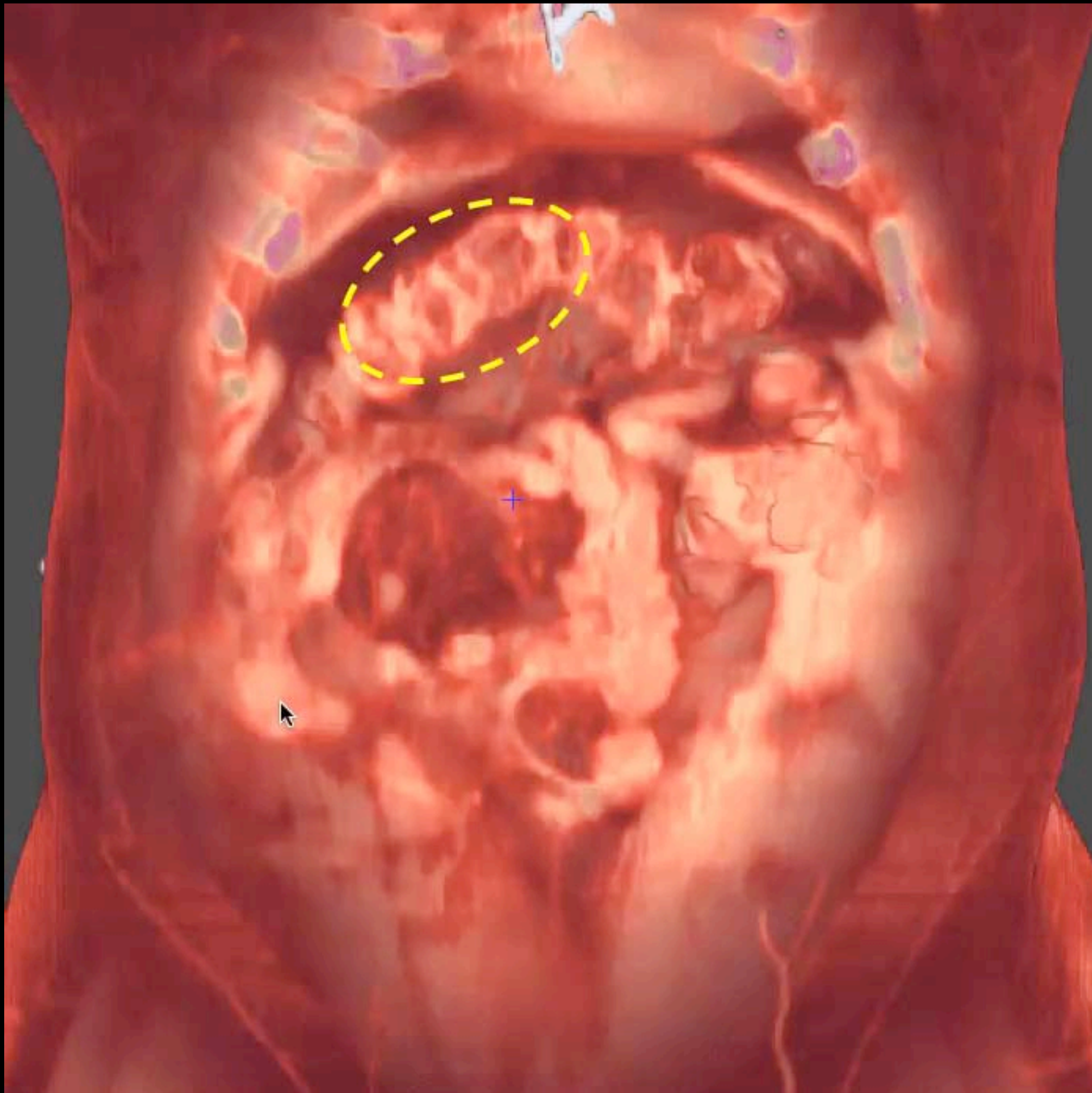
Shadie - A Domain Specific Language for Radiation Oncology

Hanspeter Pfister
pfister@seas.harvard.edu

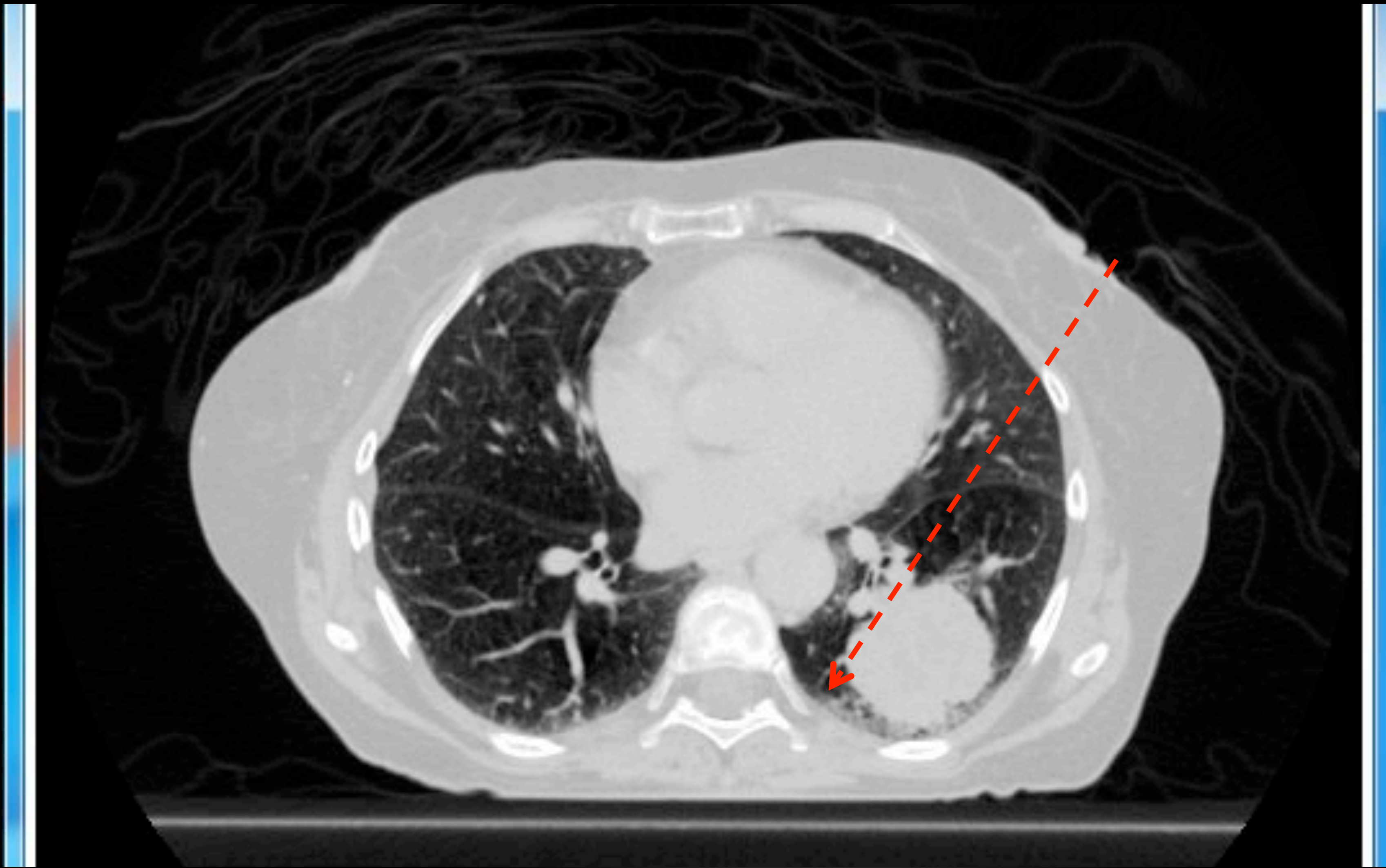
The Team

- MGH / HMS Radiation Oncology Group
  - George Chen
  - John Wolfgang


- School of Engineering and Applied Sciences
  - Milos Hasan (UC Berkeley)
  - Hanspeter Pfister

Quantitative 4D Image Understanding



GTV: 4cm dia, 5cm long

S/I Motion: 1.25cm

- Shader to display range from BEV source to distal ITV
- rpl.mov
- Green less range, brown greater

Applications

- Multiple data sets (serial CT)
- Visualize multimodality data sets (PET, functional imaging)
- Calculate dose on the fly
- Calculate distances, volumes as a fcn of BEV
- VISUALIZE UNCERTAINTY!!!

What Scientists Want

- Focus on the code that solves their problem
- Being able to tweak and change code
- Getting good speedup from GPUs

The Challenge

- Scientists are not CUDA hackers ...

```
// Determinte the size of each block (of threads)
dim3 dimBlock(BLOCKDIM_X,
      BLOCKDIM_Y,
      1);

// Determinte the size of the grid (of blocks)
dim3 dimGrid( (int) ceil((float)width/(float)BLOCKDIM_X),
      (int) ceil((float)height/(float)BLOCKDIM_Y),
      1);

// Allocate some device memory for the output
uchar4 *d_out;
cudaMalloc((void **)&d_out, sizeof(uchar4)*width*height);

// Setup texture and 2D array
texRef4.addressMode[0] = cudaAddressModeClamp;
texRef4.addressMode[1] = cudaAddressModeClamp;
texRef4.filterMode = cudaFilterModePoint;
texRef4.normalized = false;
cudaArray *d_tex;
cudaMallocArray( (cudaArray**)&(d_tex), &uchar4Desc, width, height );
cudaMemcpyToArray(d_tex, 0, 0, _in, sizeof(uchar4)*width*height,
cudaMemcpyHostToDevice);
cudaBindTextureToArray(texRef4, d_tex, uchar4Desc);

// Launch the CUDA kernel
kernel_boxcar_texture<<< dimGrid, dimBlock >>>(d_out, width, height,
halfkernelsize);
cutilCheckMsg("kernel_boxcar_texture execution failed.\n");
```

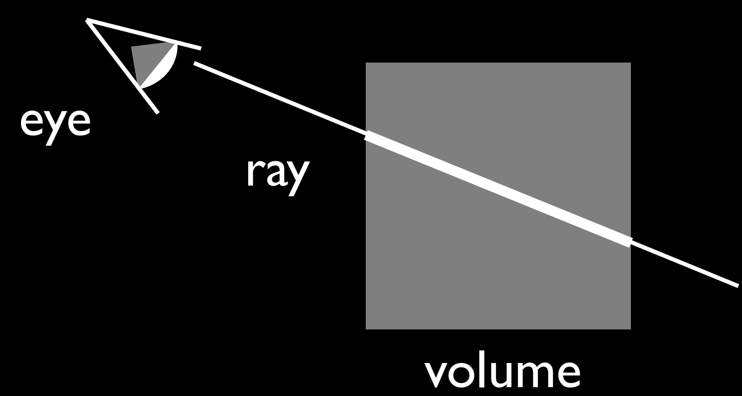
The current reality of volume visualization

file format parsing

compilation and linking

user interface

resource allocation



memory corruption debugging

CPU-GPU communication

data interpolation

progressive refinement

Enter: DSLs

- Domain Specific Languages (DSLs) promise the solution
- Two approaches:
  - Provide environment to develop DSLs (Stanford)
  - Develop small DSLs for specific target domains (Harvard)

DSLs for GPUs

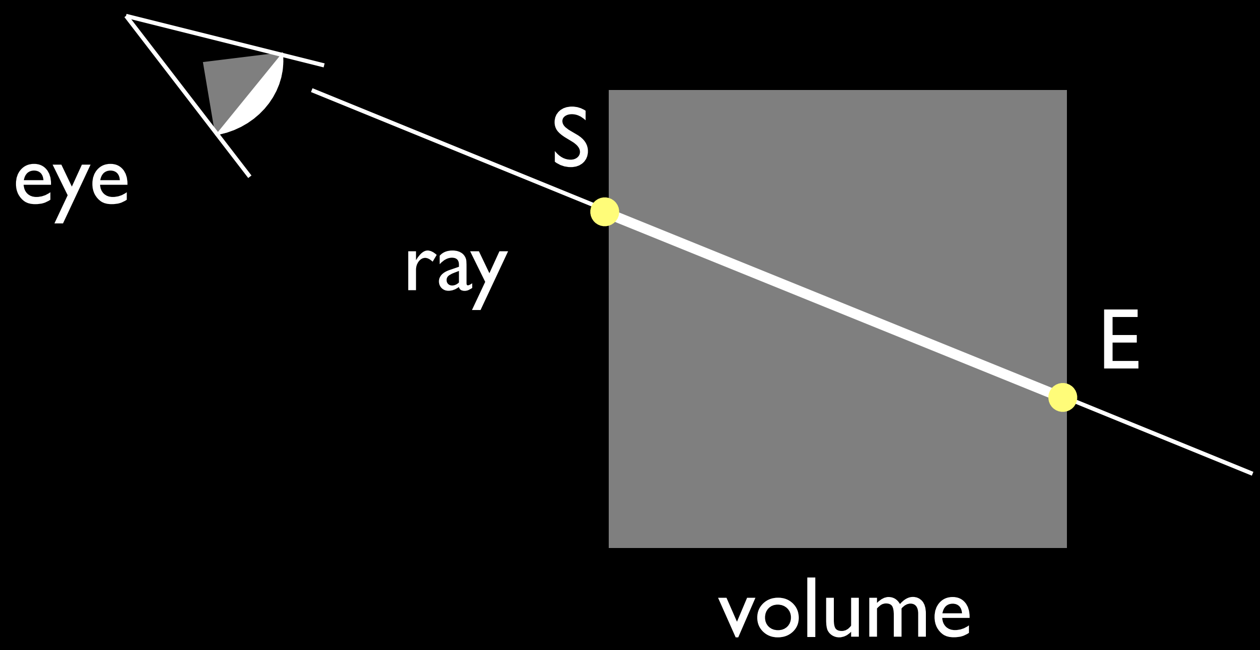
- Restrict ourselves to something modest but feasible
- A high-level language for GPU computing?
  - We don’t know the solution yet.
- A high-level language for GPU visualization of volumetric data for radiation therapy purposes?
  - Yes!

Shadie: A DSL for Radiation Oncology

PLUNC
3D Slicer

Shadie

The Ray Model



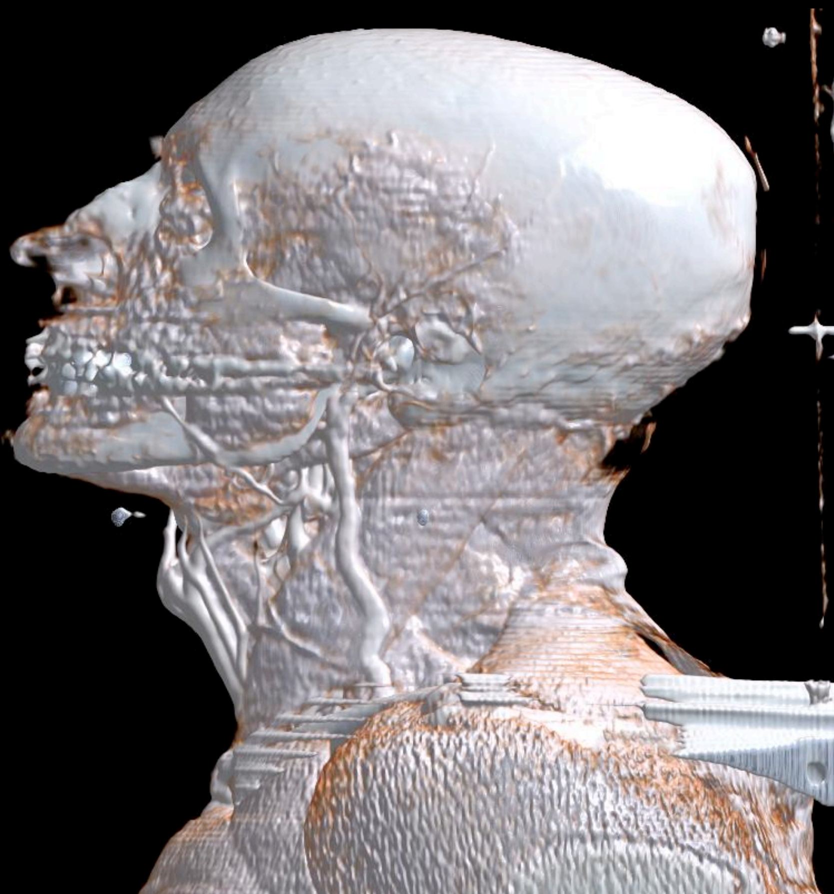
User supplies a function f : (RayStart, RayEnd) → Color

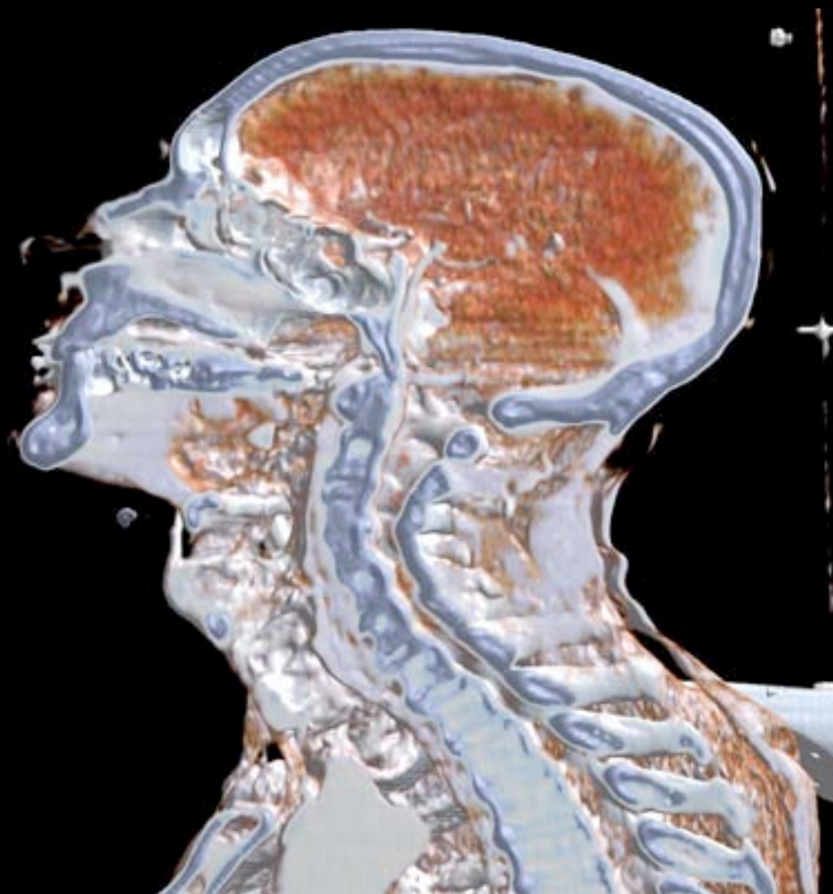
Maximum intensity projection (MIP)

MIP: The Shader

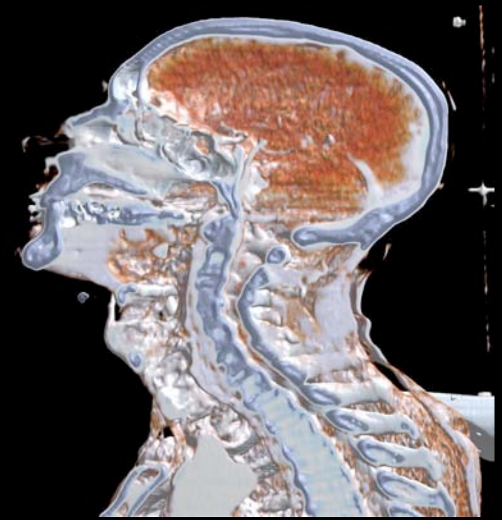
```
data = data3d('data/ct', 'short')
m = 0.0

for t in linspace(0.0, 1.0, 1000):
    # find position along ray
    P = (1-t) * S + t * E

    # update maximum
    m = max(m, linear_query_3d(data, P))

return m
```

Phong lighting + cut plane





Phong lighting + cut plane



```
# query CT
density = cubic_query_3d_cut(data, P, D, cut)

# apply transfer function
tf_query = (density - tf_pos) / tf_width
if tf_query < 0: continue
rgba = linear_query_1d_rgba(tf, tf_query*2 - 1)

# apply phong shading
N = -normalize(cubic_gradient_3d_cut(data, P, D, cut))
L = normalize(lightpos - P)
color = phong(L, N, C, rgba.xyz, 1, 50, 0.5)
```

CT + Dose Iso-Surface

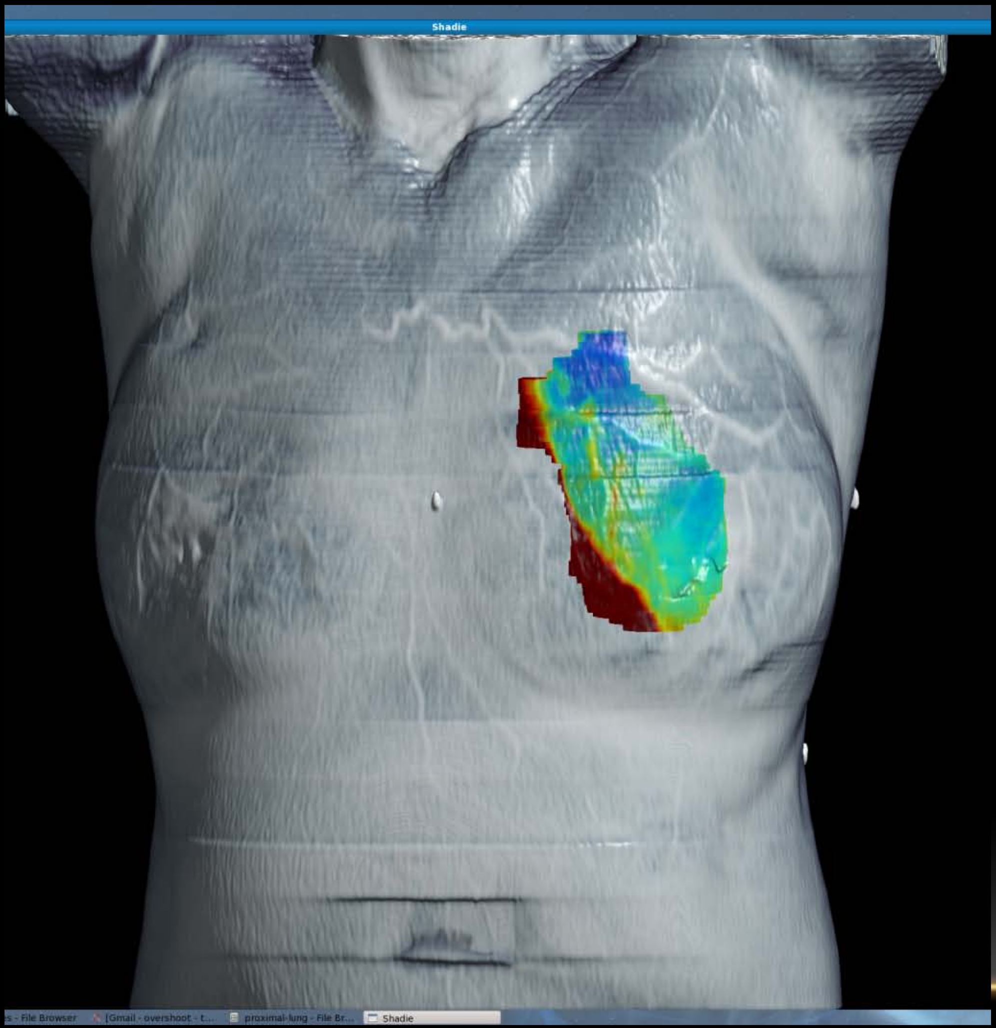
Multi-pass computation

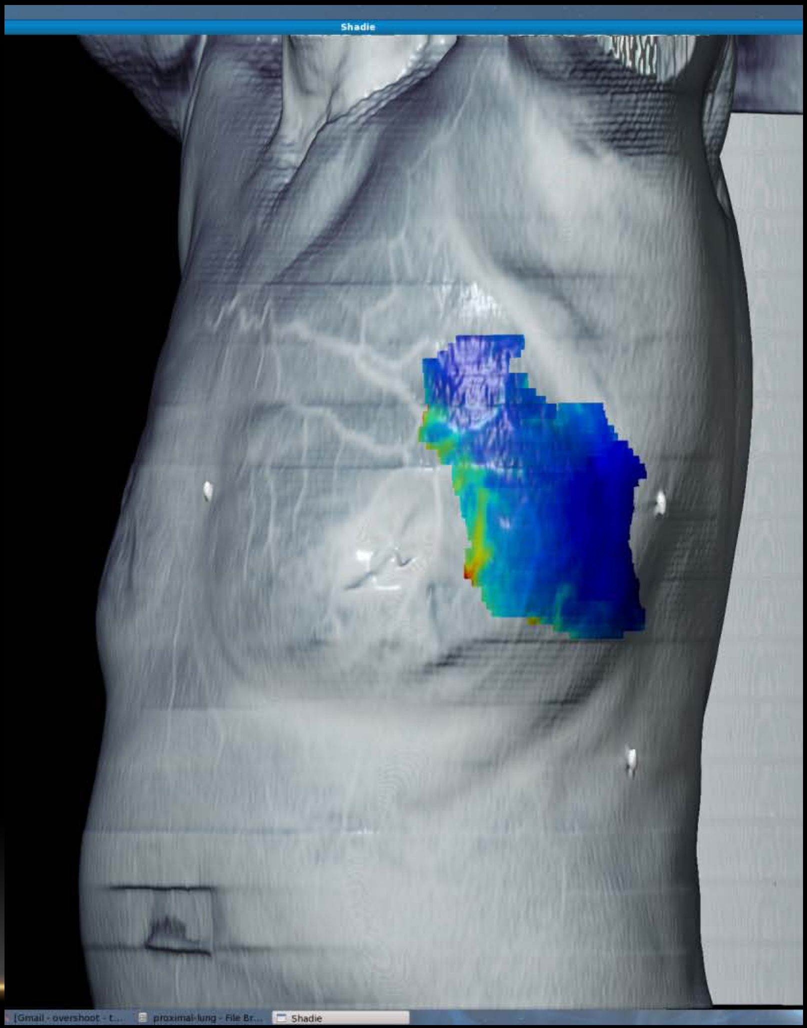
```
ct = data3d(...)

def compute_rpl_slice(X, prev_slice):
    # RPL computation at point X...

rpl = compute3d(compute_rpl_slice, rpl_size)

def compute_dose(X):
    # dose computation at point X...

dose = compute3d(compute_dose, dose_size)

def main(X, D):
    # the main visualization function
    # compute RGBA value based on CT and dose
```

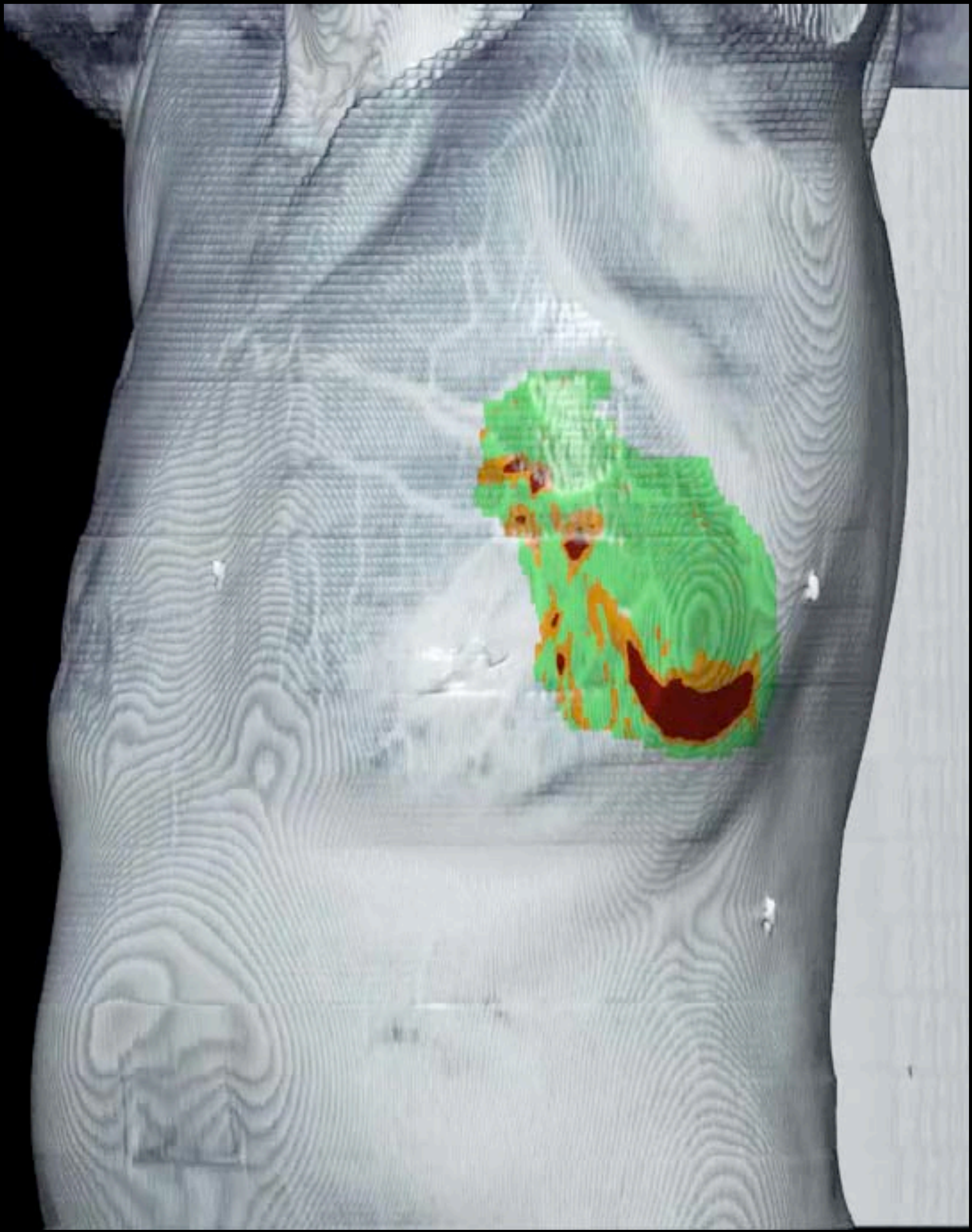
Radiological Pathlength

CW to proximal ITV Shader

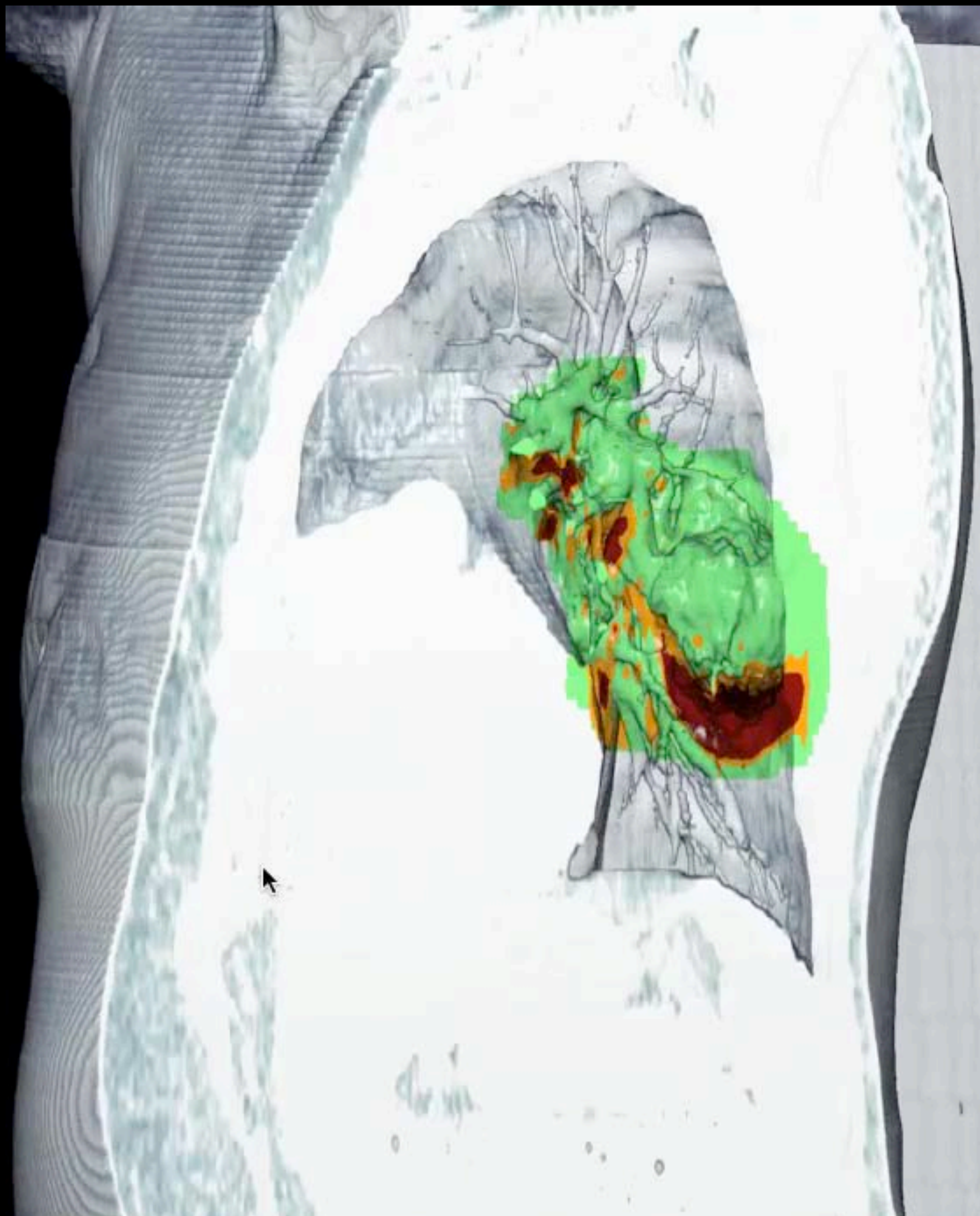
(includes intersected heart; uses ITV contours)



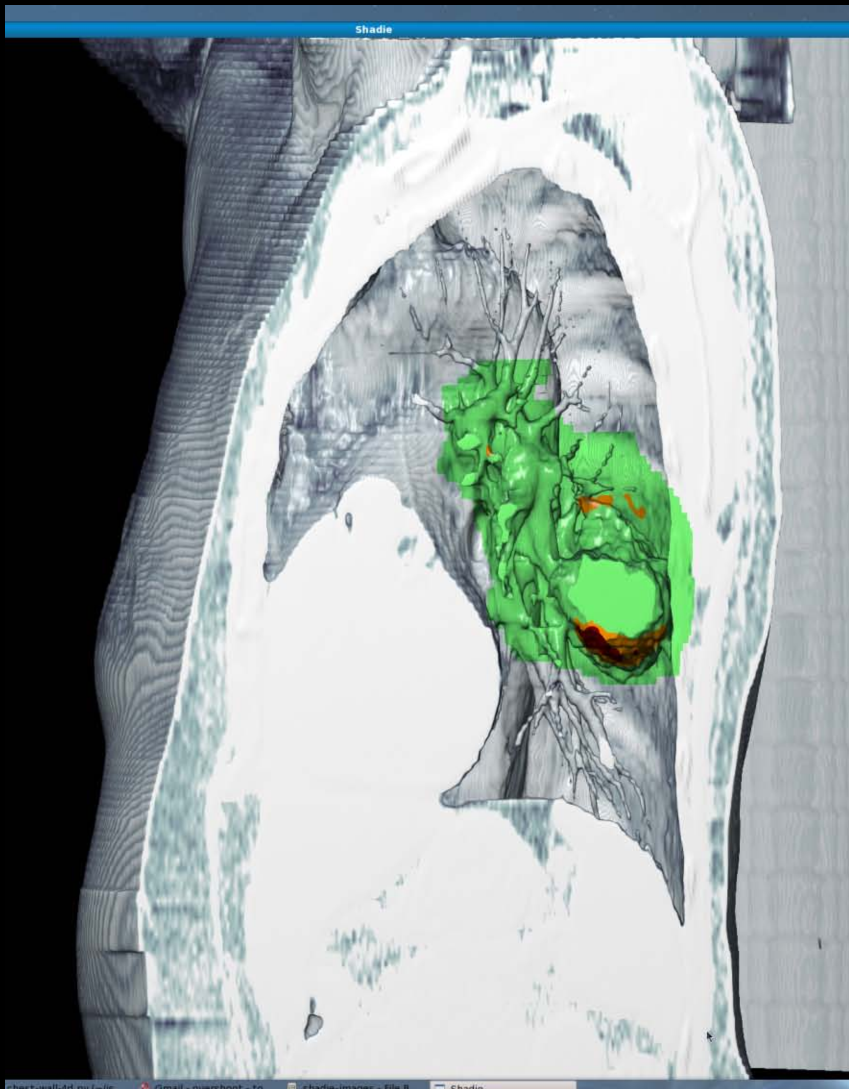


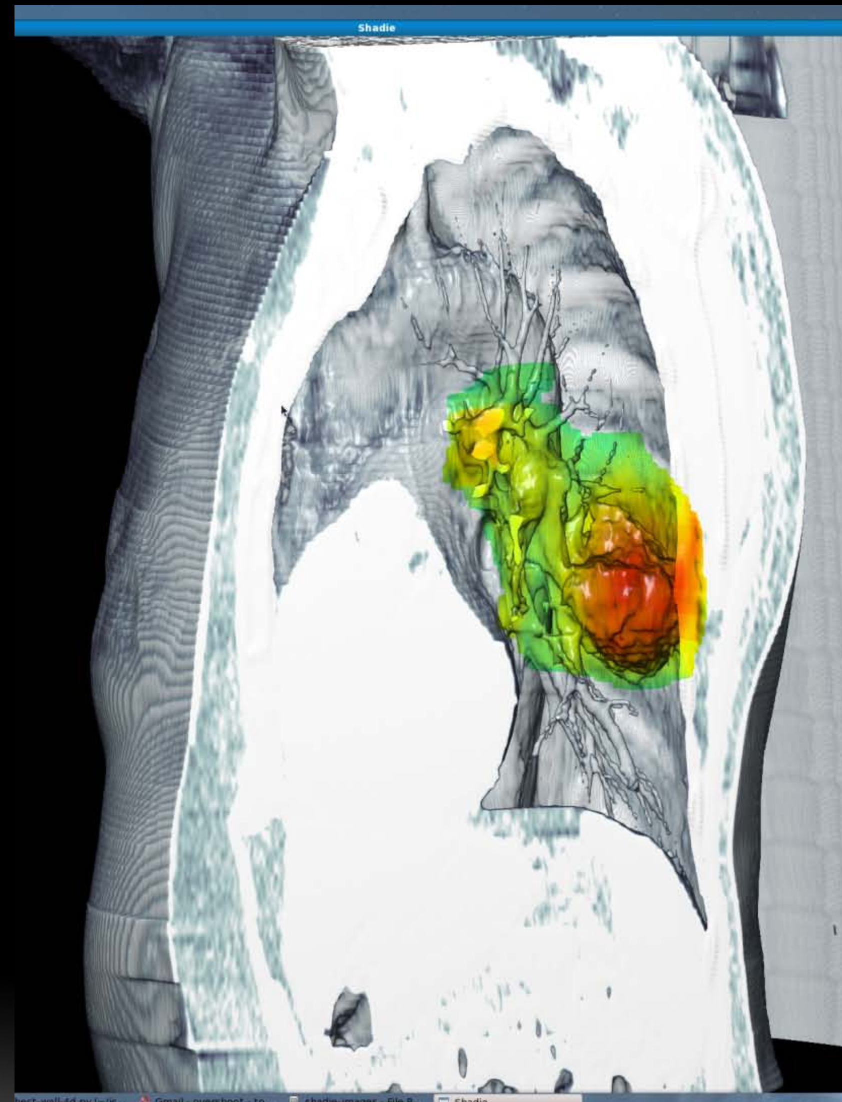


Green: <2mm; Yellow <5mm; Red >5mm



Overshoot





Time averaged

σ

Status & Future

- First implementation finished and online
  - http://code.google.com/p/shadie/
- Several shaders implemented at MGH
- Much more to on CS and medical side
  - Back ends, runtime system, computational kernels, uncertainty, more applications, etc. etc. etc.

Thank You

- The project was supported by the Federal Share of program income earned by Massachusetts General Hospital on C06 CA059267, Proton Therapy Research and Treatment Center