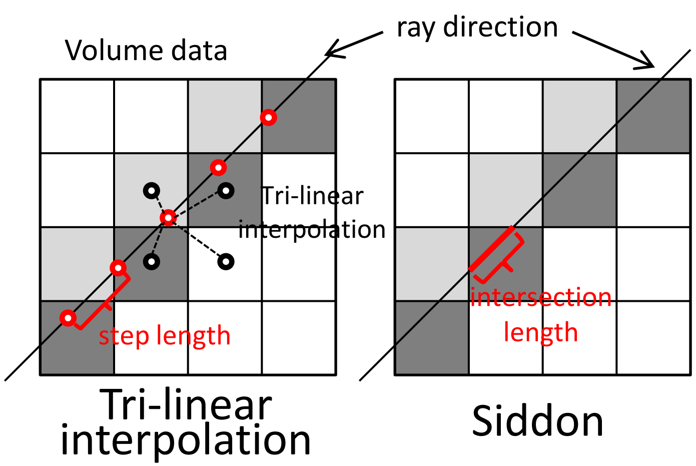




- All similarity measures showed almost the same performance in Sawbone images.
- GI worked significantly better than MI and NMI in cadaver images

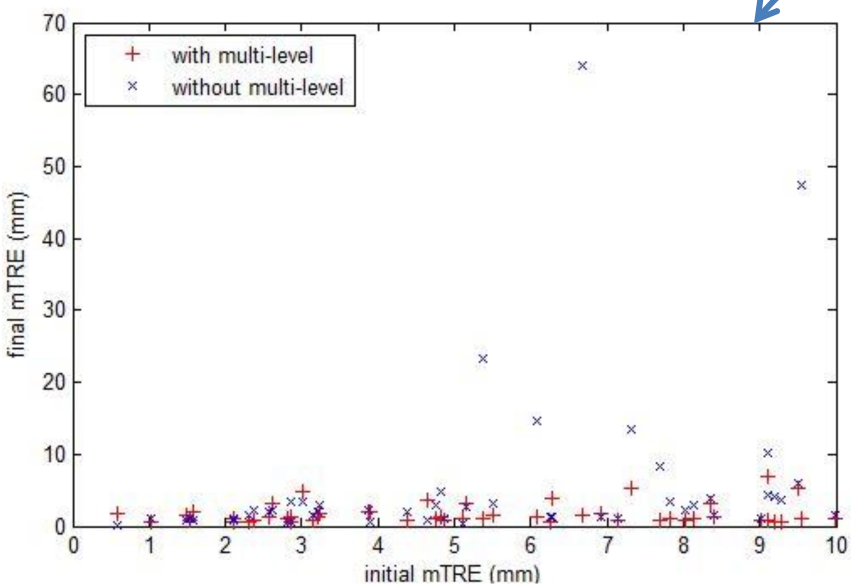
Comparison of DRR generator

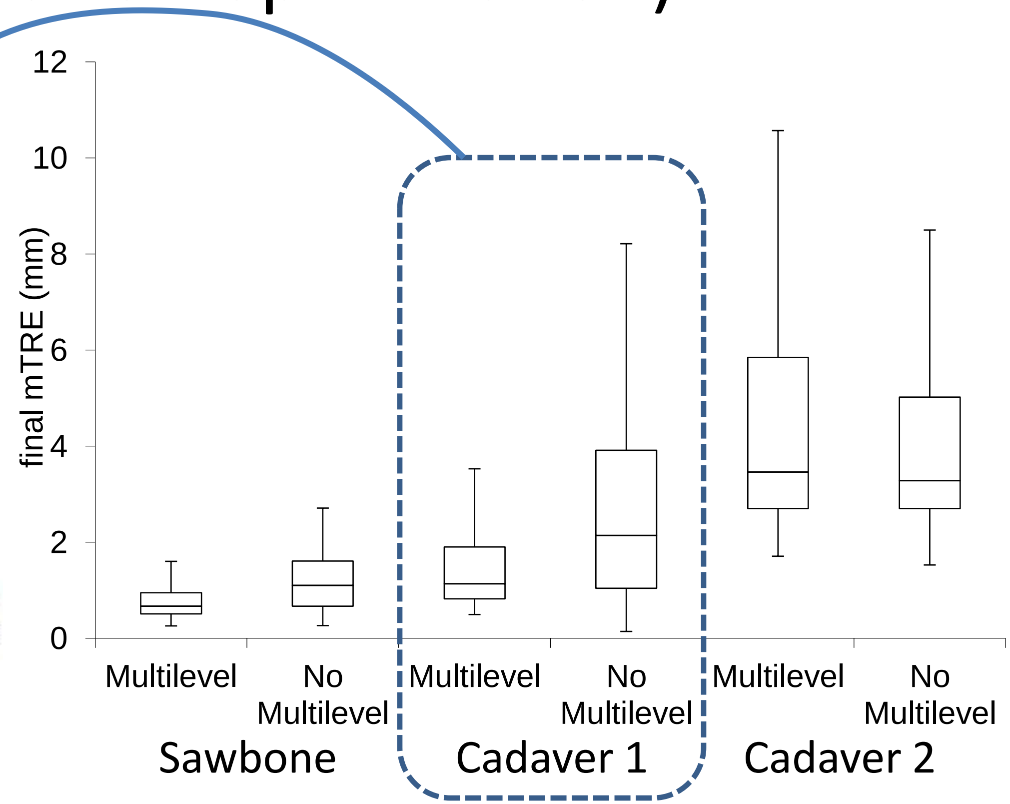




- Step length didn't affect registration result in Sawbone and Cadaver #2
- In Cadaver #1, increasing step length increased final mTRE

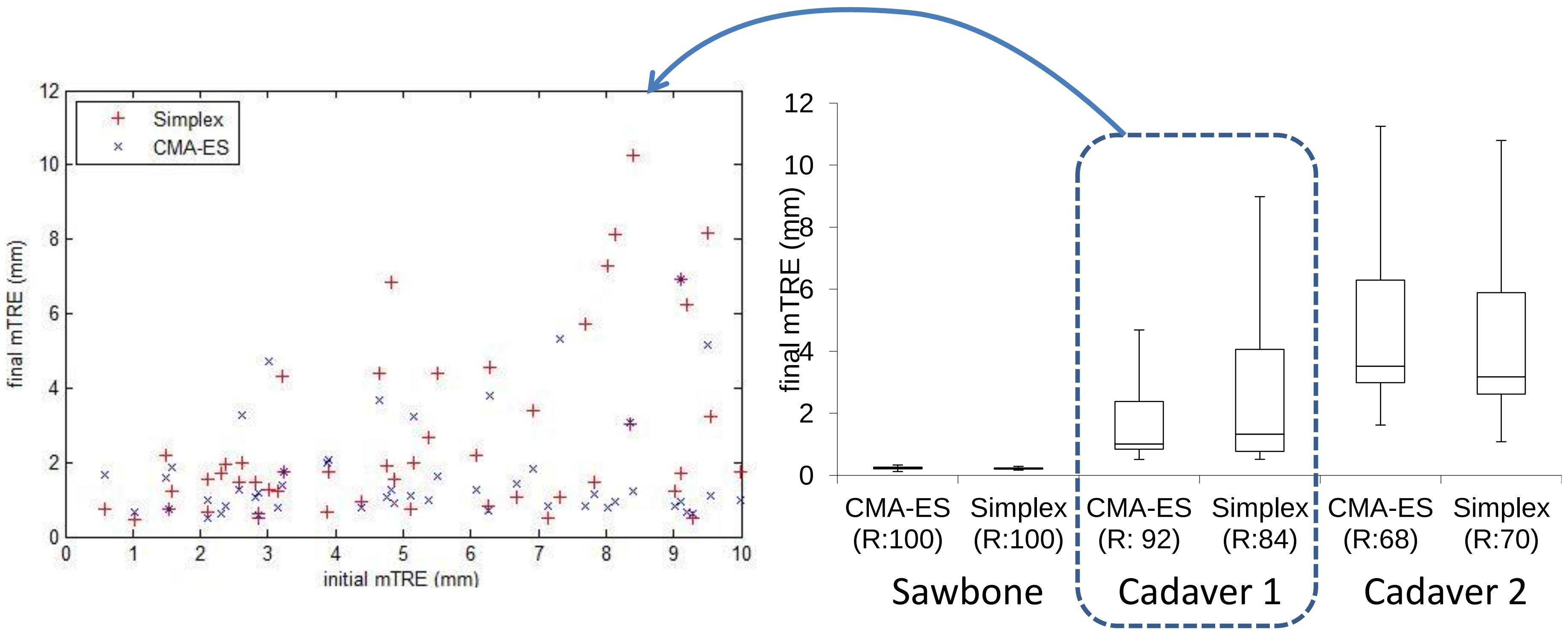
Comparison of Optimization Strategy (coarse-to-fine multi-resolution optimization)





- The effect of coarse-to-fine multi-resolution strategy was statistically significant in Sawbone (P<0.001) and Cadaver 1 (P=0.026)
- No significant difference was found in Cadaver2 (because of the resolution of the original images?)

Comparison of Optimizer



- In Cadaver #1 images, CMA-ES showed better precision than Downhill Simplex.
- Other two experiments showed no difference

(To be confirmed by a statistical analysis)

Summary

- In 3 experiments, mean TRE was 0.229 ± 0.042mm, 1.404 ± 0.967mm, 3.241 ± 0.756mm, respectively.
- GI worked significantly better than MI and NMI in presence of soft tissues.
- Coarse-to-fine multi-resolution strategy was significantly effective
- Registration accuracy may depend on the target anatomy, image quality as well as the geometries (separation angles) of each image.

Next step

- Integration with navigation systems
  - Bone augmentation navigation system, TREK, etc.
- Further cadaver studies
- Integration with reconstruction algorithms that require registered prior CT
  - SxMAC, ROI recon, Hybrid recon, etc.