The ASTRA Toolbox

Fast and flexible tomographic reconstructions

Wim van Aarle[1], W. J. Palenstijn[1,2], J. Batenburg[1,2], J. Sijbers[1]

1 iMinds–Vision Lab, University of Antwerp, Belgium

2 Centrum Wiskunde & Informatica (CWI), Amsterdam, The Netherlands

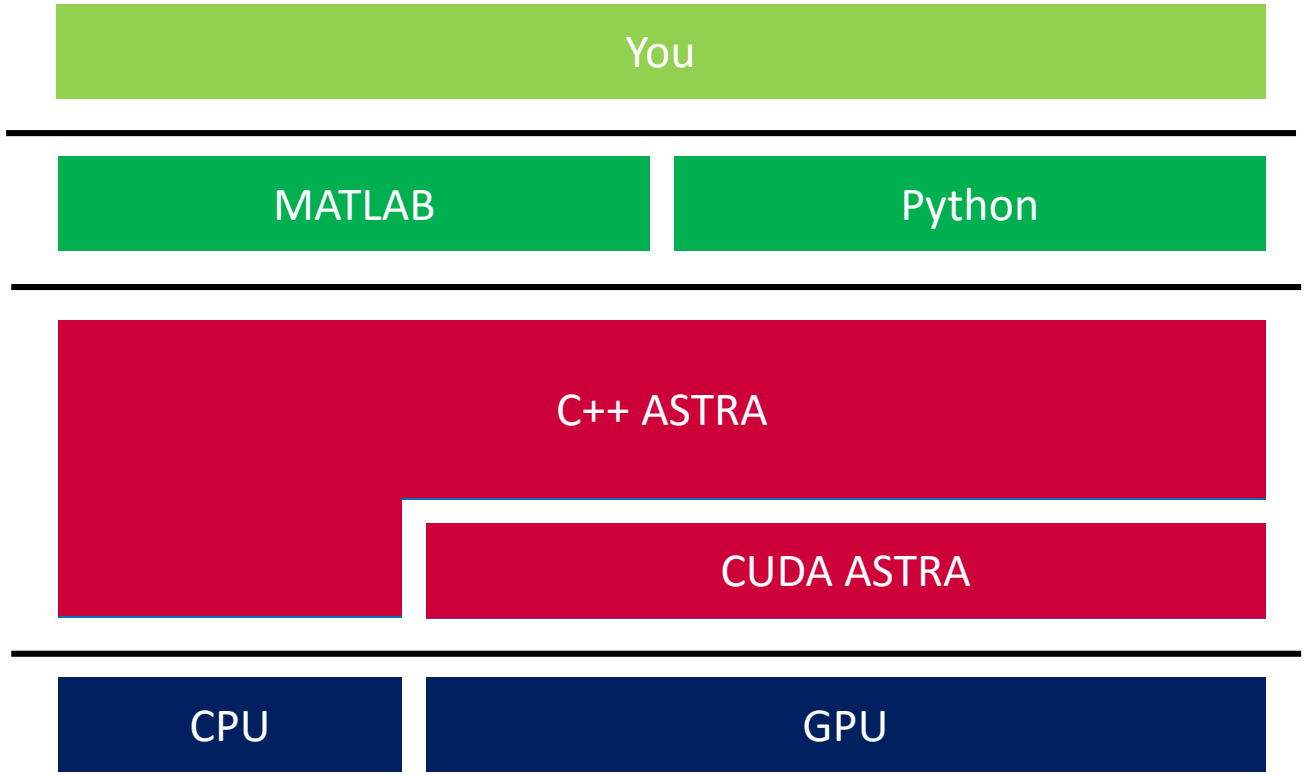
History

The ASTRA toolbox was originally developed at the Vision Lab of the University of Antwerp, Belgium

- 2007: toolbox was born
  - initial goal: reduced implementation work for internal PhD projects
- 2010: interest from external labs and companies
  - e.g. ESRF (France) and FEI (The Netherlands)
- August 2012: first open source release
- 2014-: active development continues at Vision Lab and CWI

What is ASTRA?

All Scale Tomographic Reconstruction Antwerp



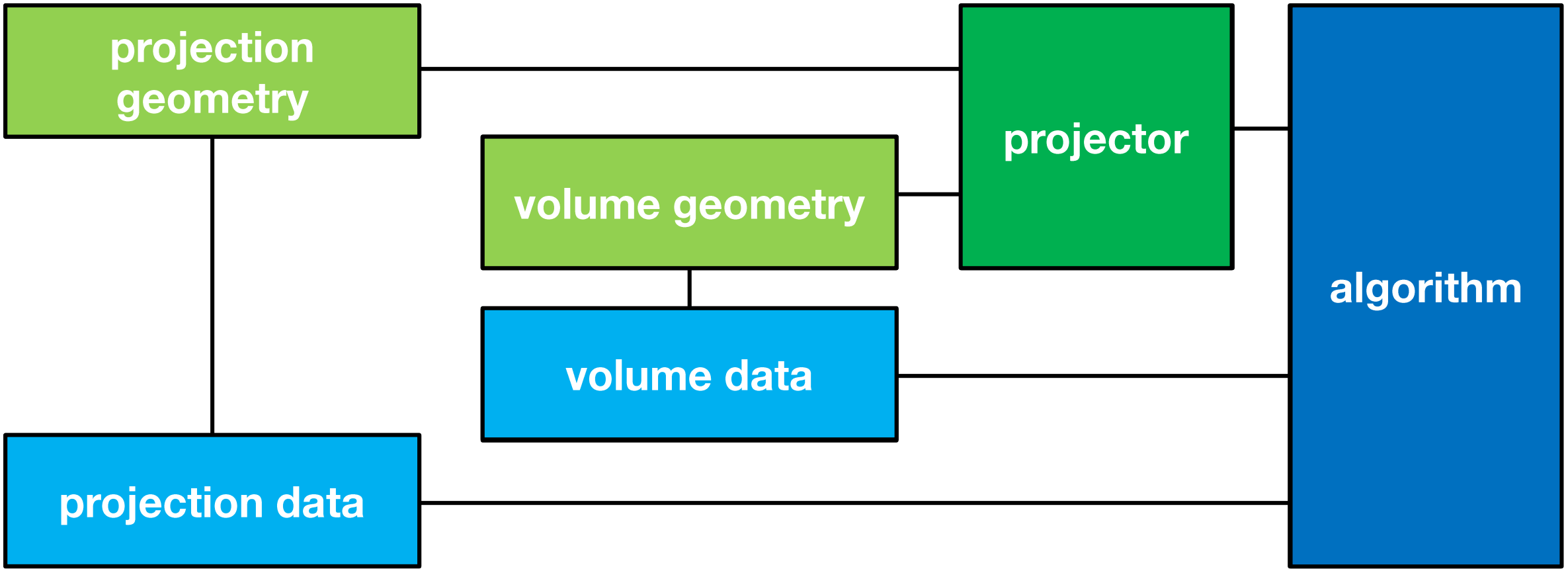
Easy to use
MATLAB and Python interface

Flexible
building block for custom algorithms

Broad support
algorithms and geometries

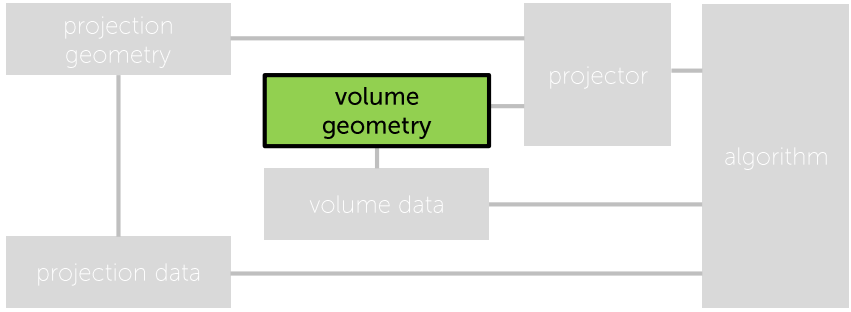
Powerful
C++ and CUDA

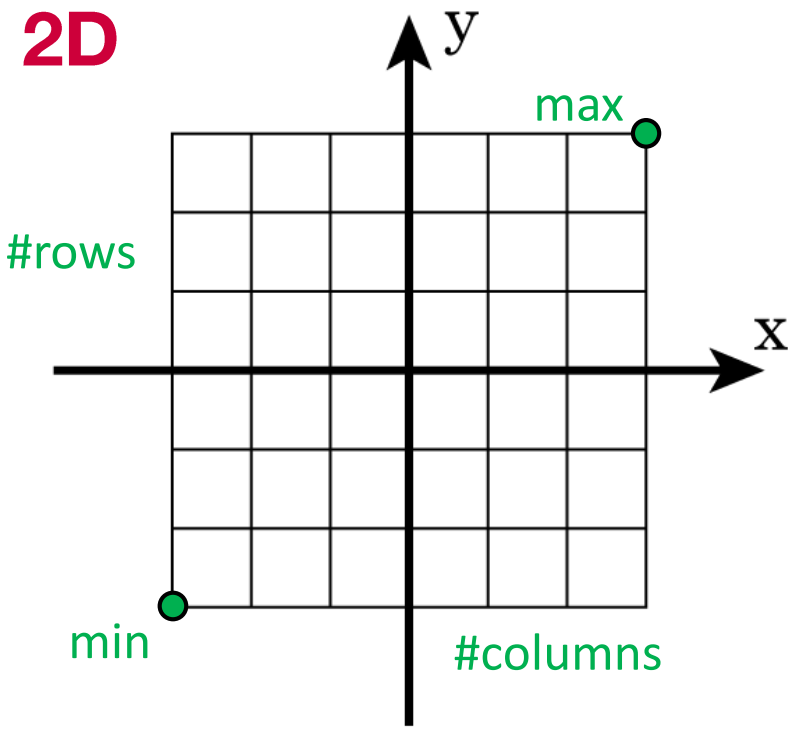
Modules

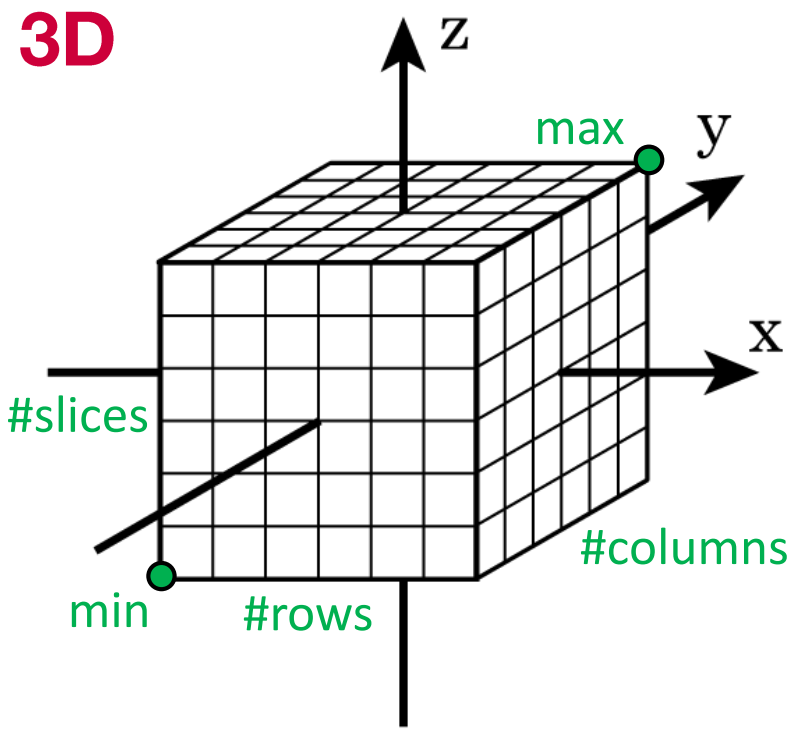

projection geometry
volume geometry
projector
algorithm
volume data
projection data

Volume Geometry

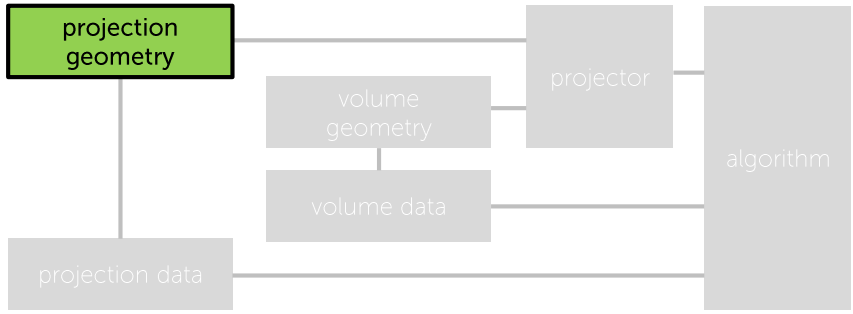
Defines properties of the reconstruction volume





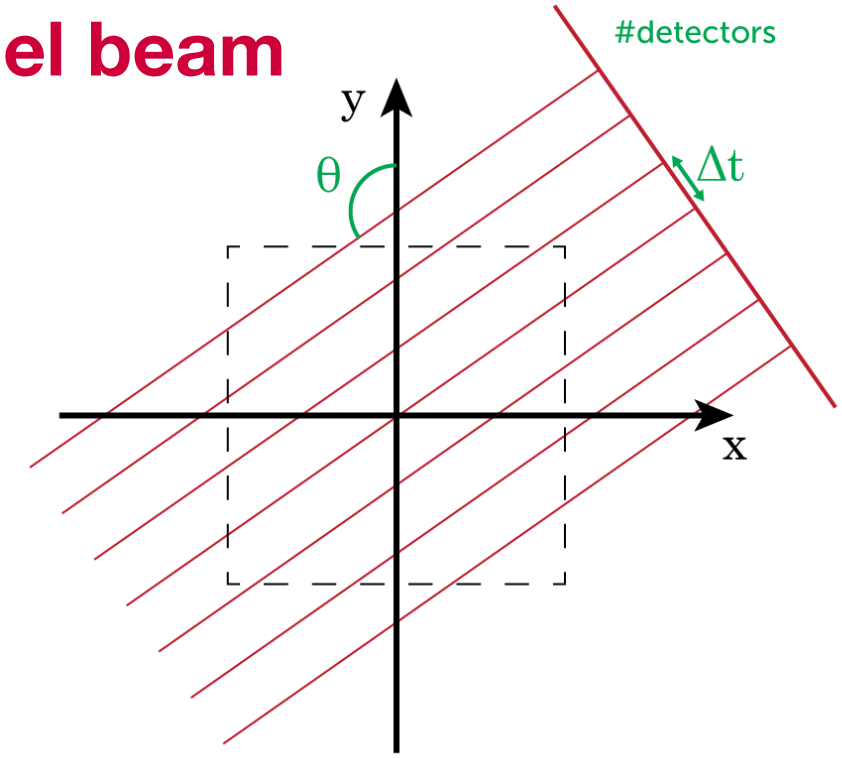


Projection Geometry

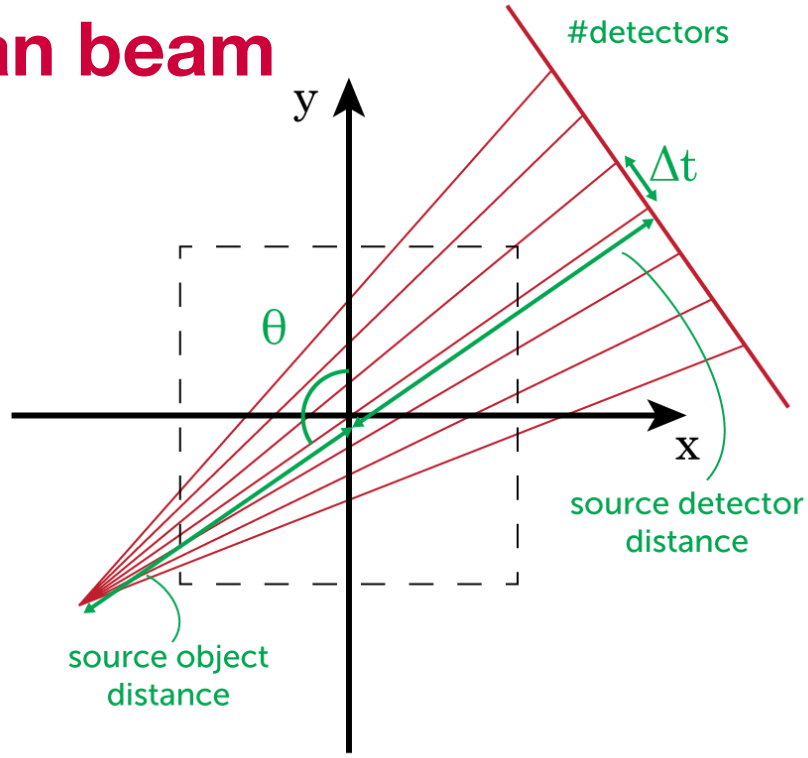


Defines the trajectory of the source/detector with respect to the volume

Parallel beam

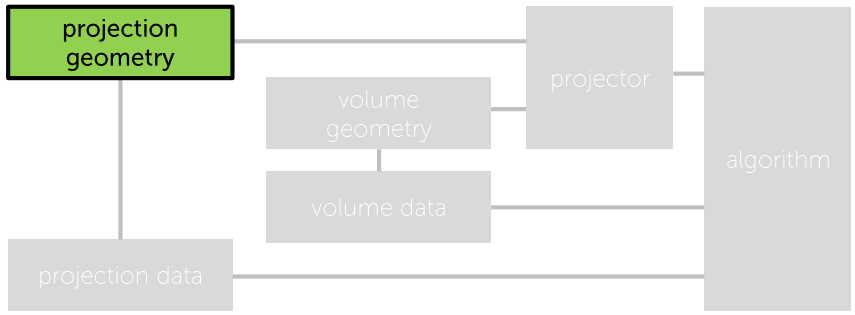


Fan beam

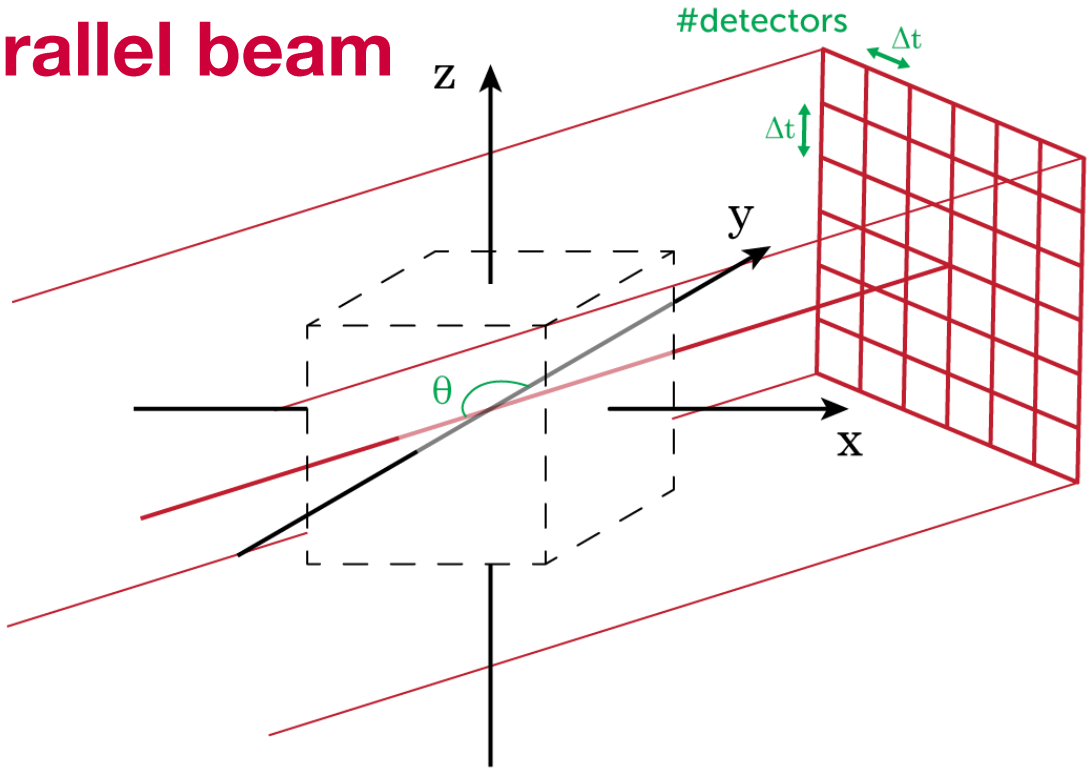


Projection Geometry

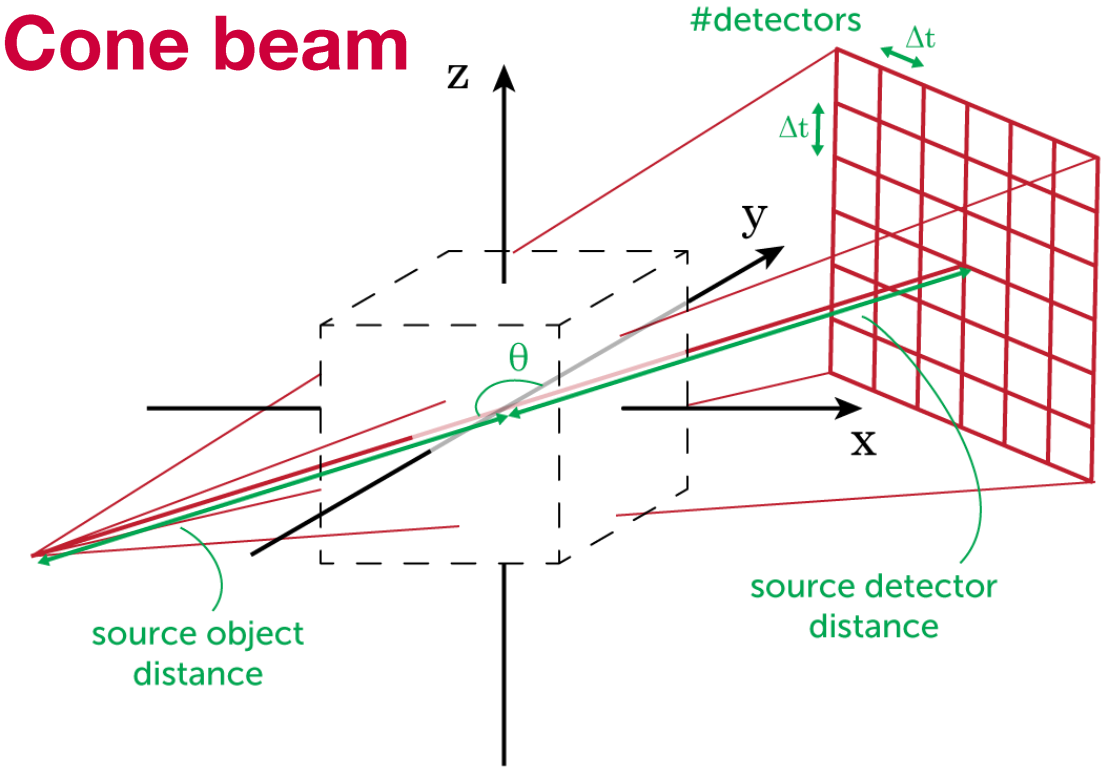
Defines the trajectory of the source/detector with respect to the volume



3D Parallel beam



Cone beam

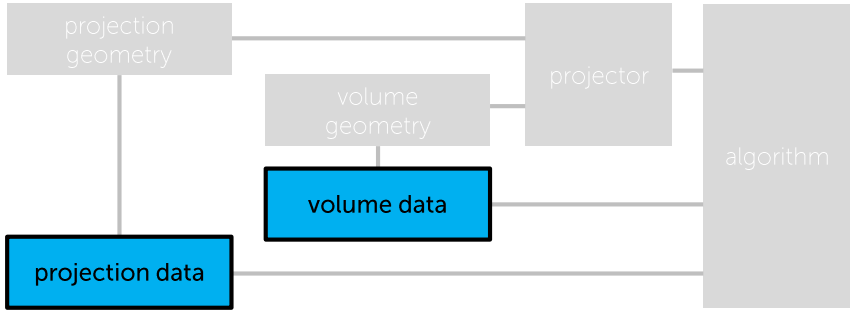


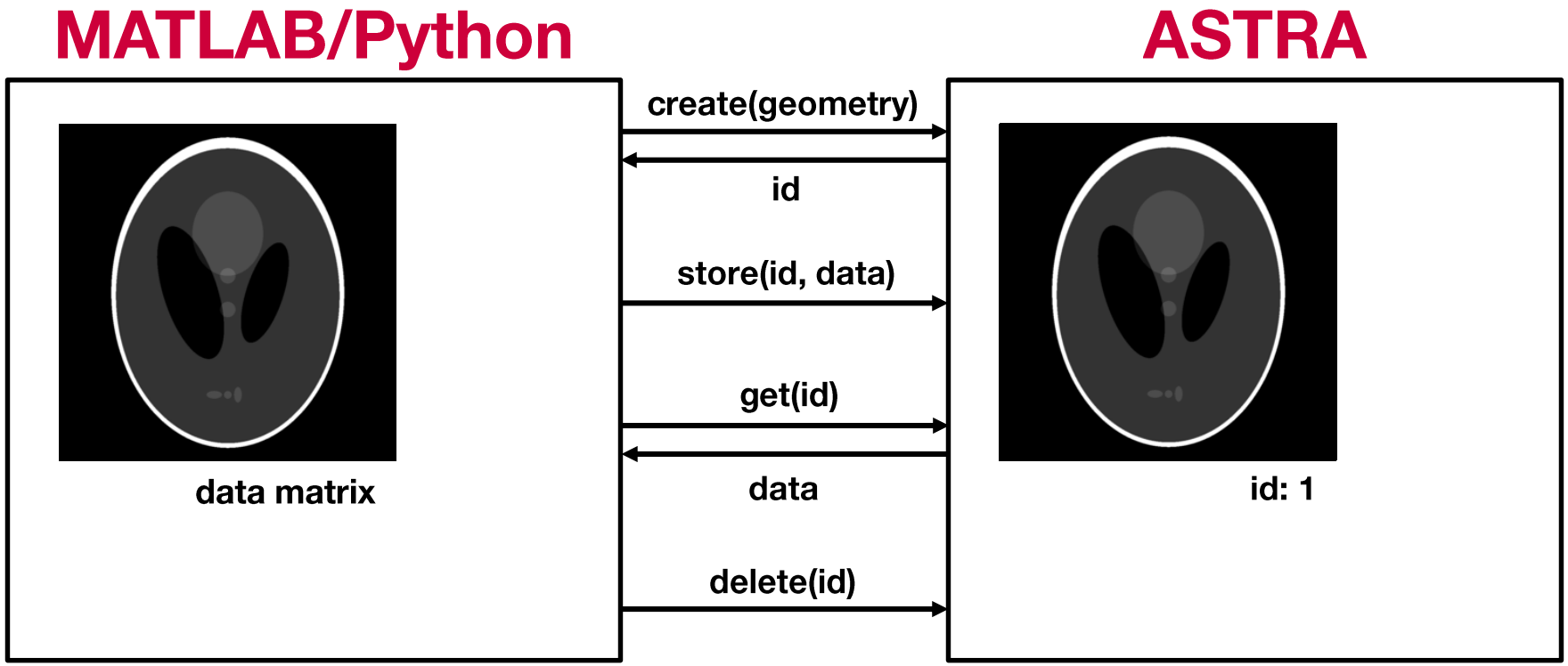
Data

Place to store data (e.g. reconstruction, projection data)

- In ASTRA (C++): float*
- In MATLAB/Python: id

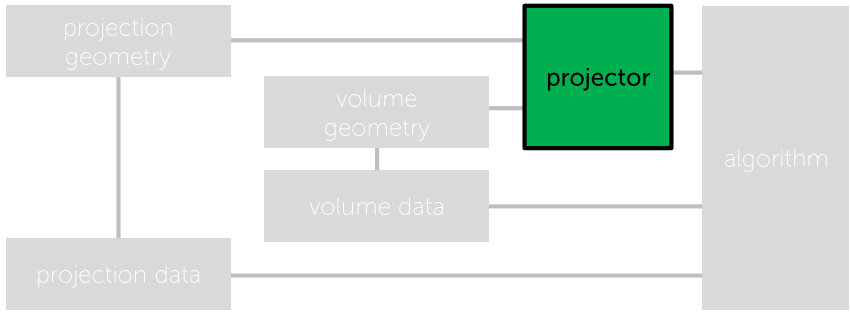
Links to geometry



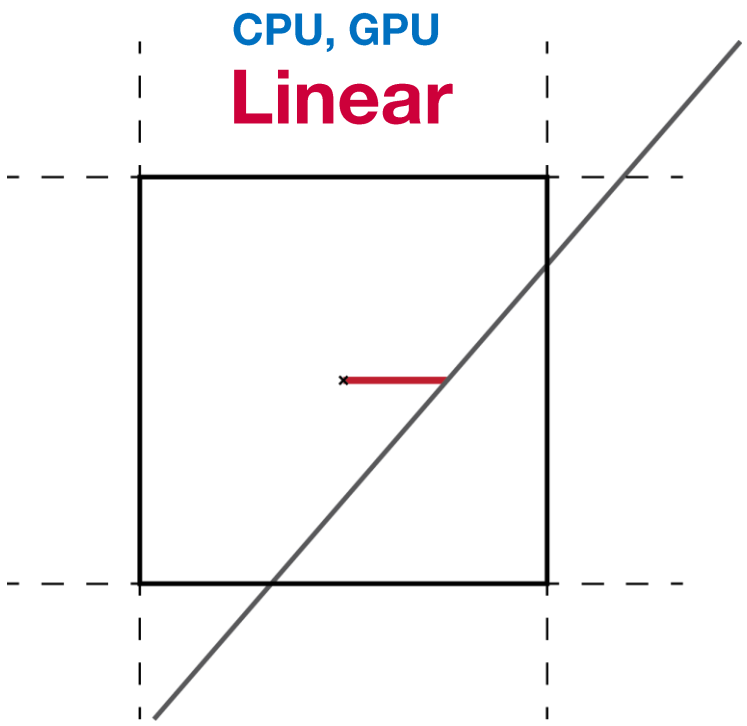


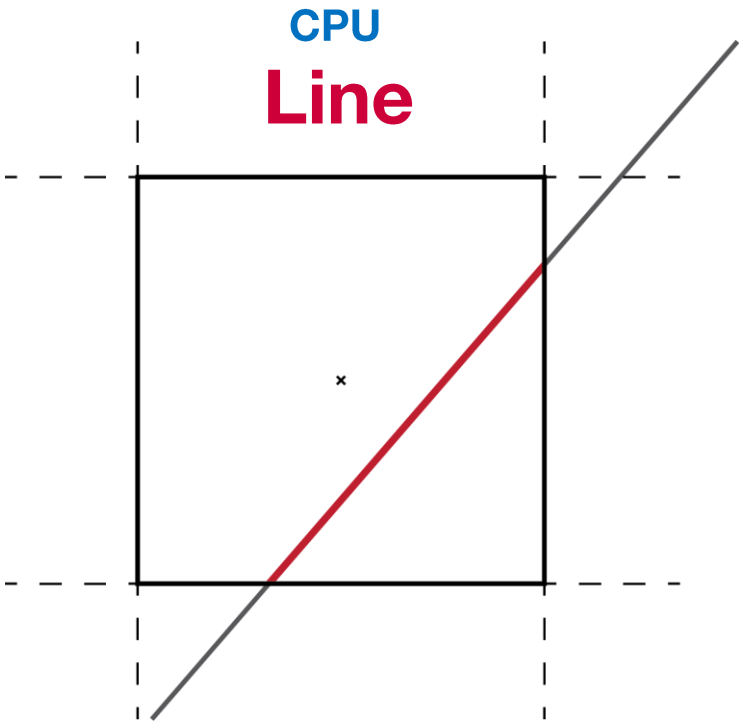
Projector

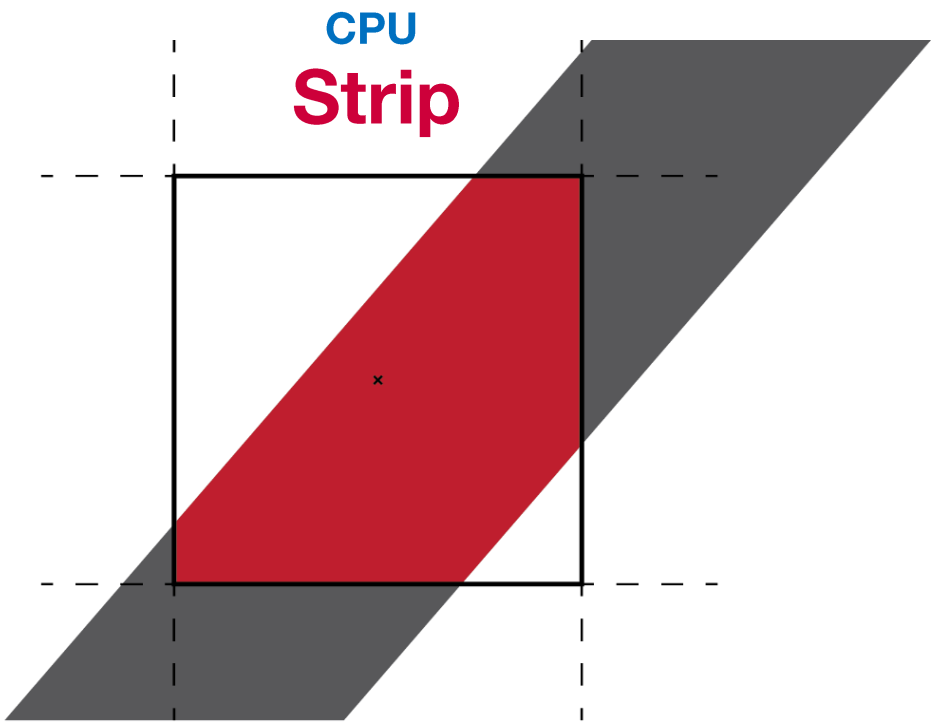
- Links volume geometry to projection geometry
- Defines and computes projection matrix



Forward projection $p = Wv$ Backprojection $v = W^T p$

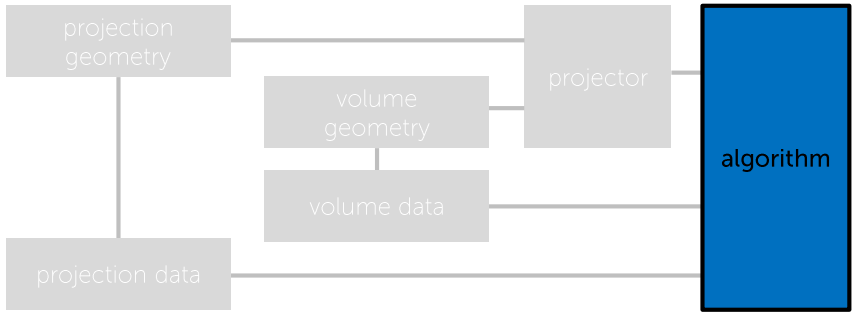




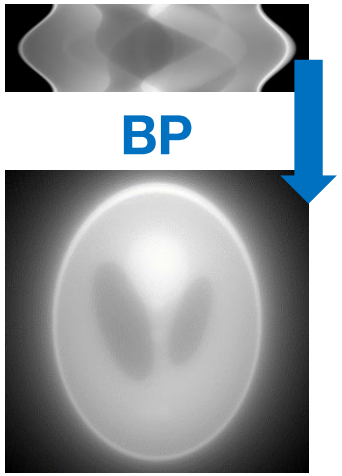


Algorithms

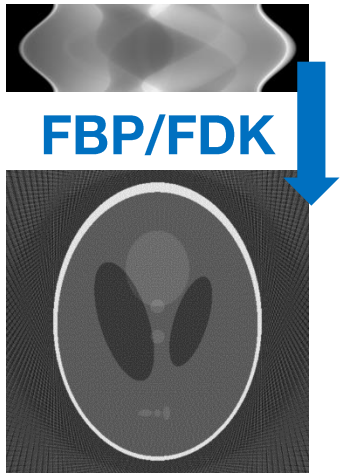
Where the actual computations happen



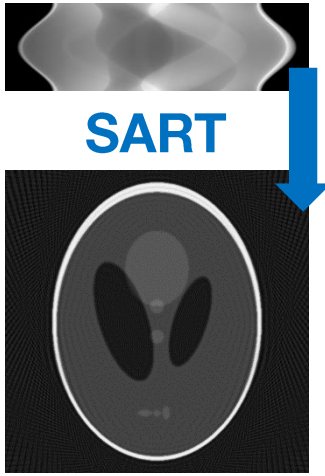
Projection

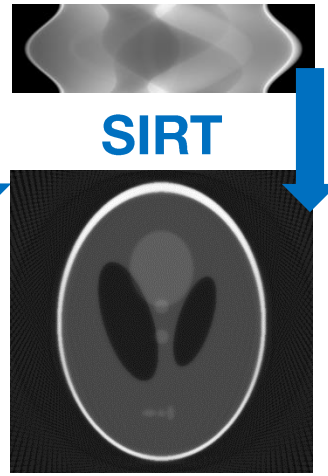


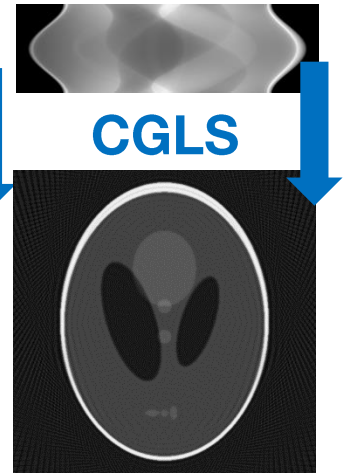
Analytical Reconstruction



Algebraic Reconstruction



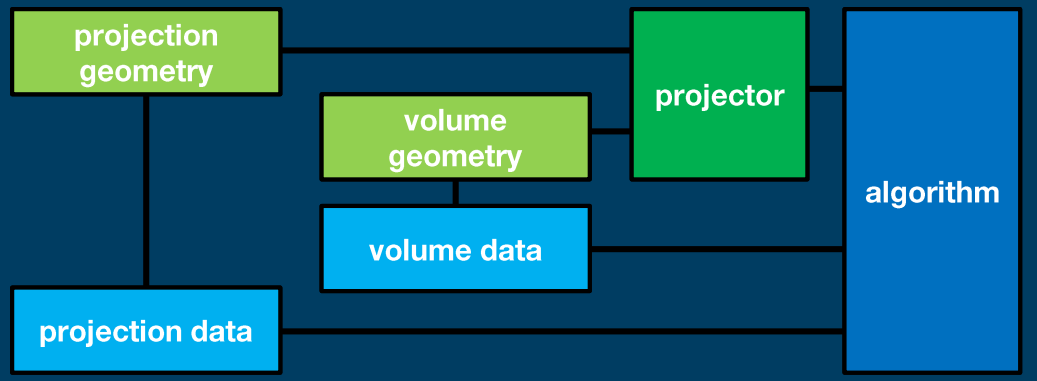




Options: reconstruction masks, min/max constraints, …

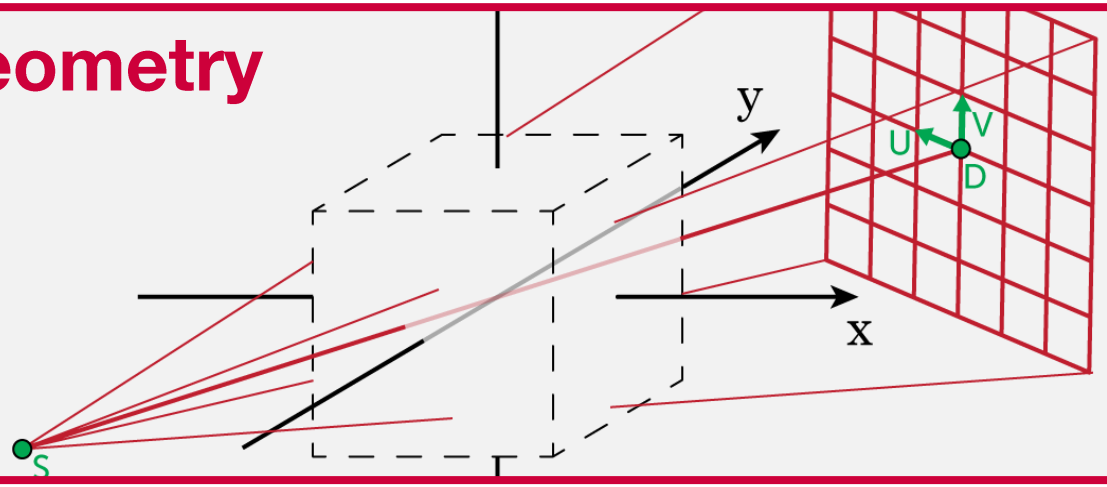
Code example

```
% create geometries and projector
vol_geom = astra_create_vol_geom(512, 512);
proj_geom = astra_create_proj_geom('parallel', 1, 512, linspace2(0,pi,180));

% create projector
proj_id = astra_create_projector('linear', proj_geom, vol_geom);

% store projection data into the toolbox and create a data object for the reconstruction
sino_id = astra_mex_data2d('create', '-sino', proj_geom, data);
rec_id = astra_mex_data2d('create', '-vol', vol_geom, 0);

% set up the parameters for a reconstruction algorithm
cfg = astra_struct('SIRT');
cfg.ProjectorId = proj_id;
cfg.ReconstructionDataId = rec_id;
cfg.ProjectionDataId = sino_id;
alg_id = astra_mex_algorithm('create', cfg);

% run 20 iterations of the algorithm and retrieve the result
astra_mex_algorithm('iterate', alg_id, 20);
rec = astra_mex_data2d('get', rec_id);
```

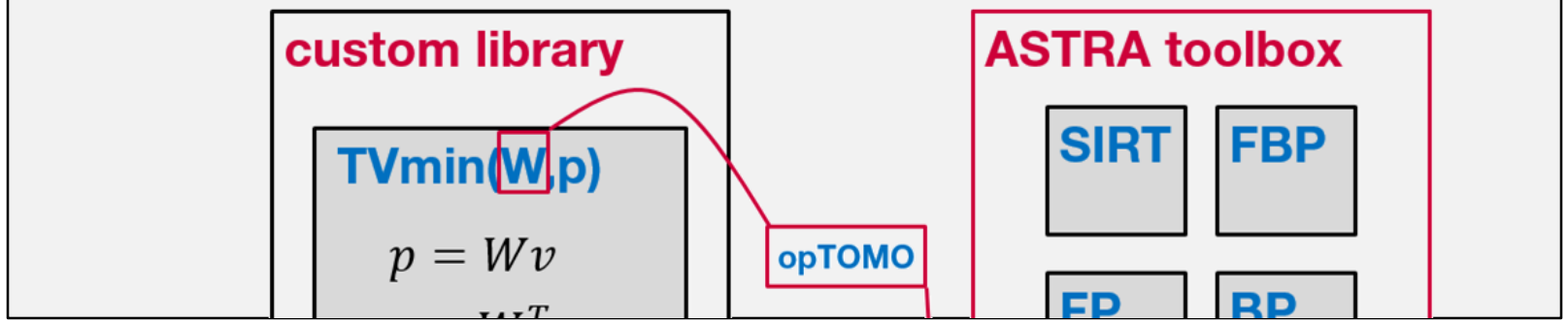


Advanced Features

Vector Geometry



Flexible Building Blocks



Easy to use
MATLAB and Python interface

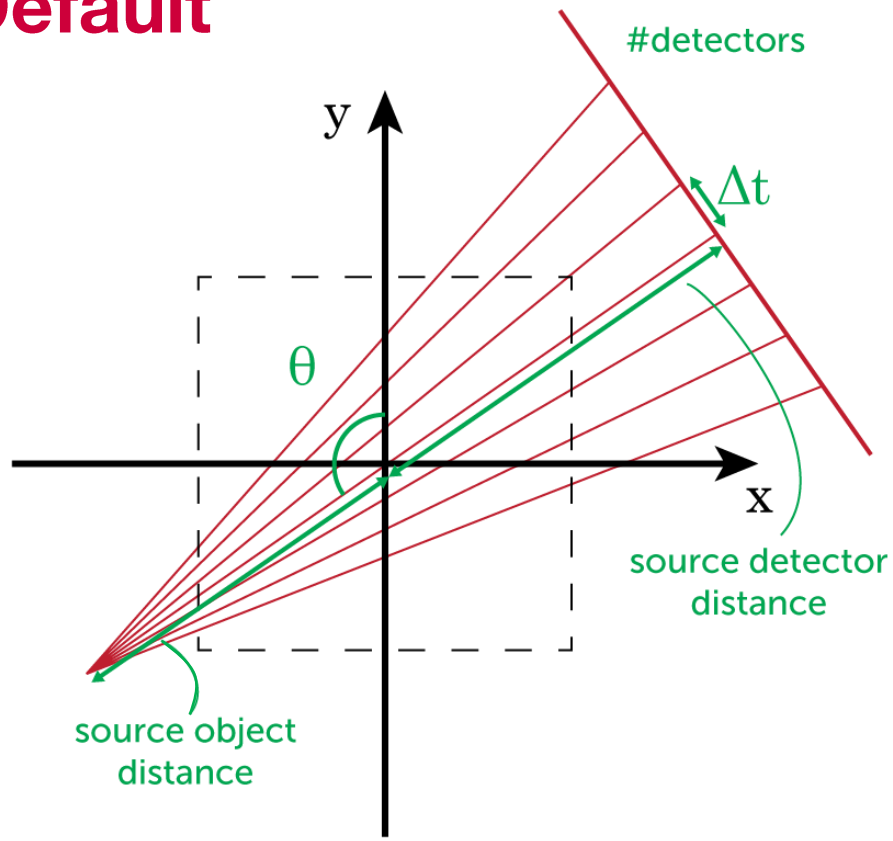
Broad support
algorithms and geometries

Flexible
building block for custom algorithms

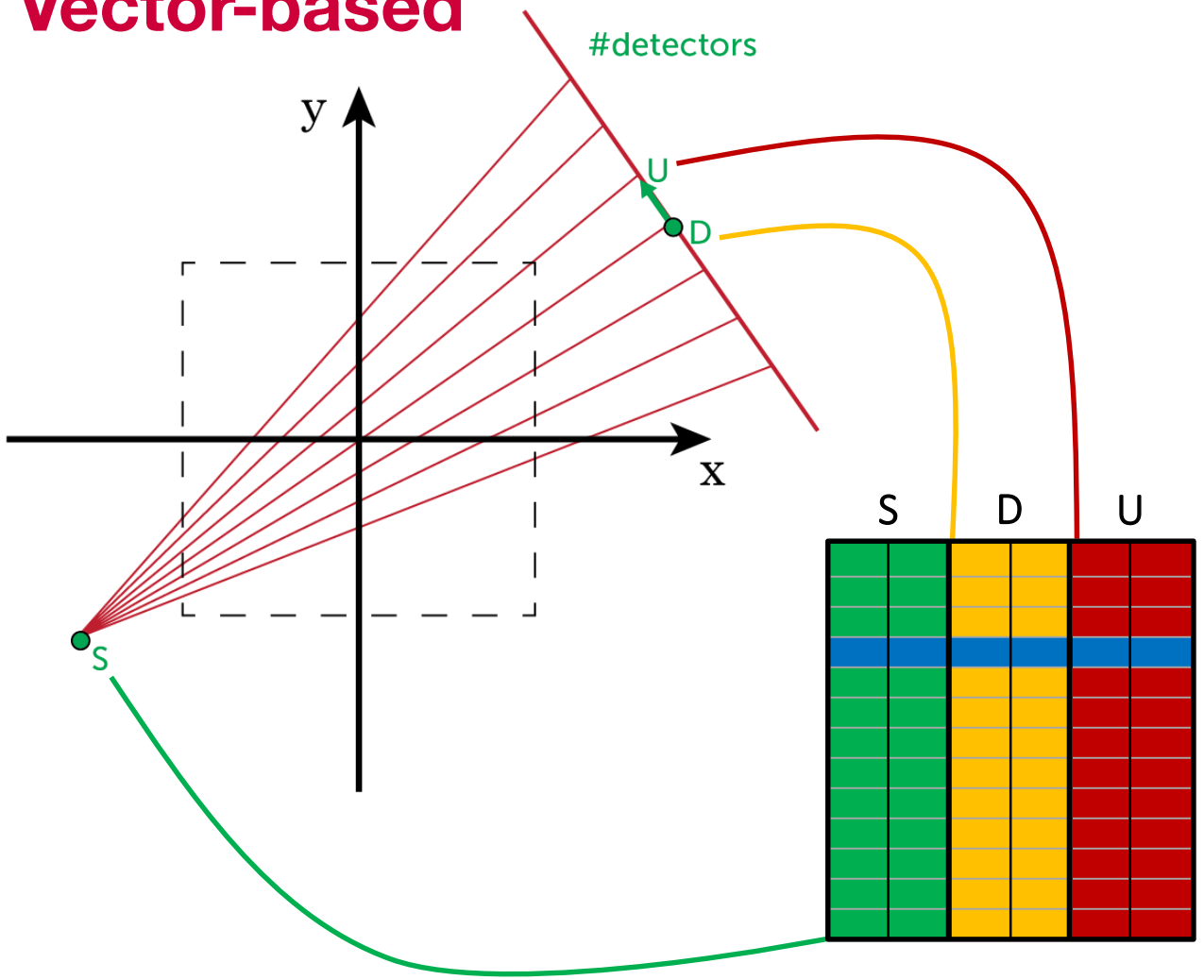
Powerful
C++ and CUDA

Vector Geometry

Default

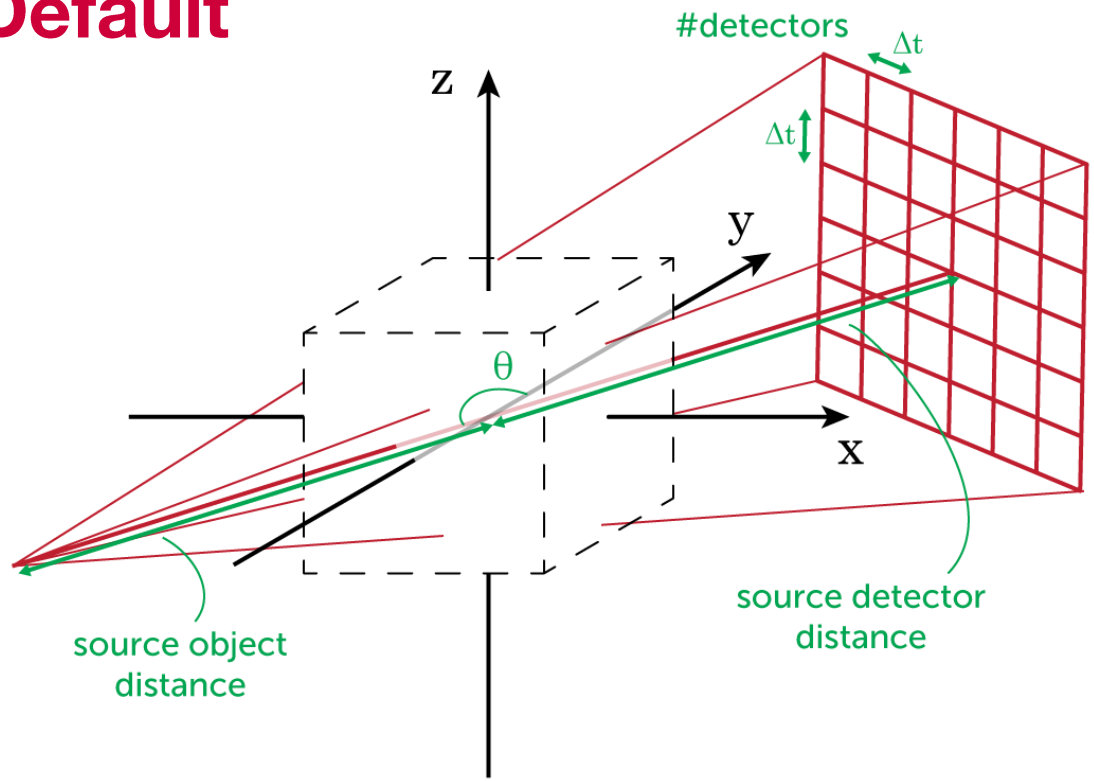


Vector-based

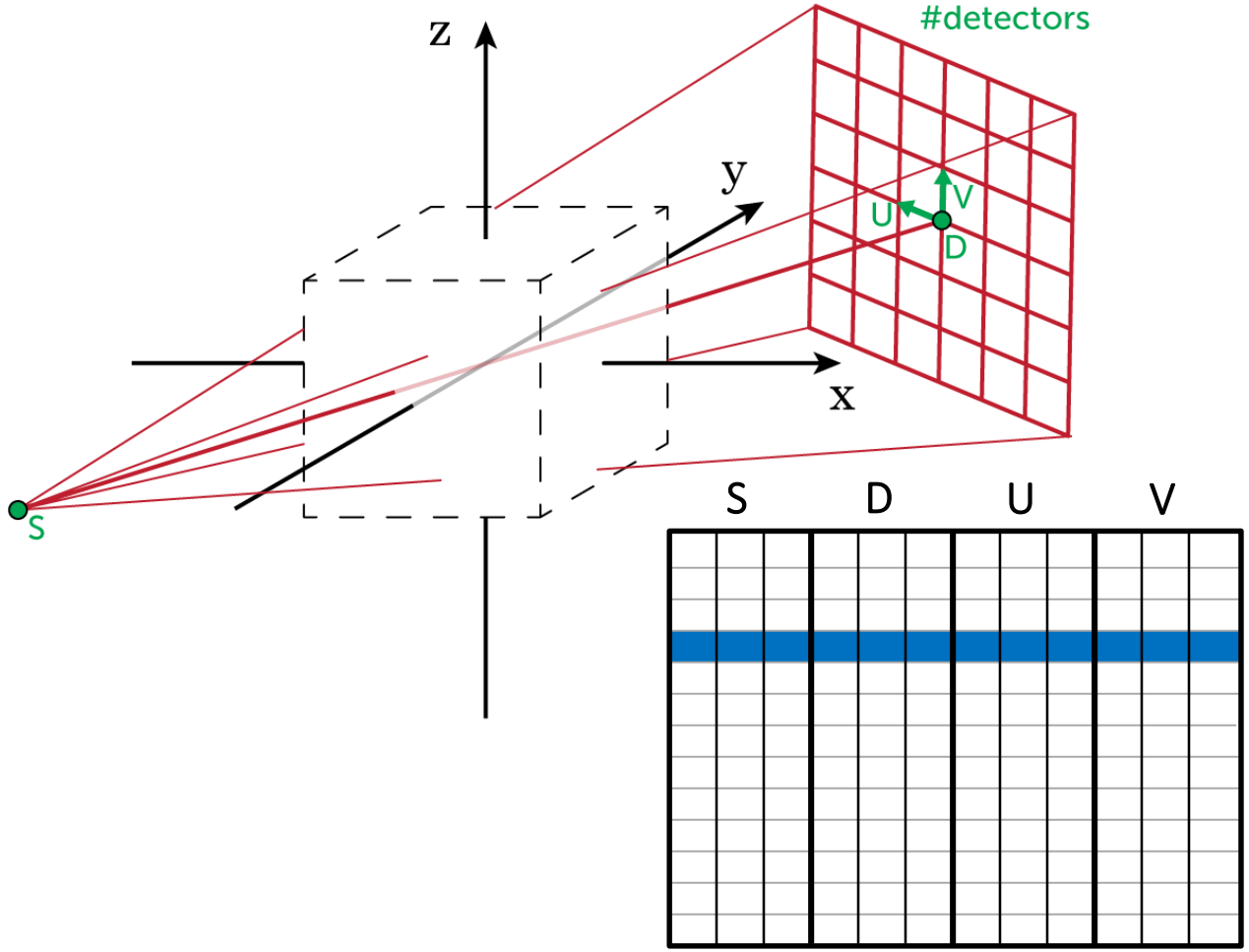


Vector Geometry

Default



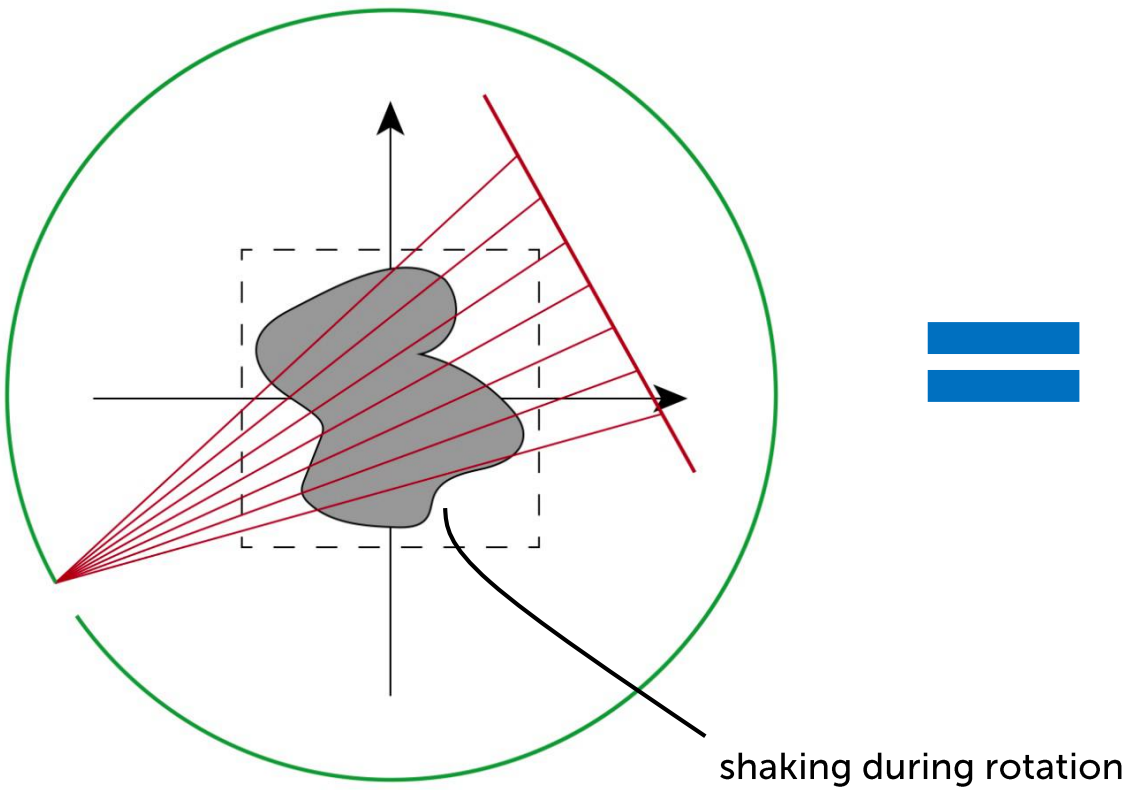
Vector-based



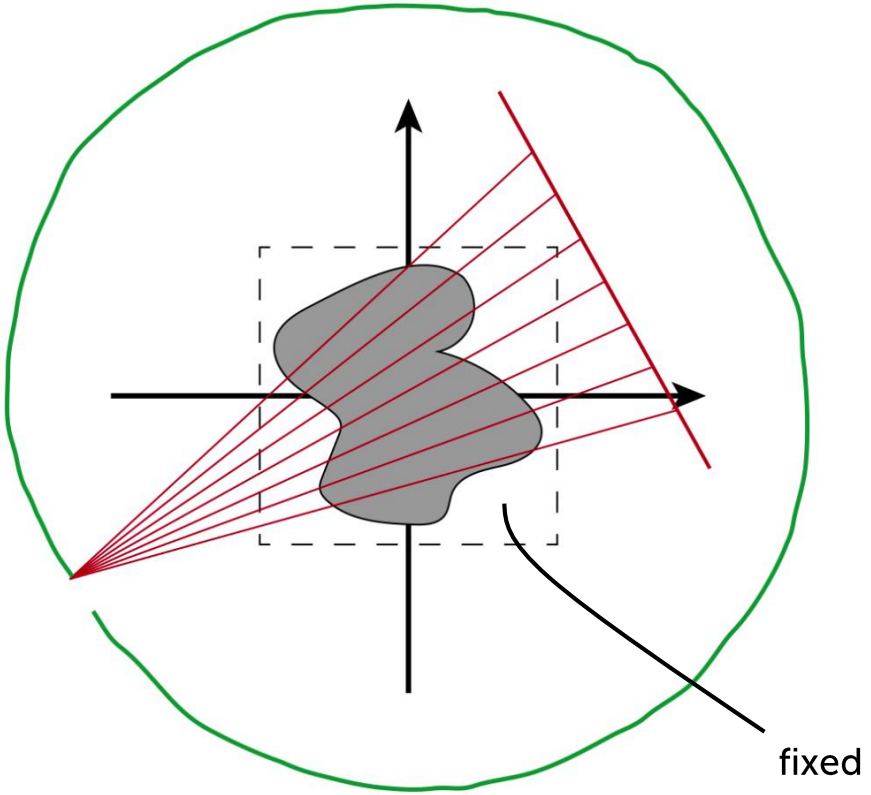
Unconventional Geometry

Object Movement

G. Van Eyndhoven, et al., "Combined Motion Estimation and Reconstruction in Tomography", 12th European Conference on Computer Vision, vol. 7583, Firenze, Italy, Lecture Notes on Computer Science, pp. 12-21, 2012
V. Van Nieuwenhove, et al., "Affine deformation correction in cone beam Computed Tomography", Fully Three-Dimensional Image Reconstruction in Radiology and Nuclear Medicine, Newport, Rhode Island, USA, pp. 182-185, 2015



=



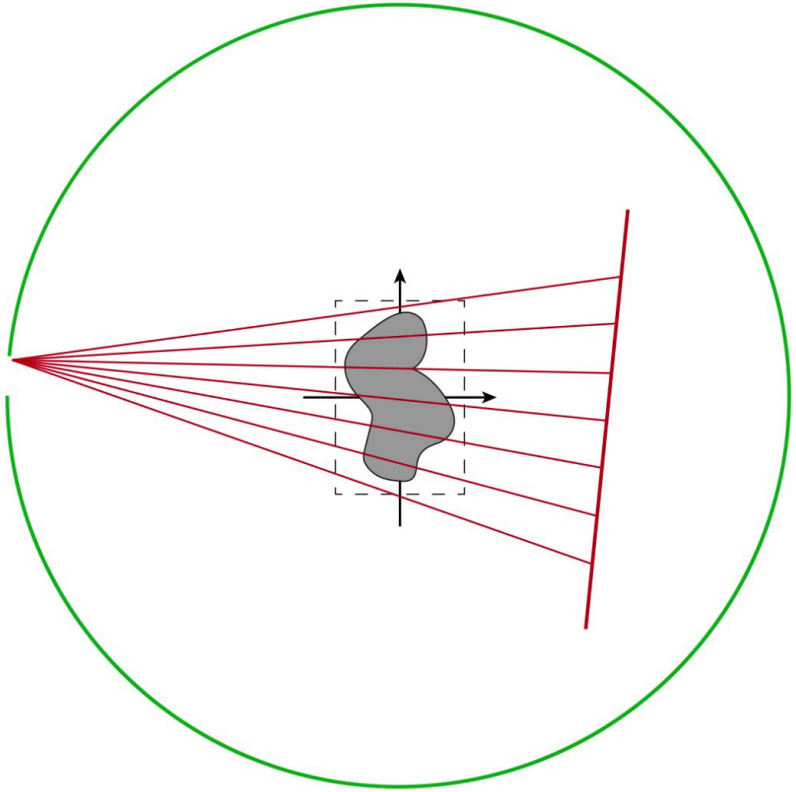
fixed source, moving object

moving source, fixed object

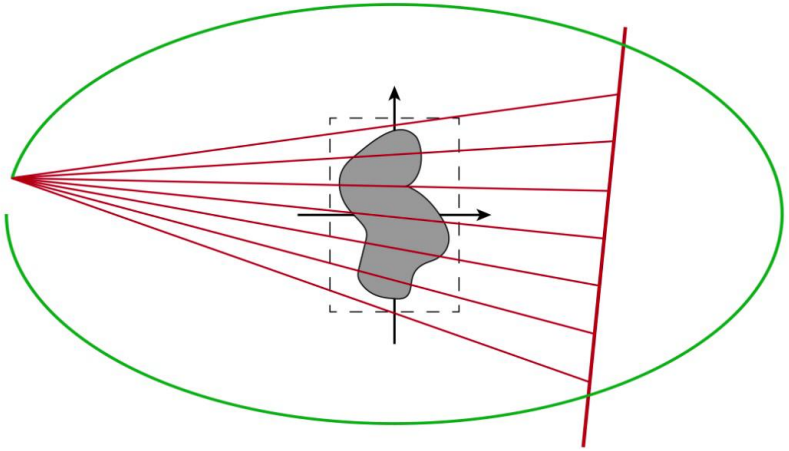
Unconventional Geometry

Elongated Object

A. Dabravolski, K. J. Batenburg, and J. Sijbers, "Adaptive zooming in X-ray computed tomography", Journal of X-Ray Science and Technology, vol. 22, no. 1, pp. 77-89, 2014.



fixed zooming

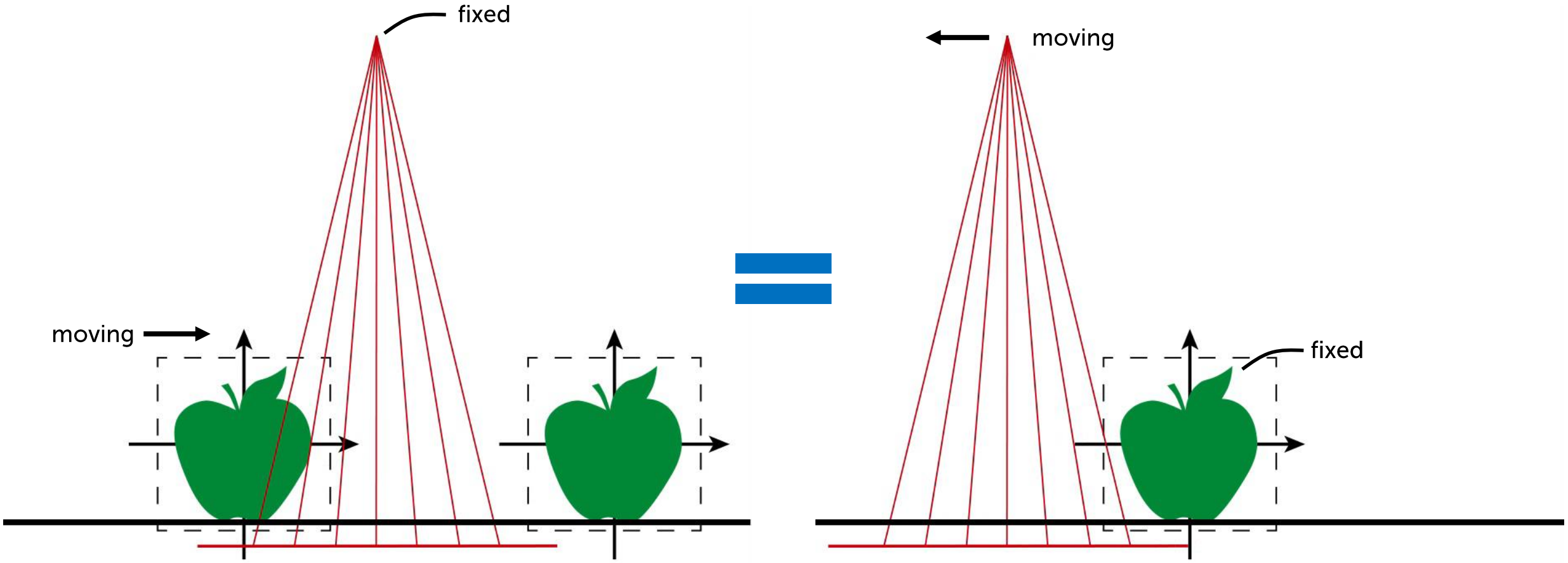


adaptive zooming

Unconventional Geometry

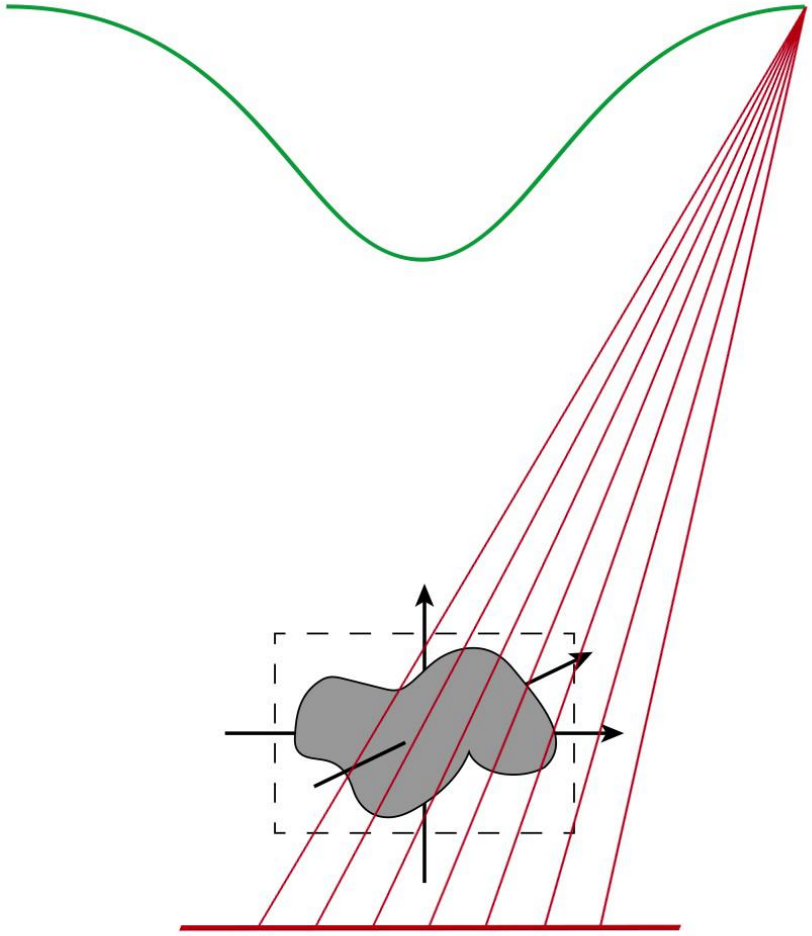
Conveyor Belt Tomography

E. Janssens, et al., "Neural Network Based X-Ray Tomography for Fast Inspection of Apples on a Conveyor Belt", IEEE International Conference on Image Processing, (2015)
L. F. Alves Pereira, et al., "Conveyor belt X-ray CT using Domain Constrained Discrete Tomography", Sibgrapi conference on Graphics, Patterns and Images, pp. 290 - 297, (2014)

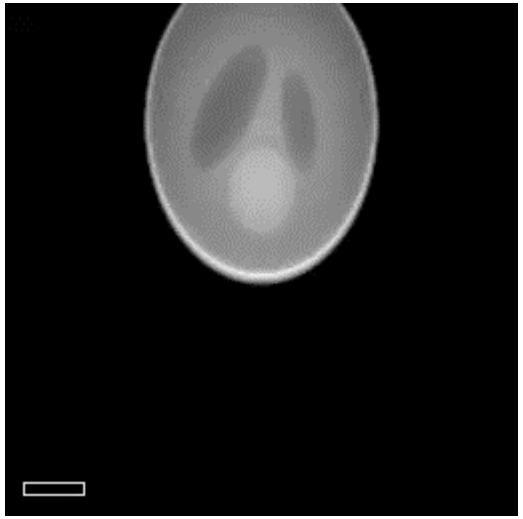


Vector Geometry

Code example

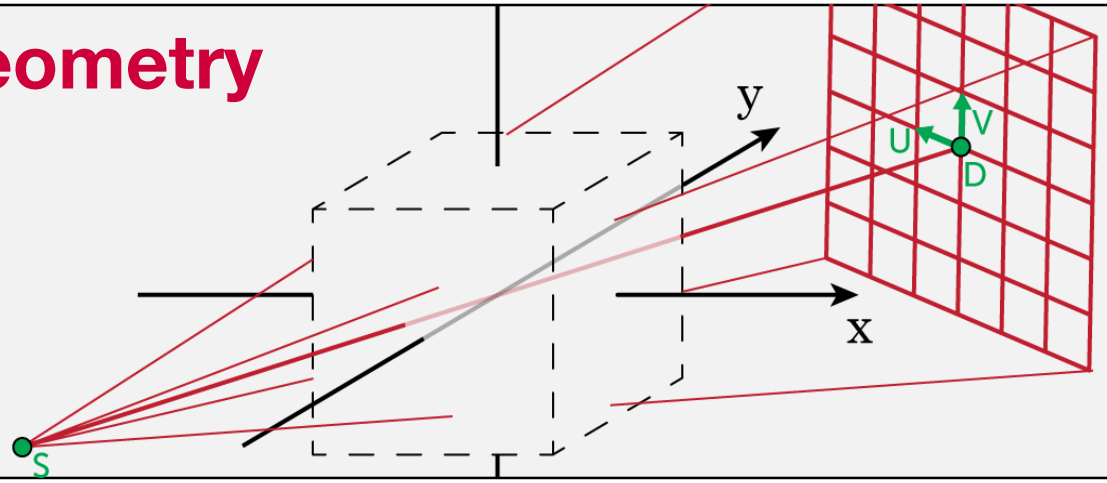


```
vectors = zeros(100,12);
vectors(:,1:3) = [linspace(-200,200,100)' zeros(100,1)
                  600+300*sin(linspace(0,2*pi,100))'];  % S
vectors(:,4:6) = repmat([0 0 -200], [100 1]);           % D
vectors(:,7:9) = repmat([1 0 0], [100 1]);              % U
vectors(:,10:12) = repmat([0 1 0], [100 1]);            % V

proj_geom = astra_create_proj_geom('cone_vec',256,256,vectors);
vol_geom = astra_create_vol_geom(128,128,128);

[~, proj] = astra_create_sino3d_cuda(data, proj_geom, vol_geom);
```

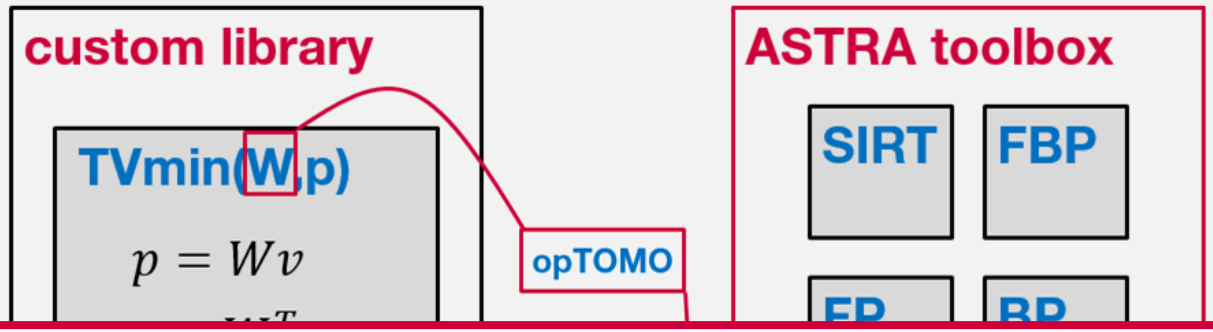


Advanced Features

Vector Geometry



Flexible Building Blocks



Easy to use
MATLAB and Python interface

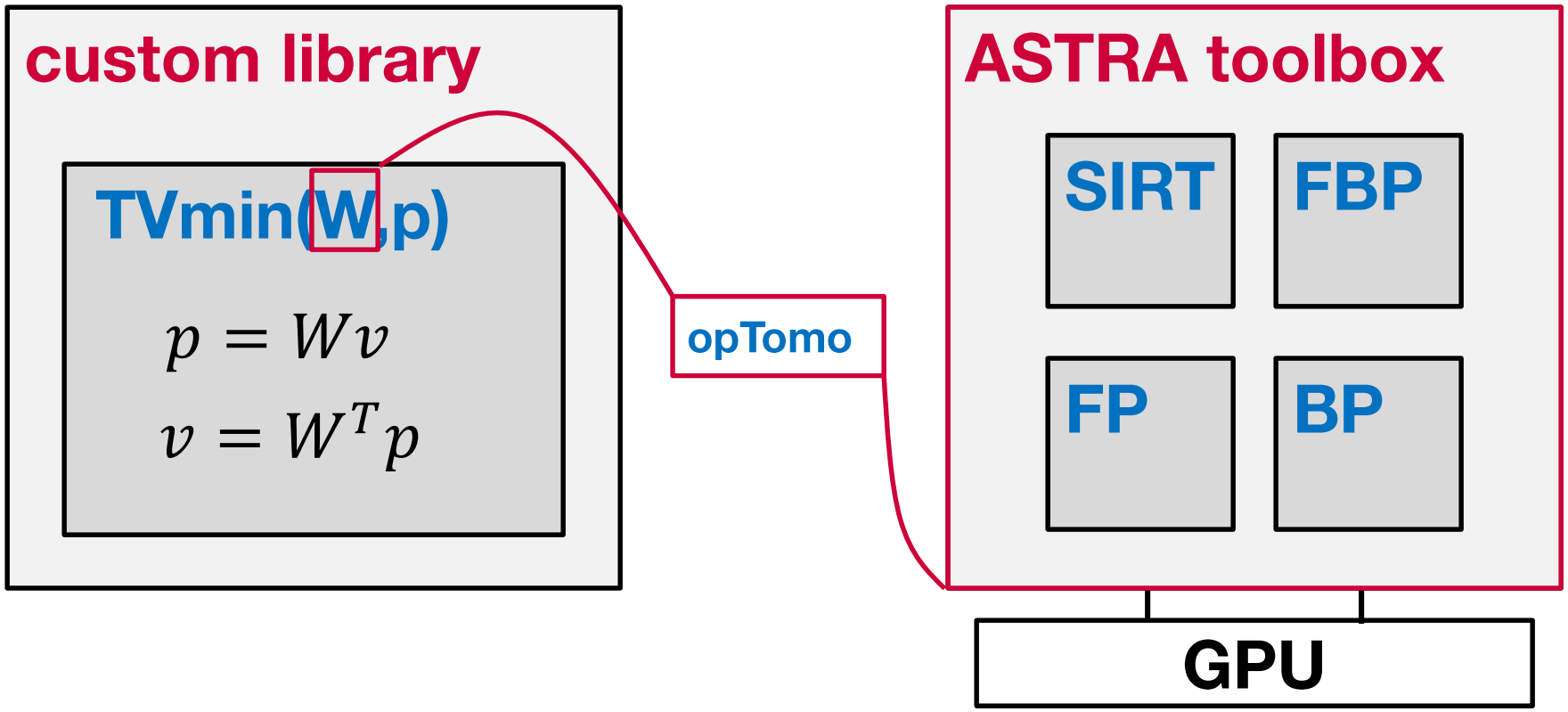
Broad support
algorithms and geometries

Flexible
building block for custom algorithms

Powerful
C++ and CUDA

Flexible Building Blocks

Core building blocks (FP and BP) of the ASTRA Toolbox can be used independently of the rest



Flexible Building Blocks

original script

```
function v = SIRT(W, p, v0, iters)
   R = 1 ./ W*ones(size(W,1));
   C = 1 ./ W'*ones(size(W,2));
   v = v0;
   for k = 1:iters
      v = v + C.*(W'*(R.*(p-W*v)));
   end
end
```

```
W = astra_projection_matrix('line', proj_geom, vol_geom);
v = SIRT(W, p, v0, iters);
```

ASTRA opTomo

```
W = opTomo('cuda', proj_geom, vol_geom);
v = SIRT(W, p, v0, iters);
```

ASTRAified script

```
function v = SIRT_ASTRAified(proj_geom, vol_geom, p, v0,iters)
   [~, tmp] = astra_fp(ones(size(v0)), proj_geom, vol_geom);
   R = 1 ./ tmp;
   [~, tmp] = astra_bp(ones(size(p)), proj_geom, vol_geom);
   C = 1 ./ tmp;
   v = v0;
   for k = 1:iters
      [~, fp] =  astra_fp(v, proj_geom, vol_geom);
      [~, upd] =  astra_bp(R.*(p-fp), proj_geom, vol_geom);
      v = v + C .* upd;
   end
end
```

```
v = SIRT_ASTRAified(vol_geom, proj_geom, p, v0, iters);
```

note: some ASTRA function names have been shortened for brevity

Flexible Building Blocks

CWI

Universiteit Antwerpen

Timings

128x128 volumes
100 SIRT iterations

```
W = astra_projection_matrix('line', proj_geom, vol_geom);
v = SIRT(W, p, v0, iters);
```

0.71s

```
v = SIRT_ASTRAified(vol_geom, proj_geom, p, v0, iters);
```

0.53s

```
W = opTomo('cuda', proj_geom, vol_geom);
v = SIRT(W, p, v0, iters);
```

0.56s

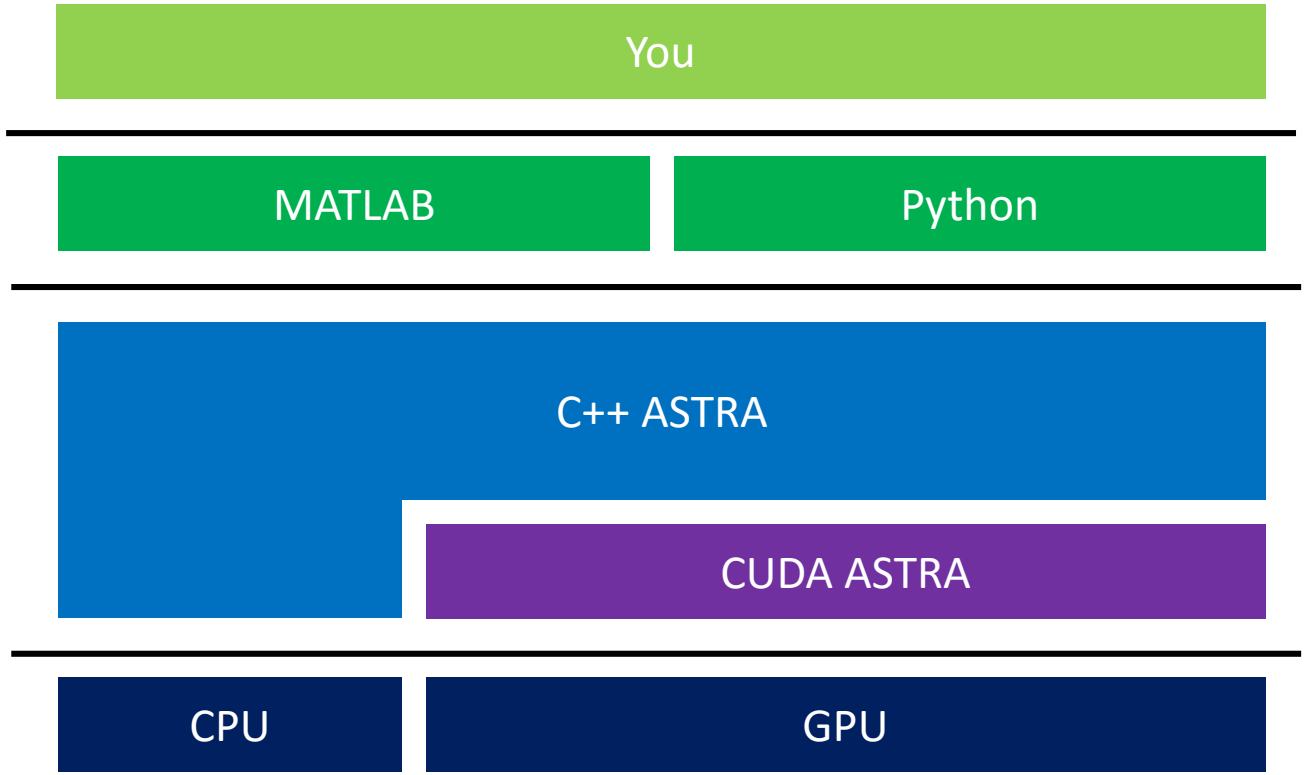
```
astra_struct('SIRT');
```

1.73s

```
astra_struct('SIRT_CUDA');
```

0.24

Conclusion



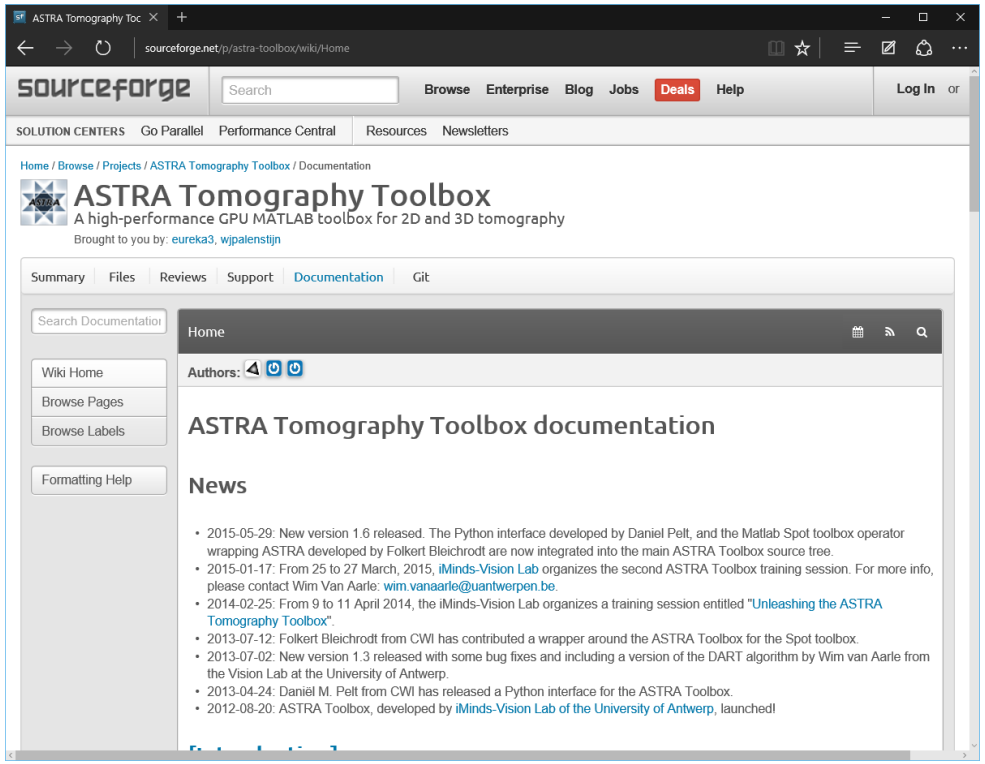
Easy to use
MATLAB and Python interface

Broad support
algorithms and geometries

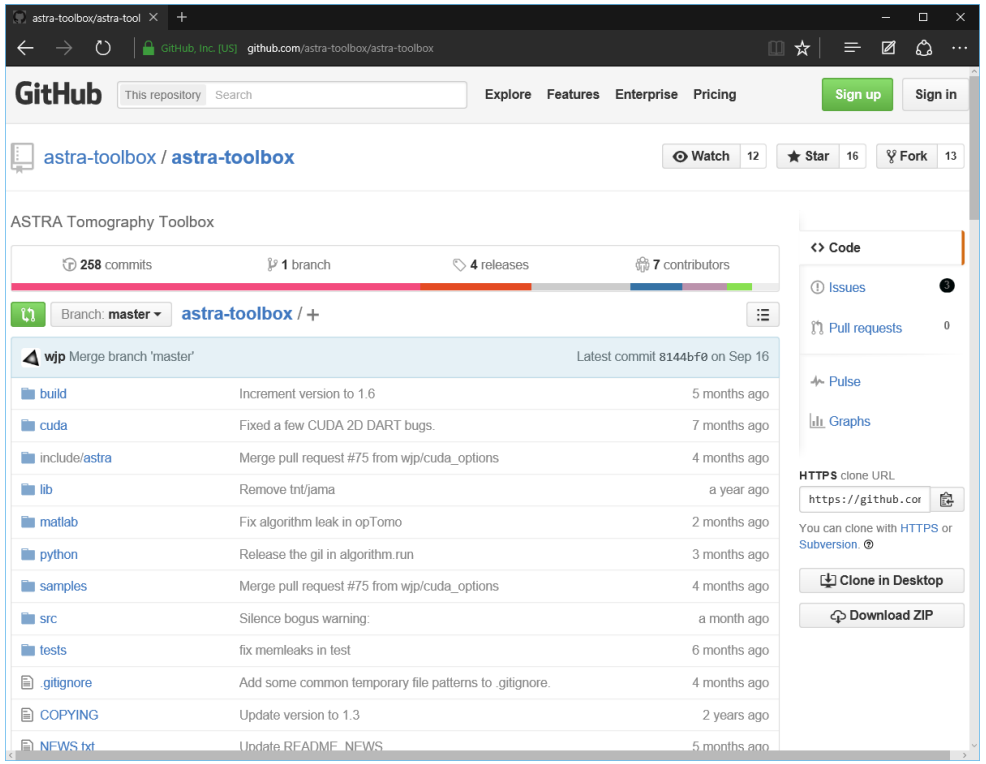
Flexible
building block for custom algorithms

Powerful
C++ and CUDA

More information



http://sourceforge.net/p/astra-toolbox/wiki



https://github.com/astra-toolbox

wim.vanaarle@uantwerpen.be