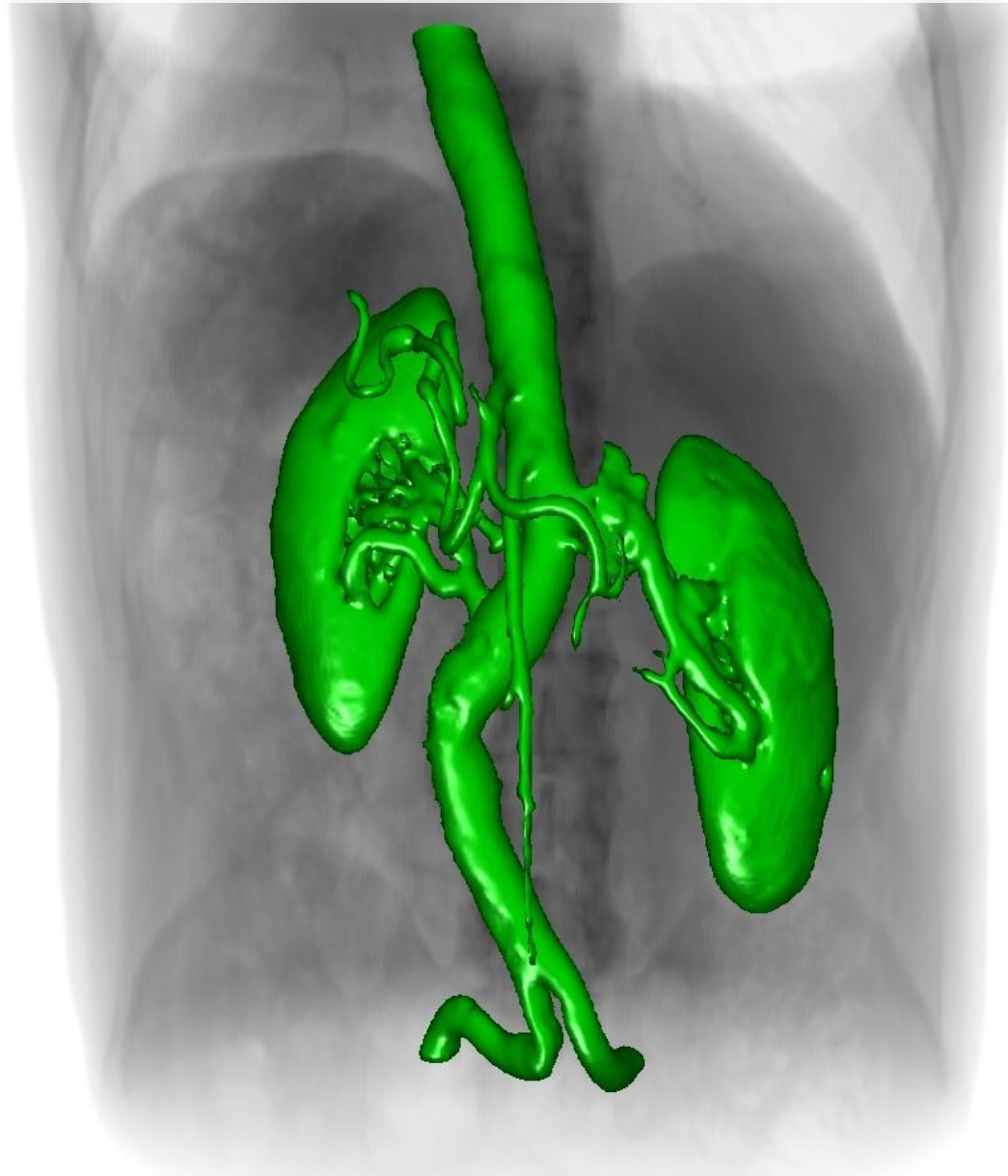


# Fast Volume Segmentation with Sparse Level Sets

Mike Roberts



**HARVARD**  
School of Engineering  
and Applied Sciences

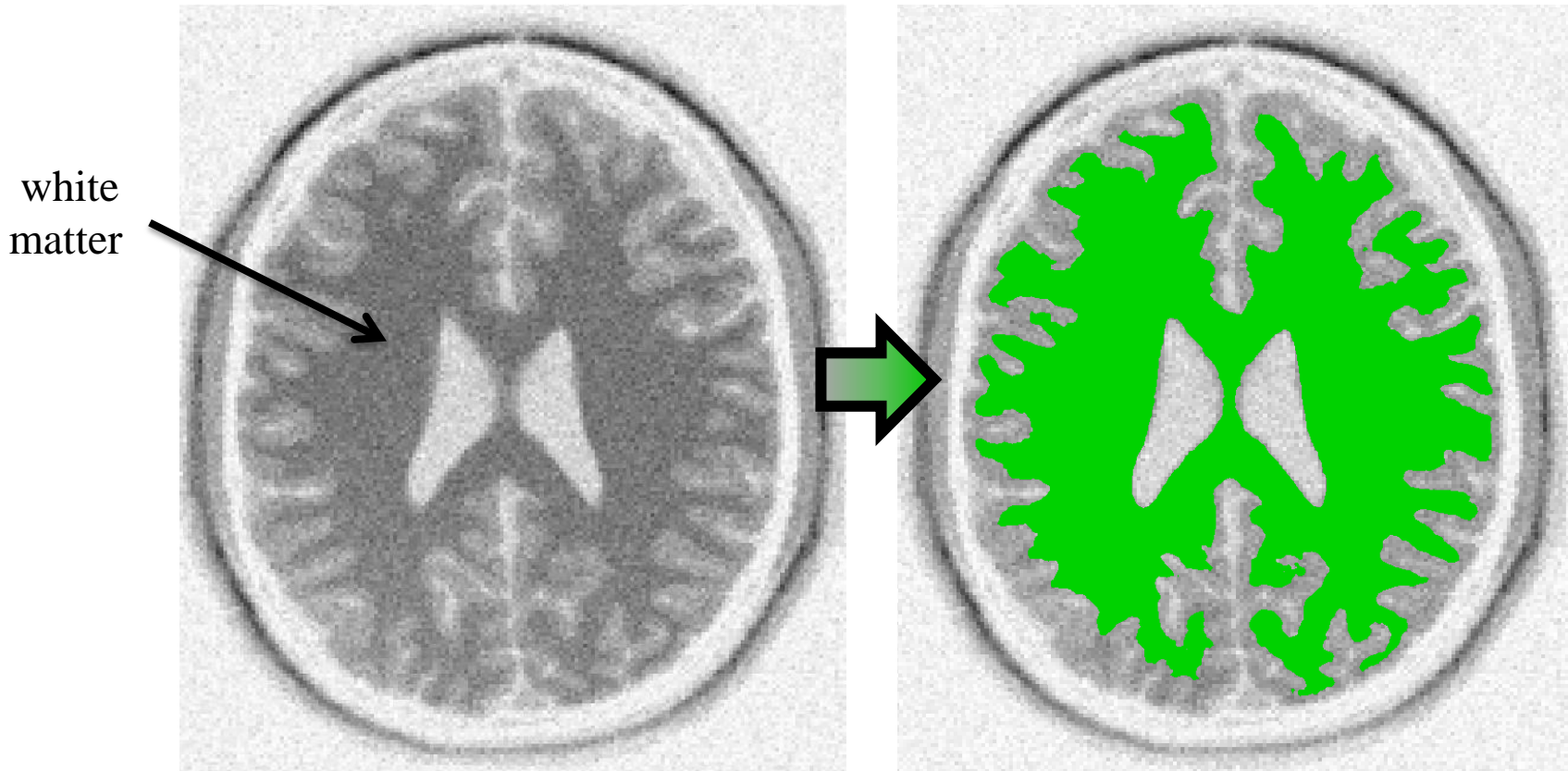
What do I mean by segmentation?

# What do I mean by segmentation?

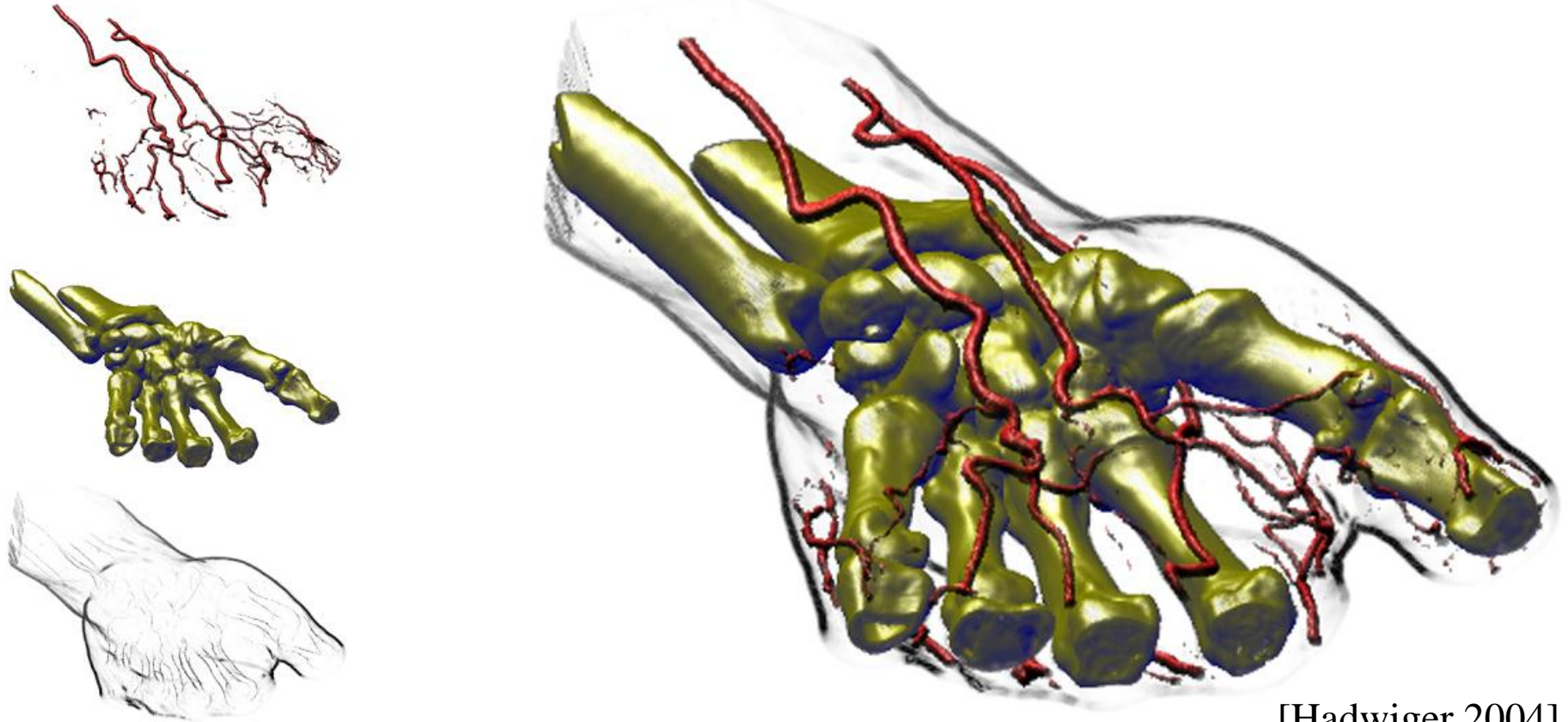
white  
matter



# What do I mean by segmentation?

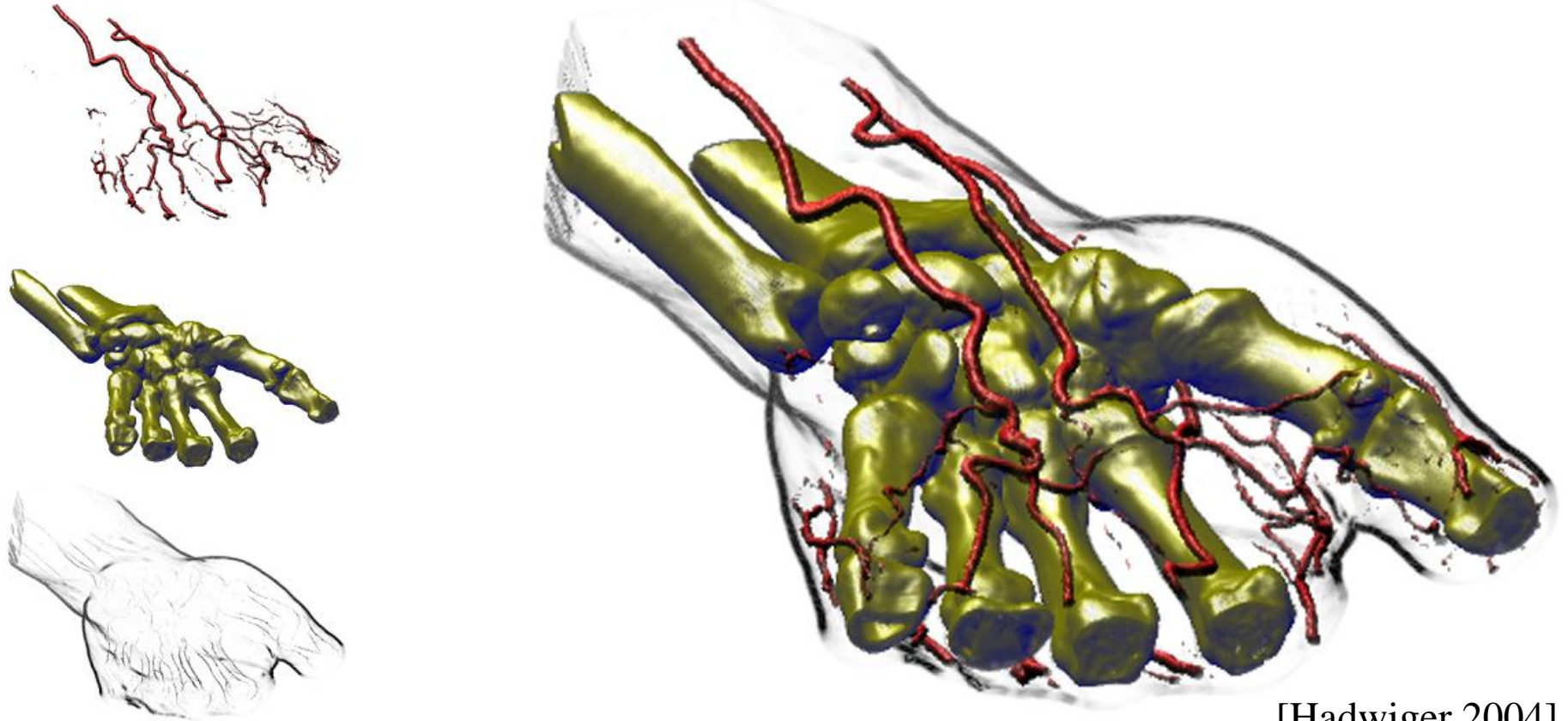


# Synergy Between Segmentation and Visualization



[Hadwiger 2004]

# Synergy Between Segmentation and Visualization



[Hadwiger 2004]

**Goals:** fast, interactive, accurate



# Why Level Sets?

**Good:**      interactive [Lefohn et al. 2004]  
                 accurate [Cates et al. 2004]

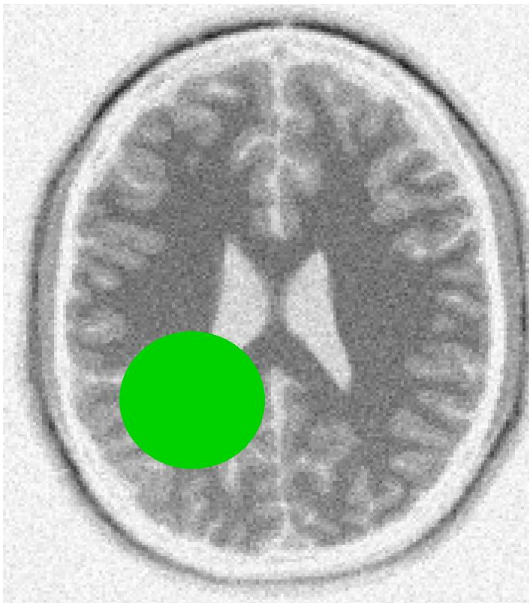
# Why Level Sets?

**Good:**      interactive [Lefohn et al. 2004]  
                 accurate [Cates et al. 2004]

**Bad:**        slow (even on the GPU)

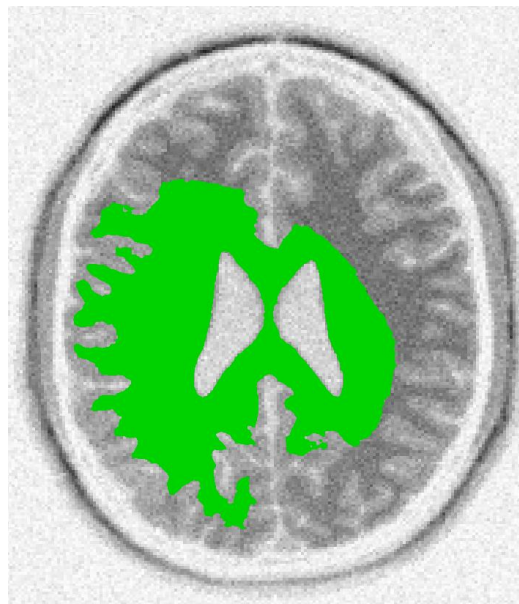
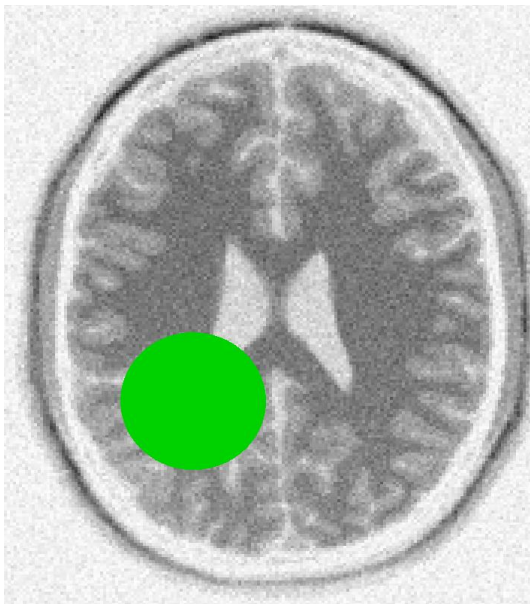


# Segmentation with Level Sets



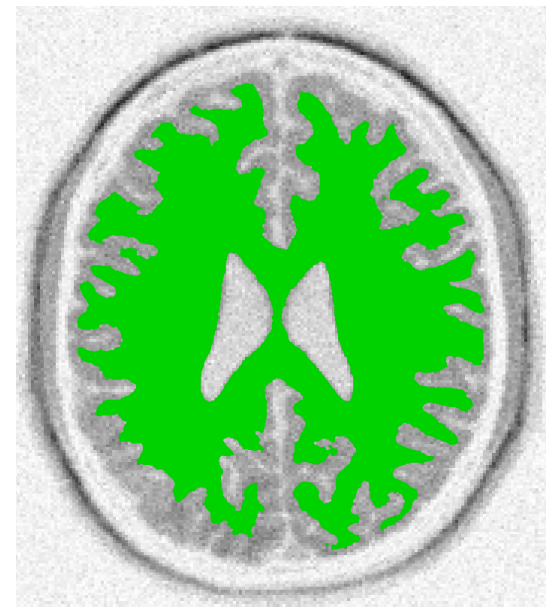
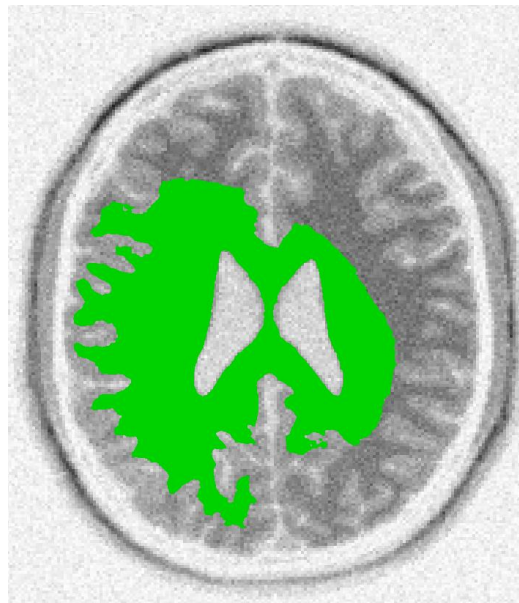
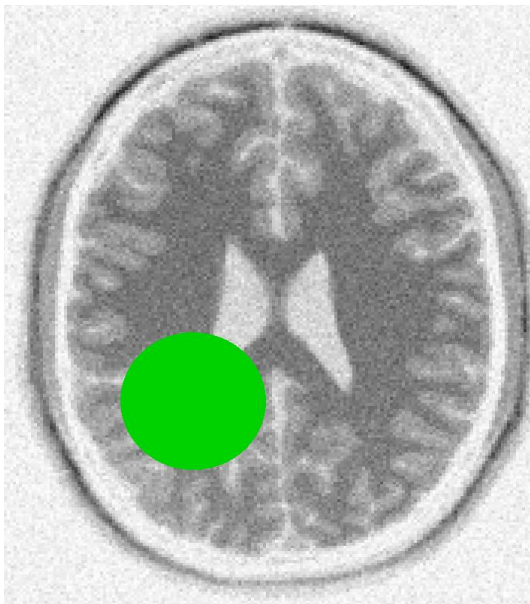
—————→ iterations

# Segmentation with Level Sets



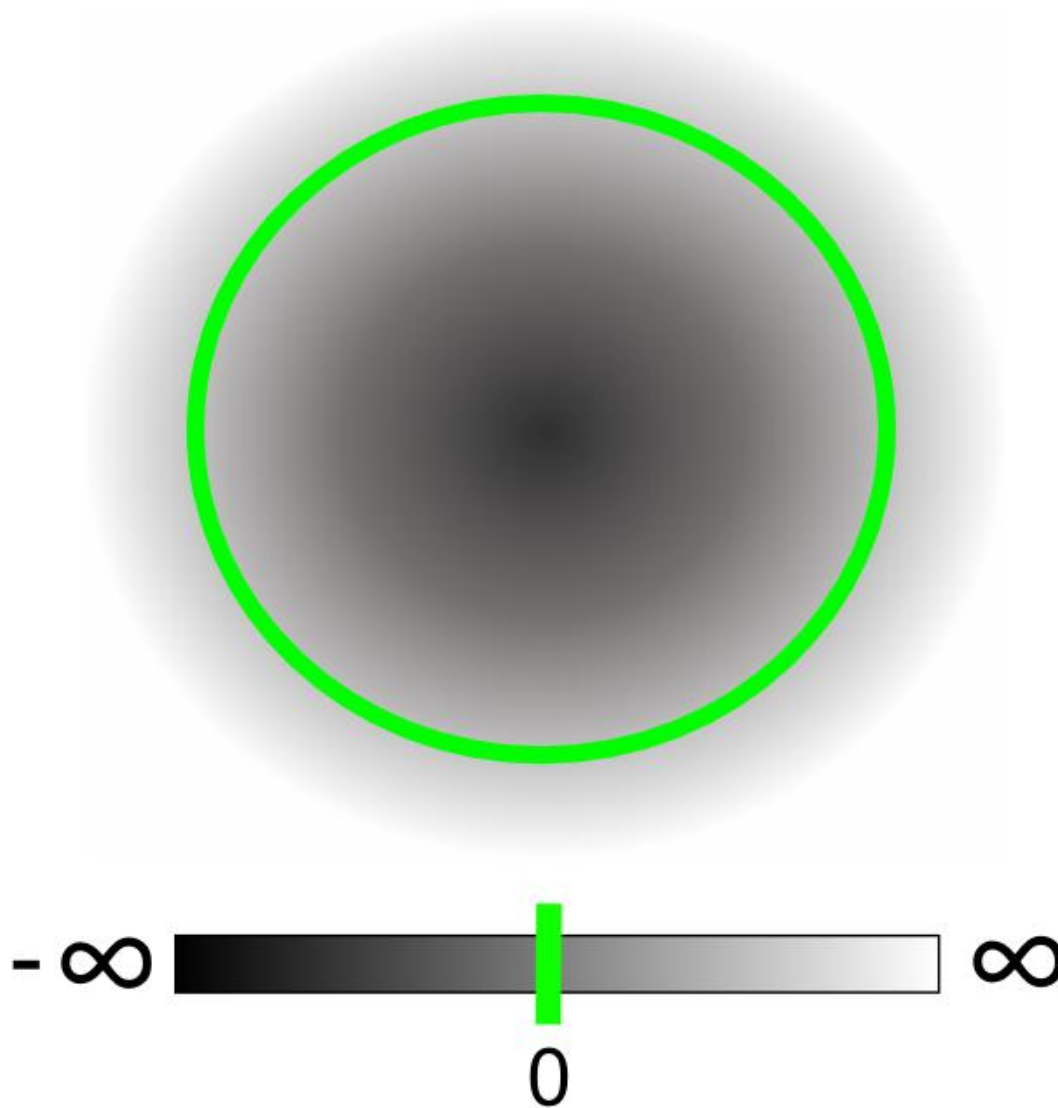
iterations

# Segmentation with Level Sets



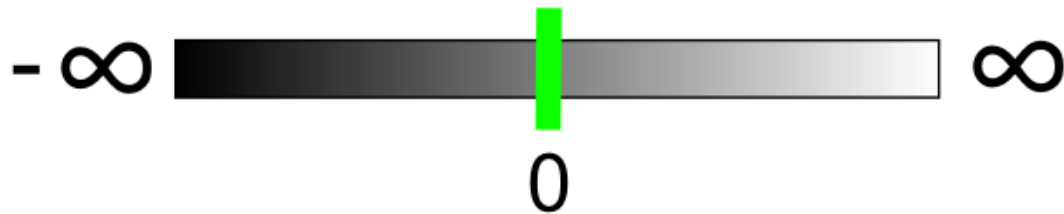
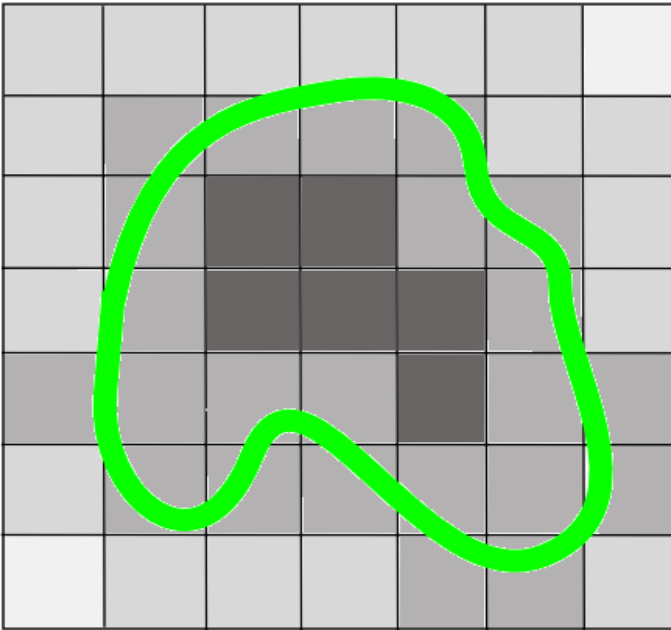
iterations

# Segmentation with Level Sets



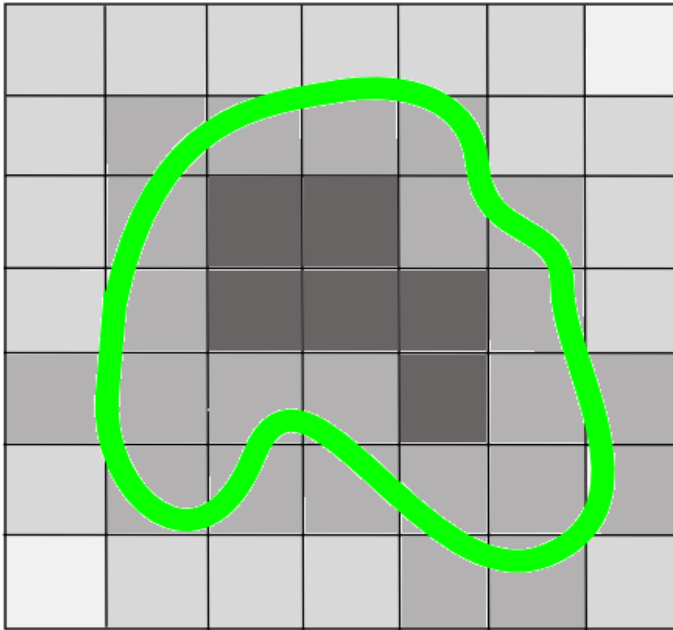
# Segmentation with Level Sets

Before:

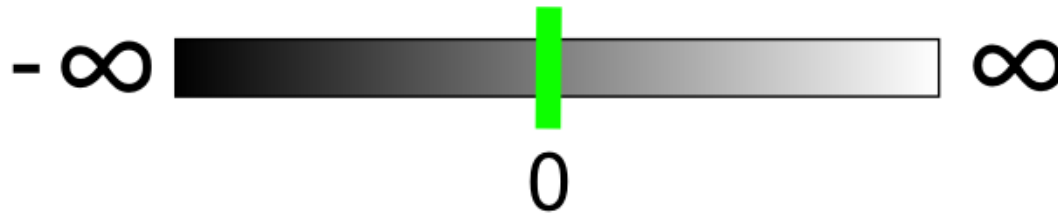
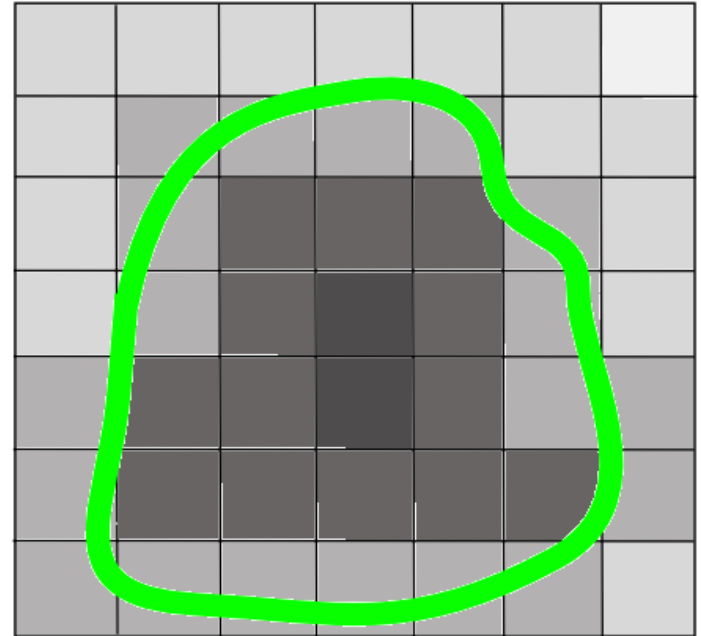


# Segmentation with Level Sets

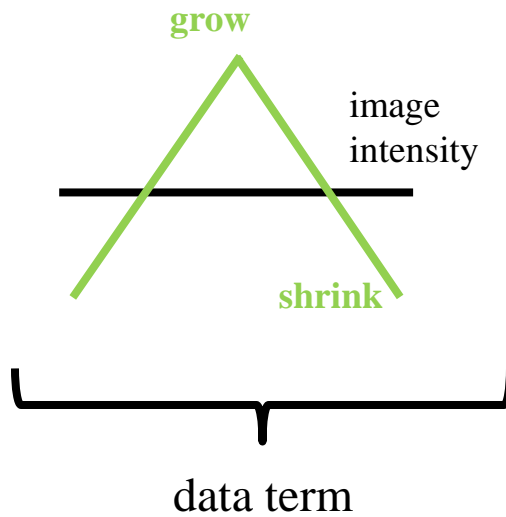
Before:



After:

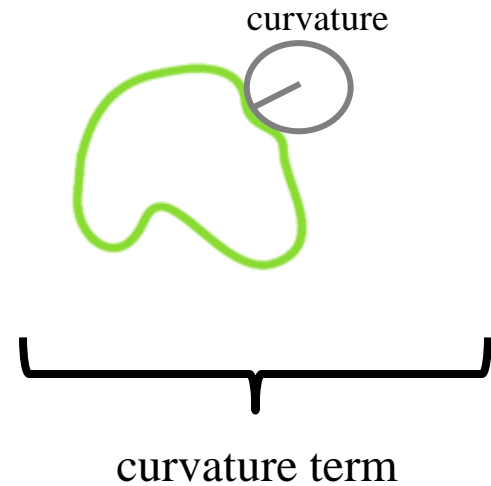
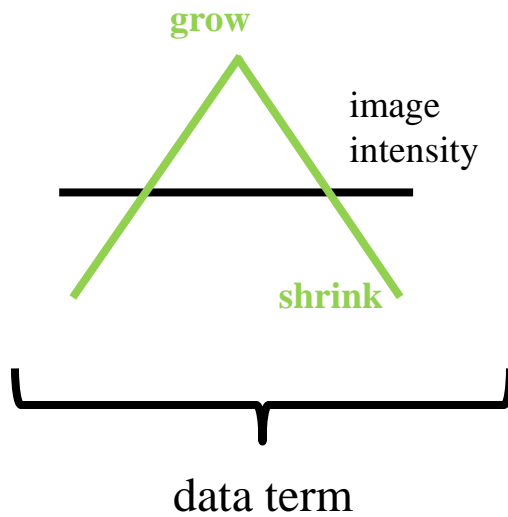


# Speed Function [Lefohn et al. 2003]





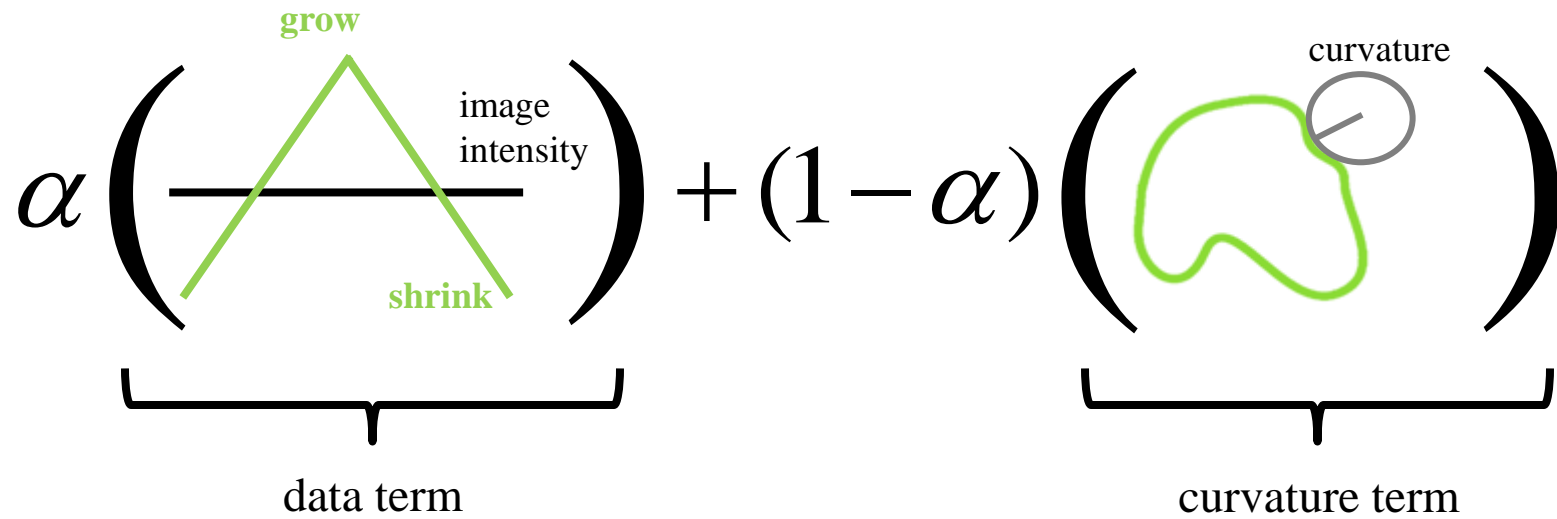
# Speed Function [Lefohn et al. 2003]



# Speed Function [Lefohn et al. 2003]

$$\alpha \left( \begin{array}{c} \text{grow} \\ \text{image} \\ \text{intensity} \\ \text{shrink} \end{array} \right) + (1 - \alpha) \left( \begin{array}{c} \text{curvature} \end{array} \right)$$

data term                      curvature term



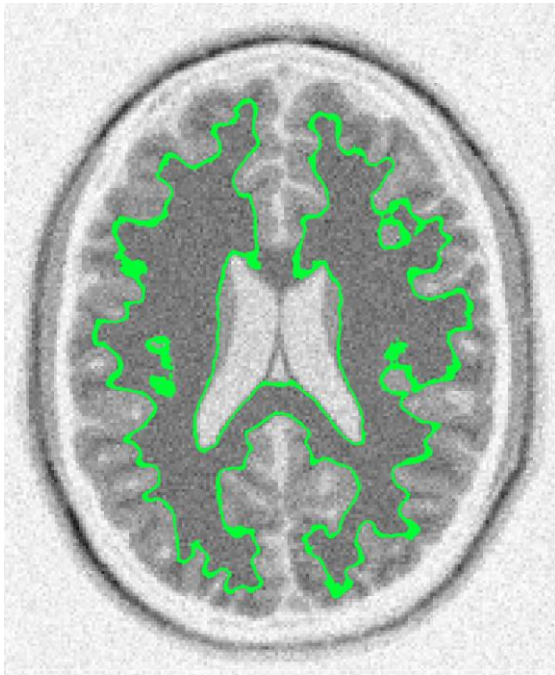
# Speed Function [Lefohn et al. 2003]

$$\alpha \left( \underbrace{\begin{array}{c} \text{grow} \\ \text{image} \\ \text{intensity} \\ \text{shrink} \end{array}}_{\text{data term}} \right) + (1 - \alpha) \left( \underbrace{\begin{array}{c} \text{curvature} \end{array}}_{\text{curvature term}} \right)$$

The diagram illustrates the Speed Function, which is a weighted sum of two terms. The first term, the data term, is represented by a green triangle with a horizontal line through its center. The top vertex is labeled "grow" and the bottom vertex is labeled "shrink". The horizontal line is labeled "image intensity". The second term, the curvature term, is represented by a green closed curve with a small circle at one of its points, labeled "curvature". The parameter  $\alpha$  is shown as a coefficient for the data term, and  $(1 - \alpha)$  is shown as a coefficient for the curvature term.

$\alpha \approx \text{smoothness}$

# Speed Function [Lefohn et al. 2003]



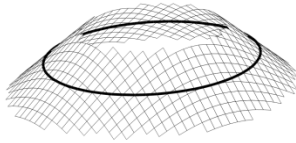
$$\alpha = 0.4$$



$$\alpha = 0.0$$

# Previous Work

CPU:



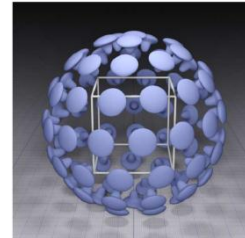
Narrow Band  
[Adalsteinson and Sethian 1995]



Sparse Field  
[Whitaker 1998]  
[Peng et al. 1999]



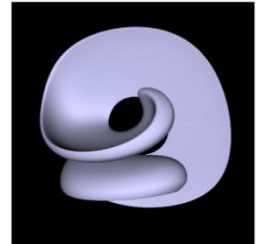
Sparse Block Grid  
[Bridson et al. 2003]



Dynamic Tubular Grid  
[Nielson and Museth 2006]

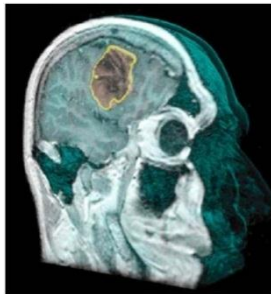


Hierarchical Run-Length Encoded  
[Houston et al. 2006]



Sorted Tile List  
[van der Laan et al. 2010]

GPU:



GPU Narrow Band  
[Lefohn et al. 2003]  
[Lefohn et al. 2004]  
[Jeong et al. 2009]

# Previous Work

Common theme: leverage **spatial coherence** to reduce computational workload

# Our Approach



# Our Approach

Also leverage **temporal coherence** to reduce computational workload

# Necessary Conditions for Voxel to Change Value

Spatial Coherence:

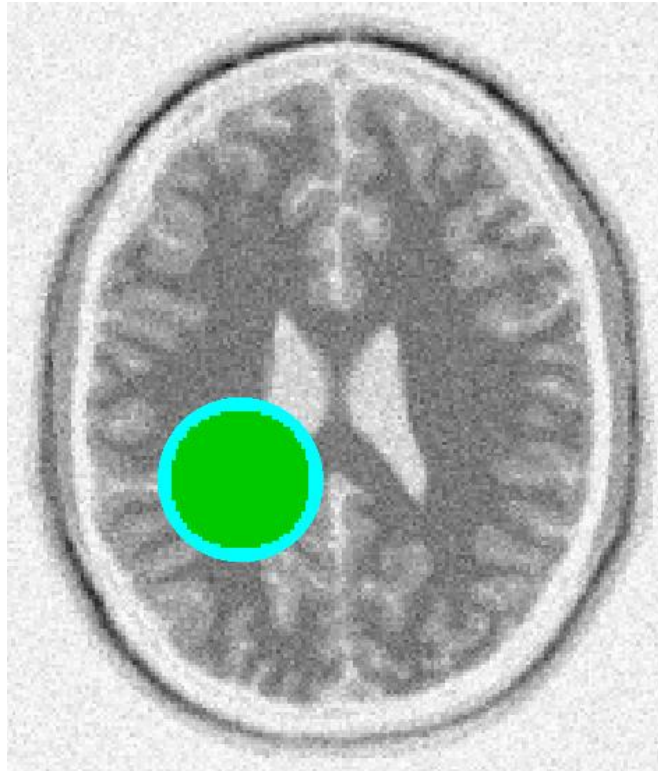
close to the level set surface

Temporal Coherence:

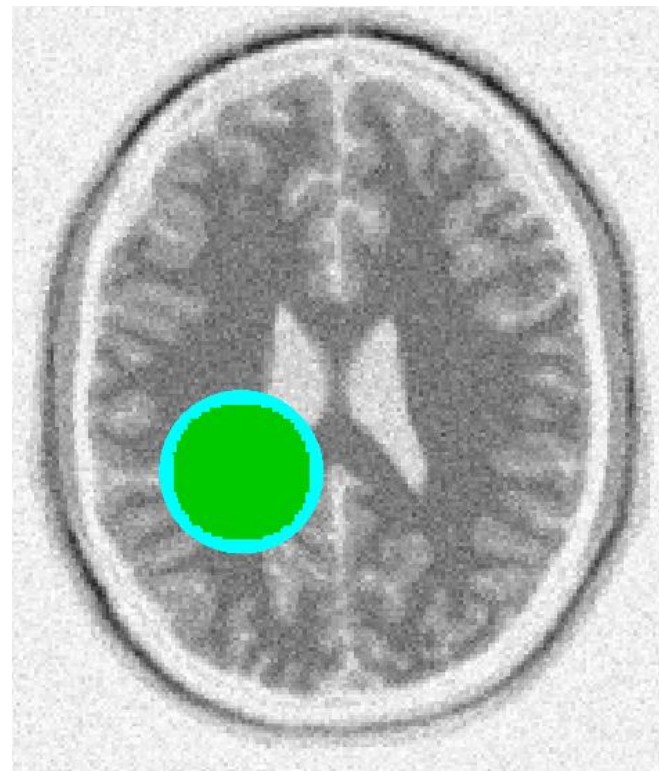
local neighborhood changing over time

[Roberts et al. 2010]

# Leveraging Temporal Coherence



spatial coherence only



spatial and temporal coherence

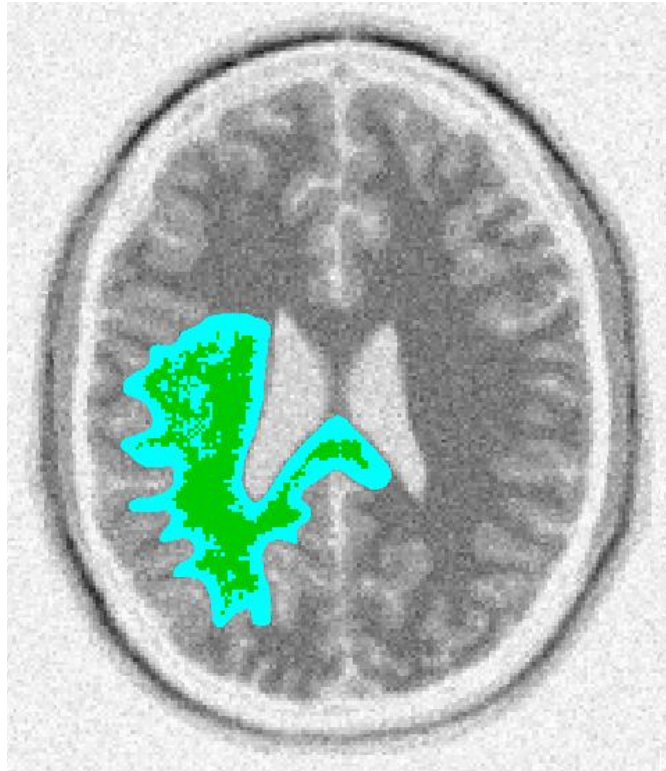


active computation

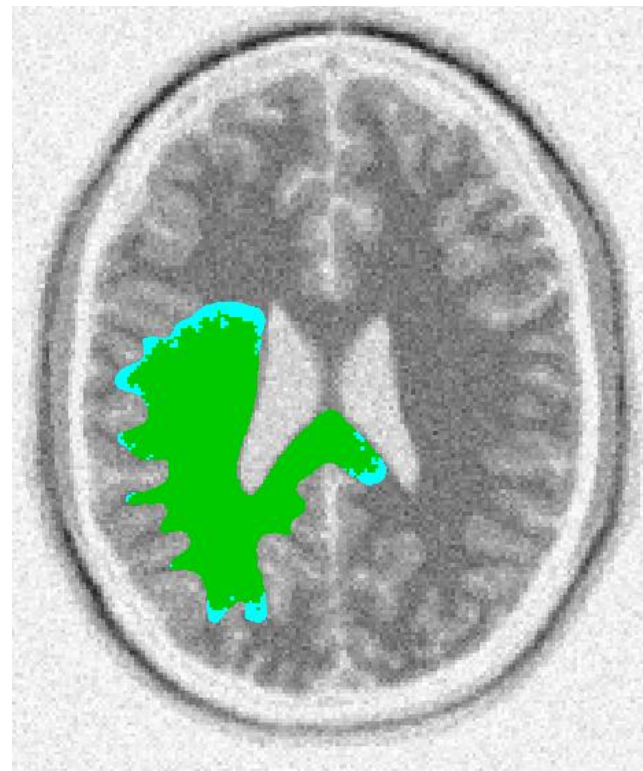


segmented region

# Leveraging Temporal Coherence



spatial coherence only



spatial and temporal coherence

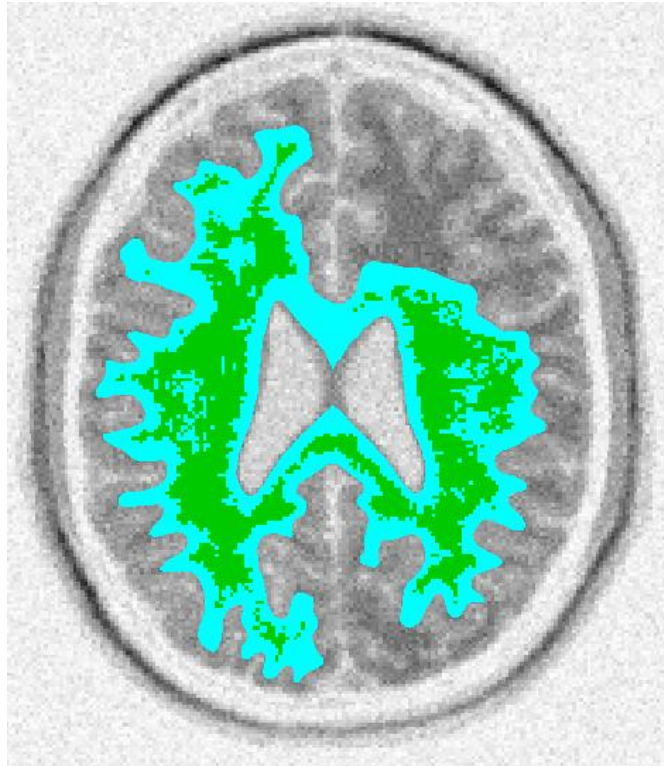


active computation

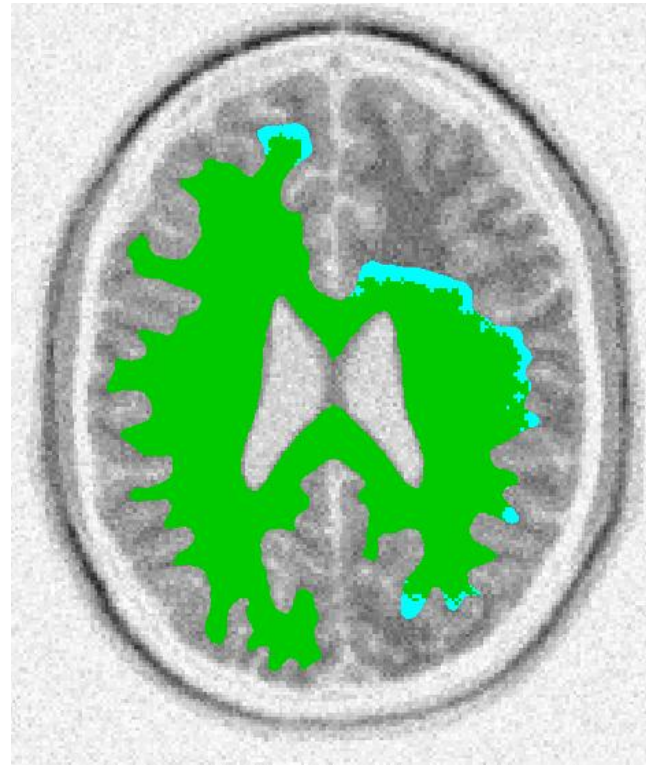


segmented region

# Leveraging Temporal Coherence



spatial coherence only



spatial and temporal coherence



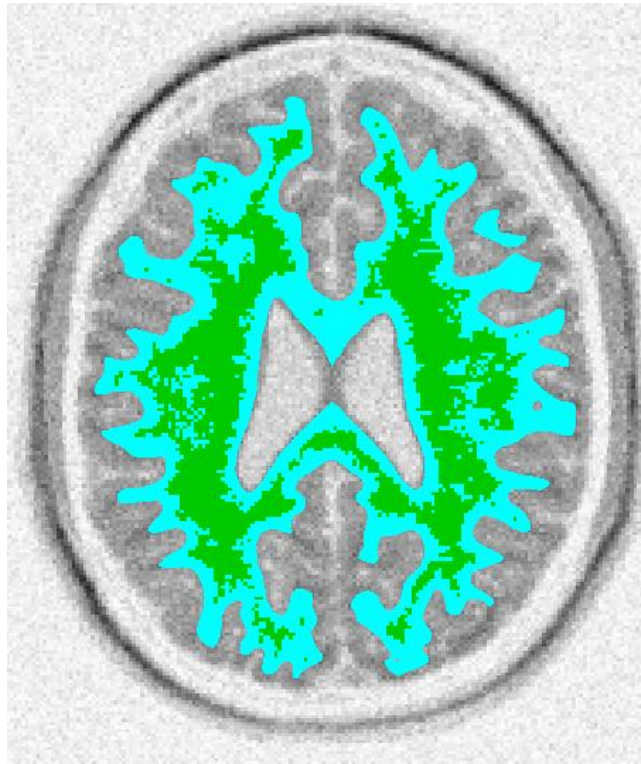
active computation



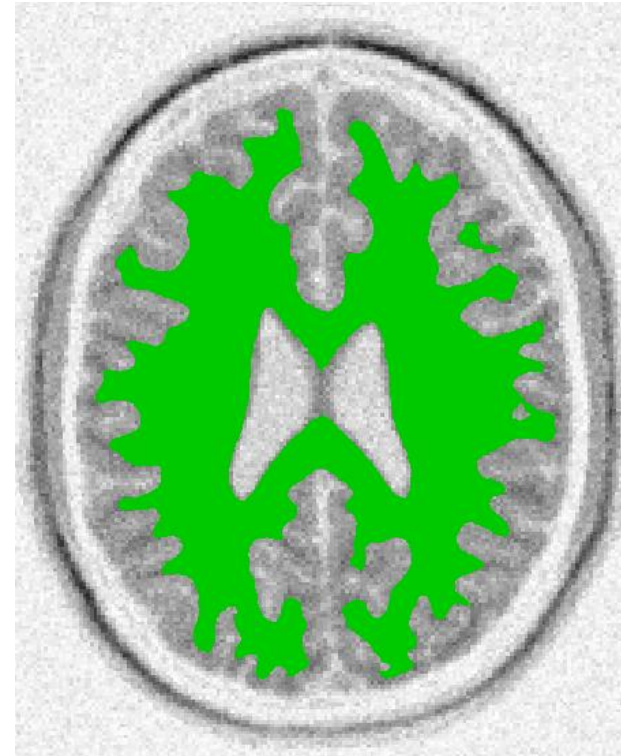
segmented region



# Leveraging Temporal Coherence



spatial coherence only



spatial and temporal coherence



active computation



segmented region

# Demo



# Temporally Coherent Algorithm

# Temporally Coherent Algorithm

Initialize level set field and active  
computational domain

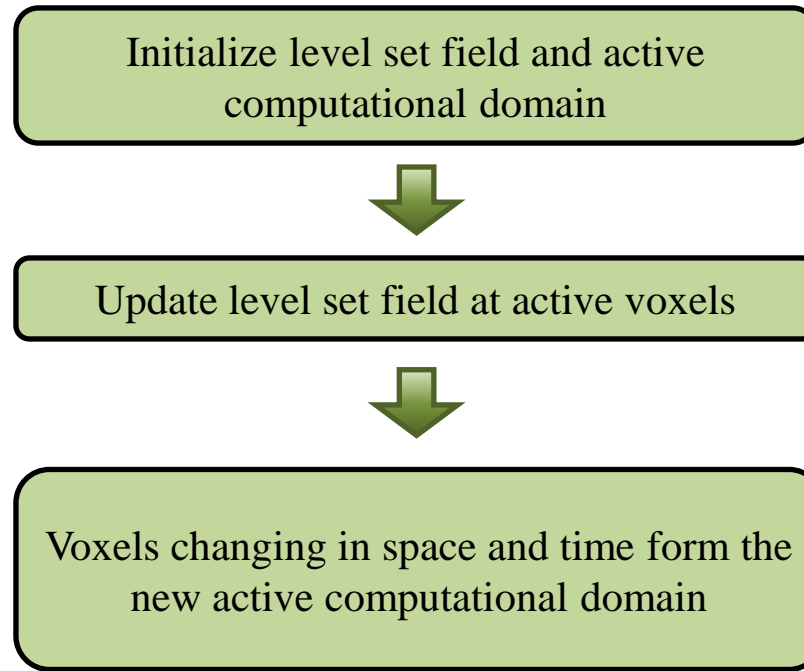
# Temporally Coherent Algorithm

Initialize level set field and active  
computational domain

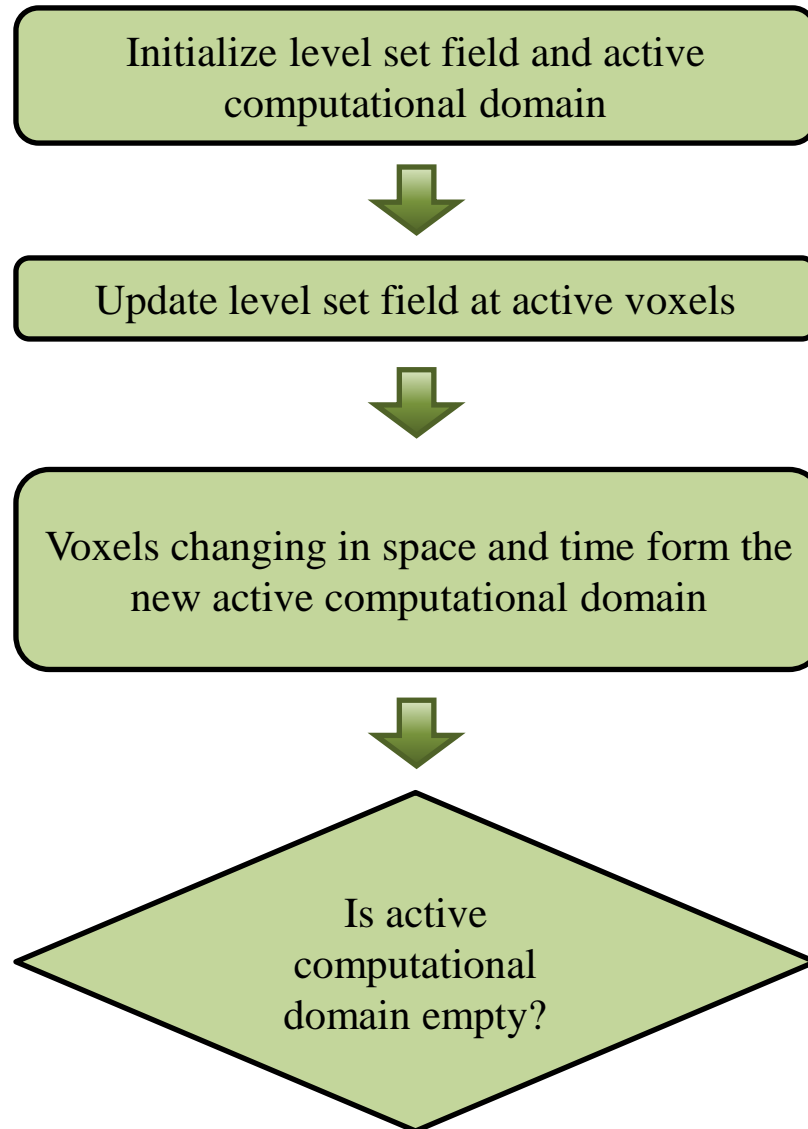


Update level set field at active voxels

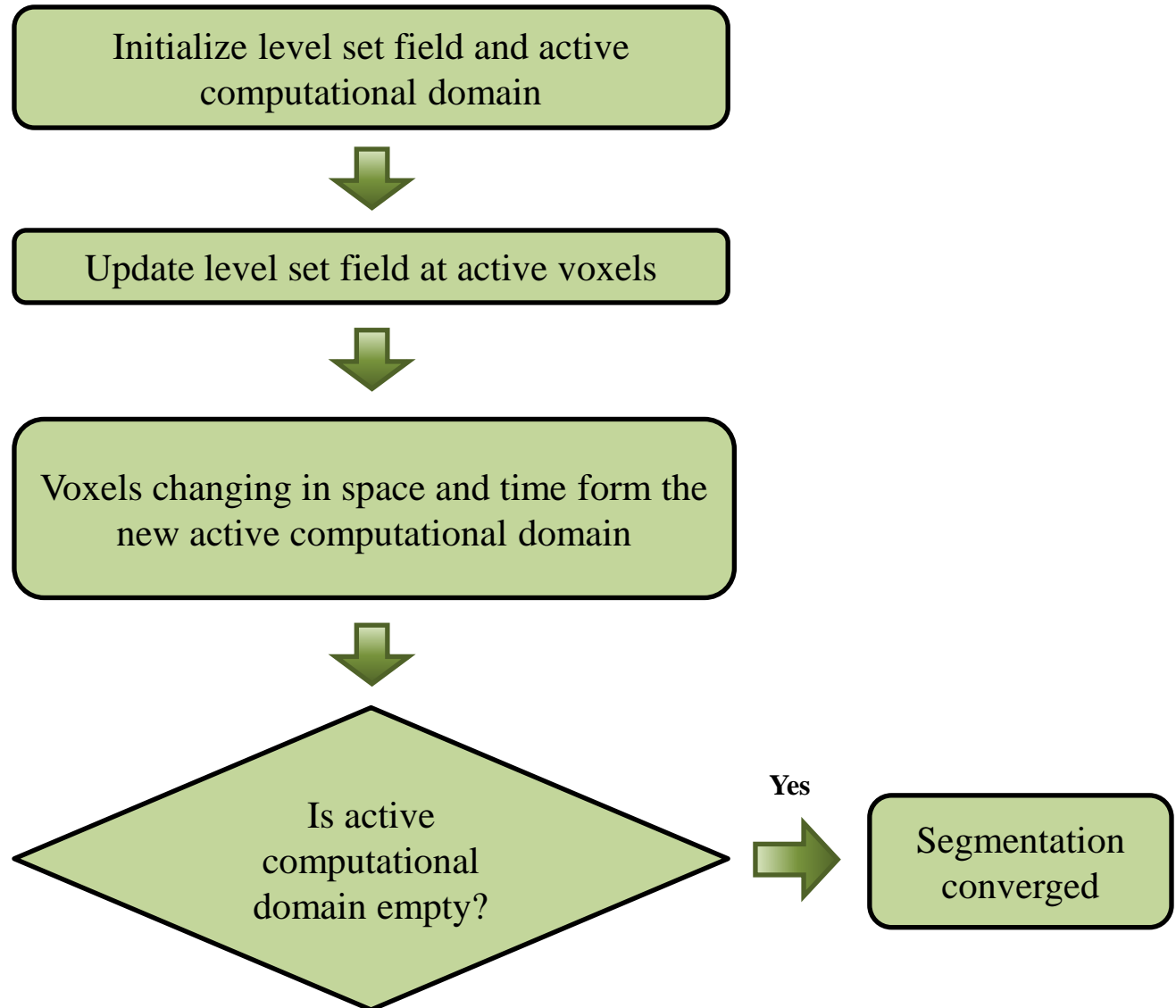
# Temporally Coherent Algorithm



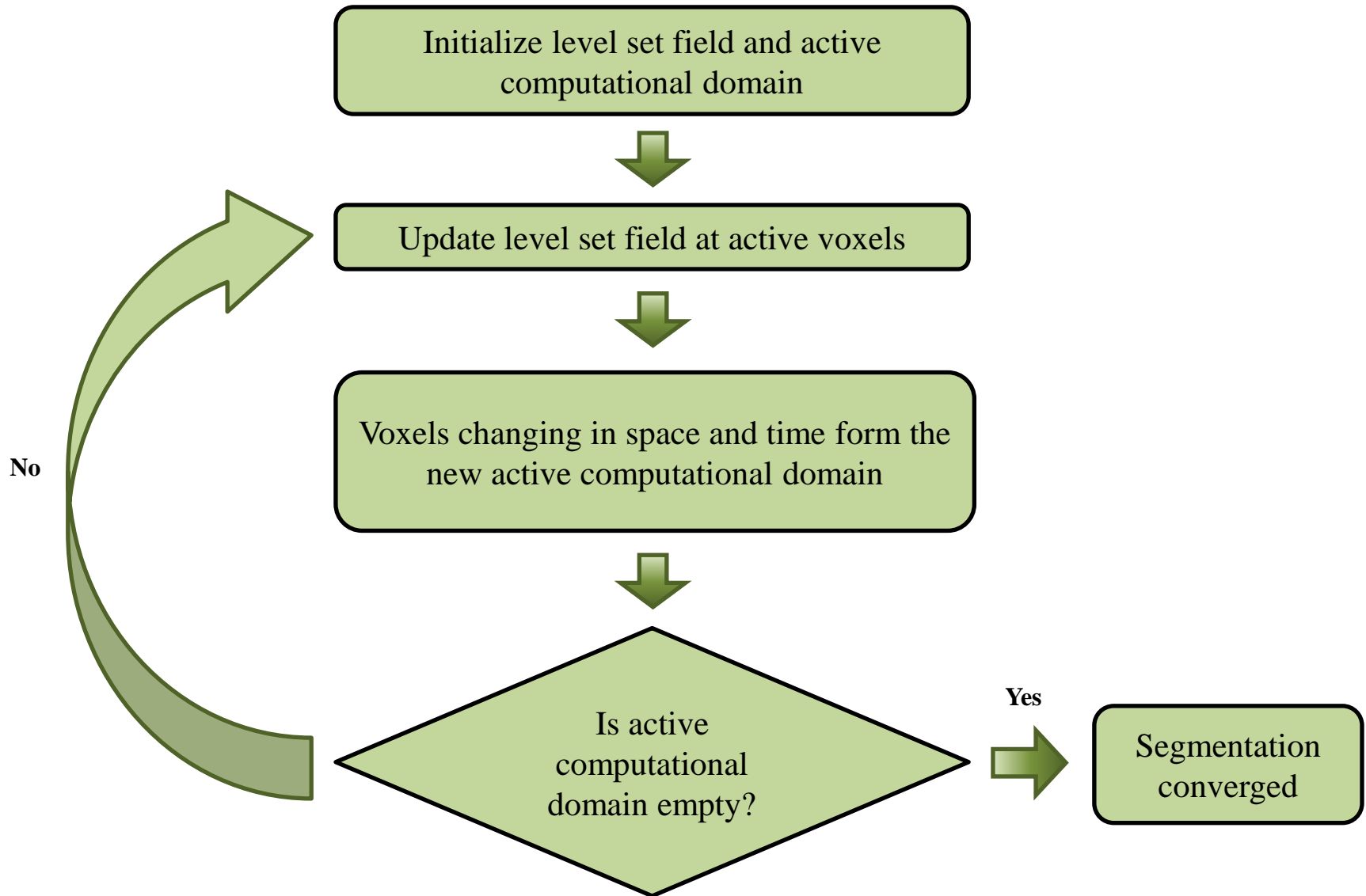
# Temporally Coherent Algorithm



# Temporally Coherent Algorithm

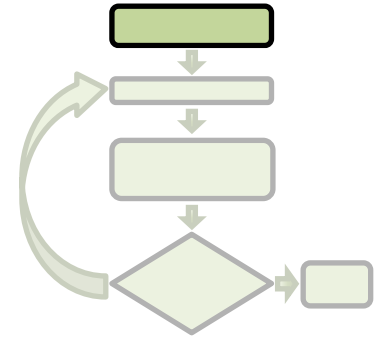


# Temporally Coherent Algorithm

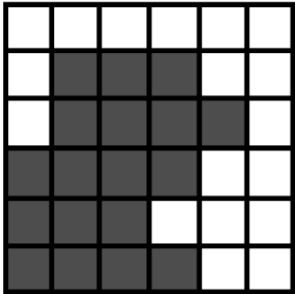




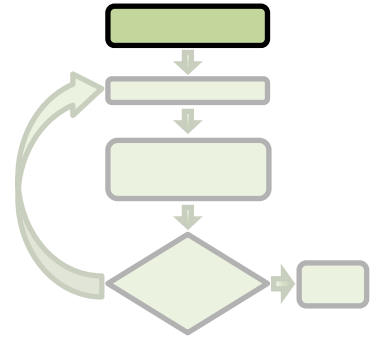
Initialize the level set field and active  
computational domain



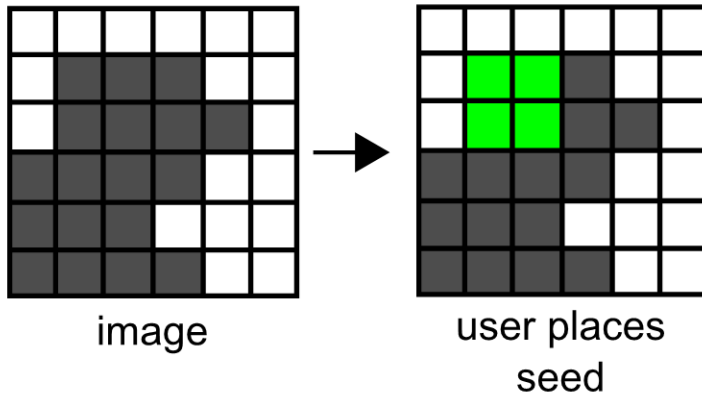
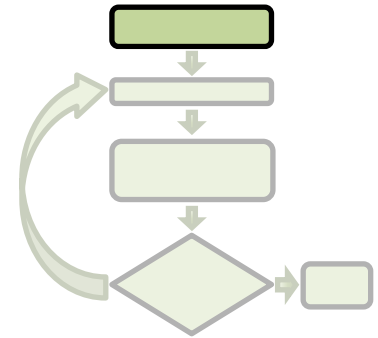
Initialize the level set field and active  
computational domain



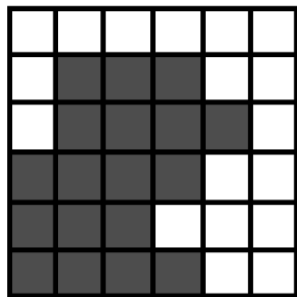
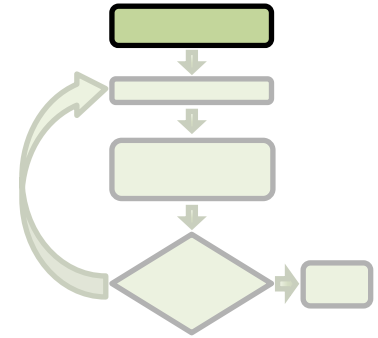
image



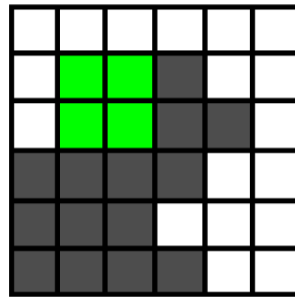
Initialize the level set field and active computational domain



Initialize the level set field and active computational domain



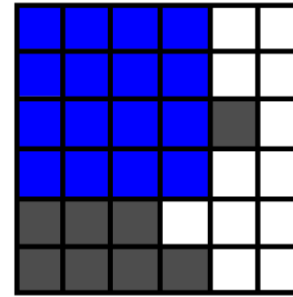
image



user places  
seed

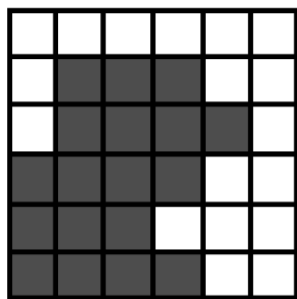
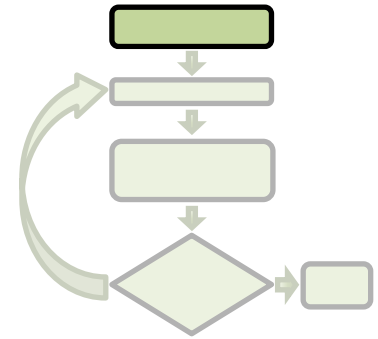


spatial  
gradient

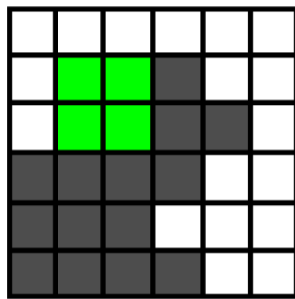


voxels changing  
in space

Initialize the level set field and active computational domain



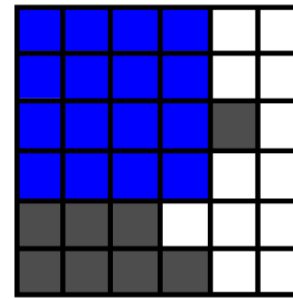
image



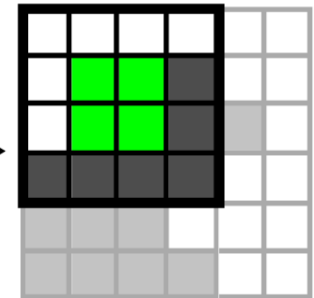
user places  
seed



spatial  
gradient

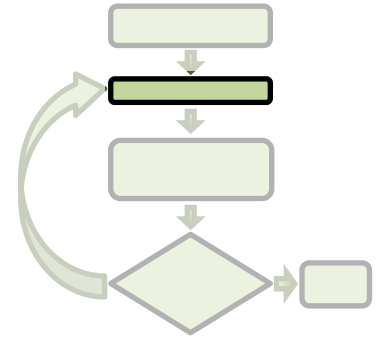


voxels changing  
in space

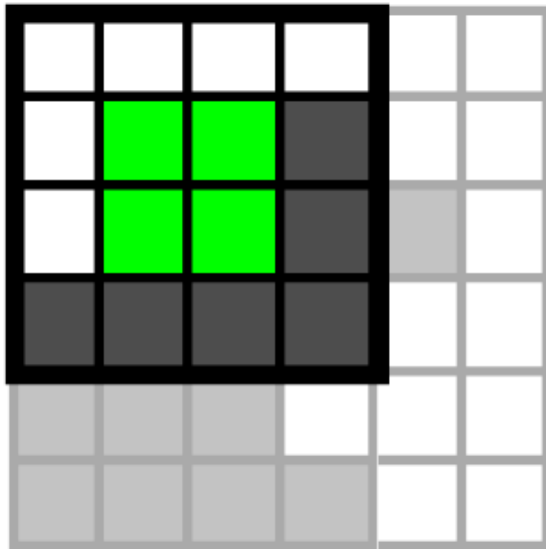
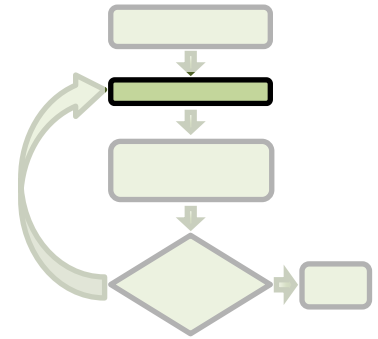


initial active set

Update the level set field at active voxels

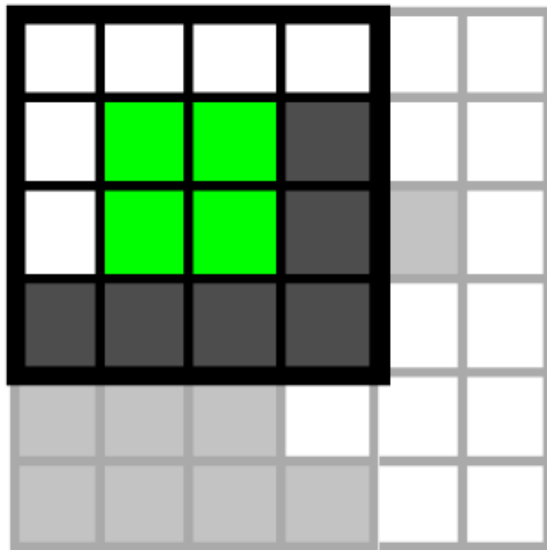
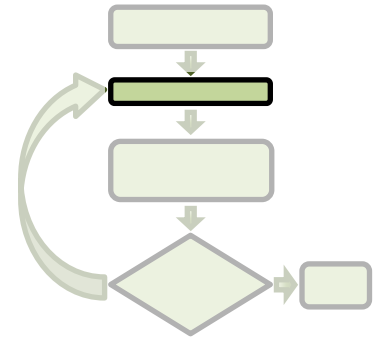


Update the level set field at active voxels

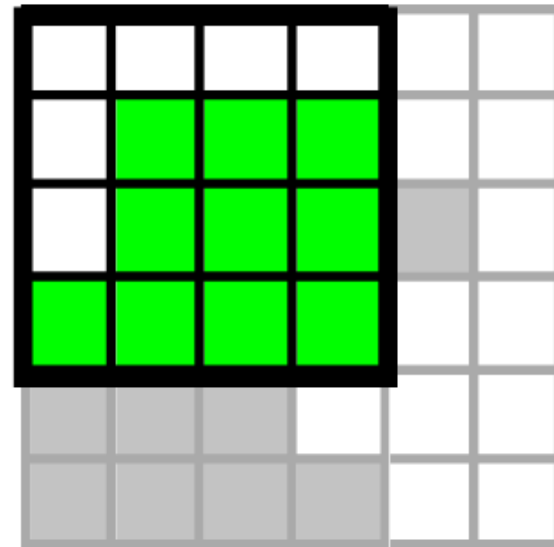


old level set field

Update the level set field at active voxels



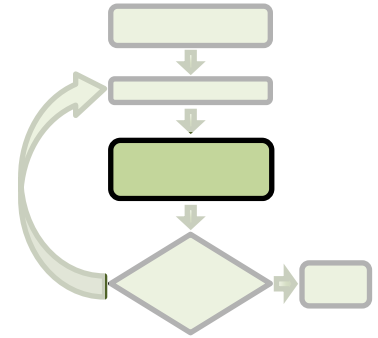
old level set field



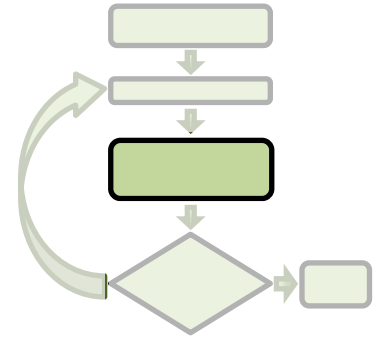
new level set field



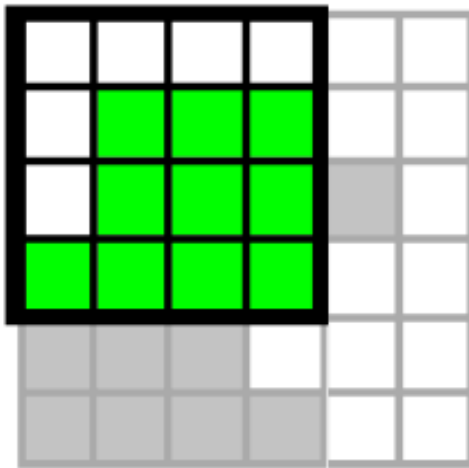
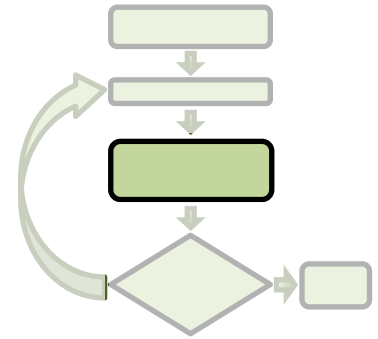
Voxels changing in space and time form the new active computational domain



Find the voxels that are changing in *space*

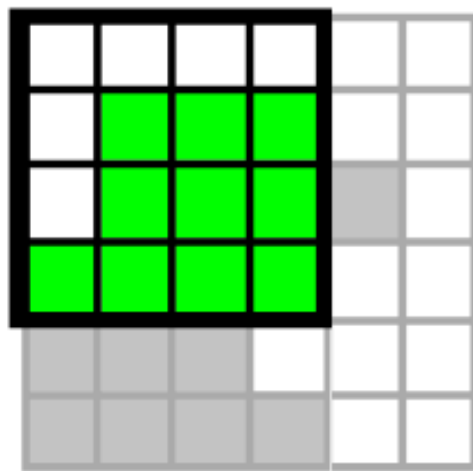
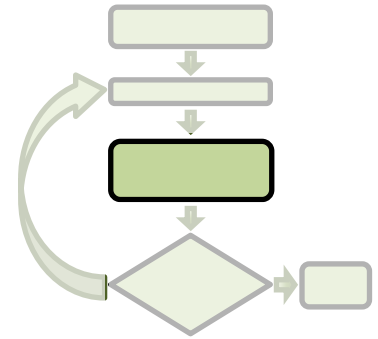


Find the voxels that are changing in *space*



current level  
set field

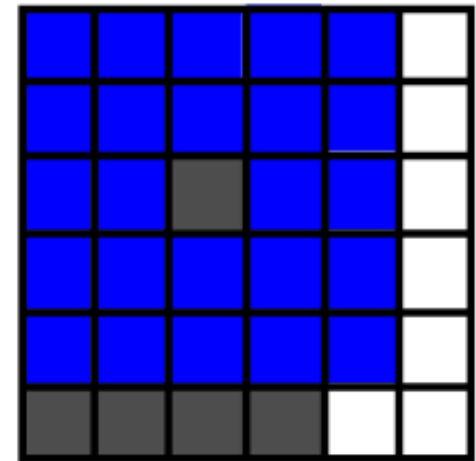
Find the voxels that are changing in *space*



current level  
set field

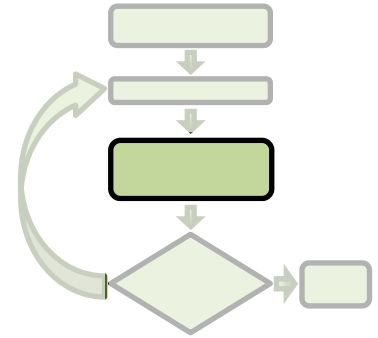


spatial  
gradient

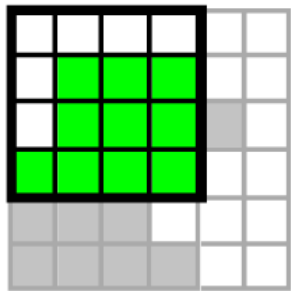


voxels changing  
in space

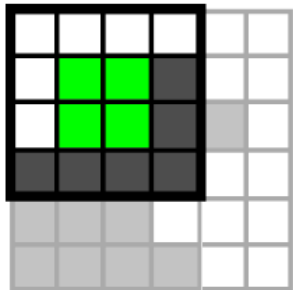
Find the voxels that are changing in *time*



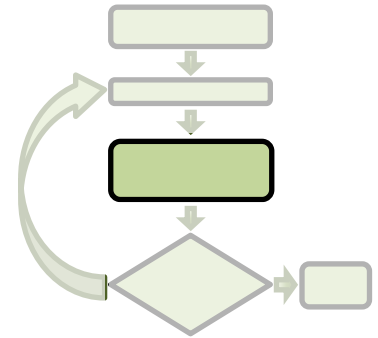
Find the voxels that are changing in *time*



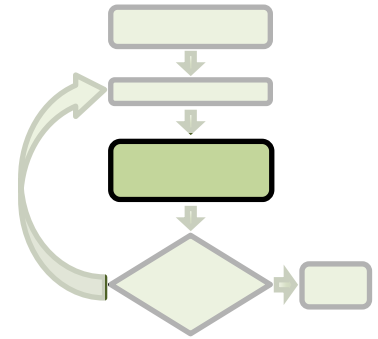
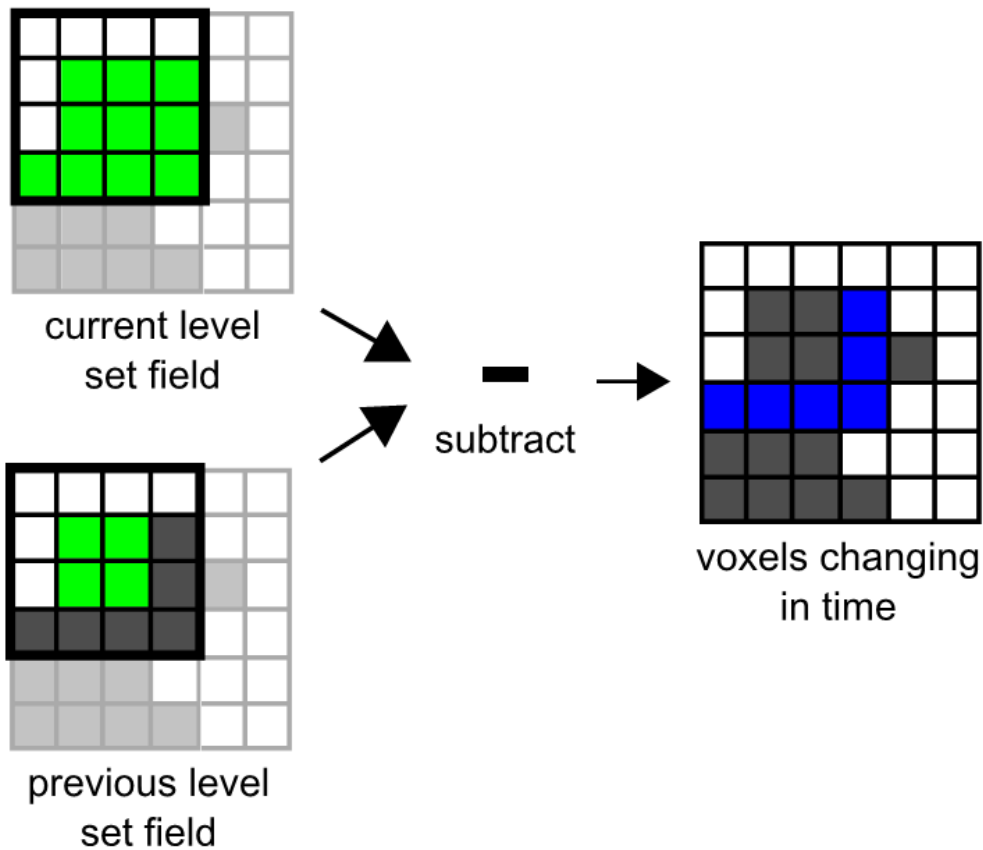
current level  
set field



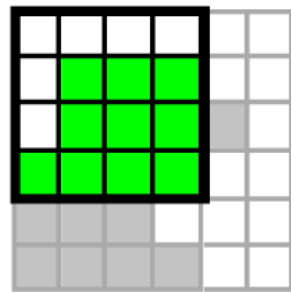
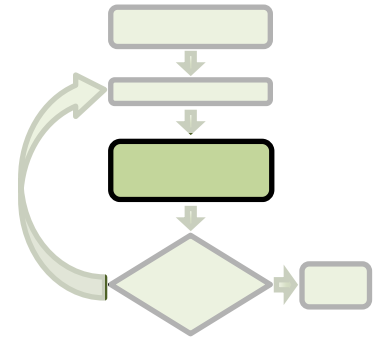
previous level  
set field



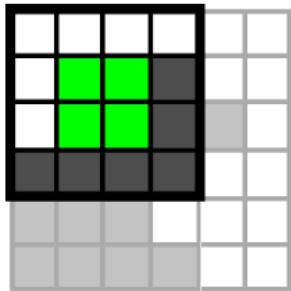
Find the voxels that are changing in *time*



Find the voxels that are changing in *time*



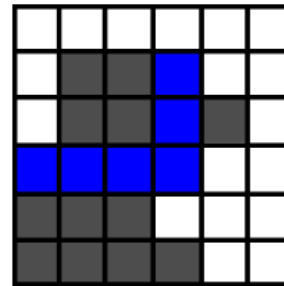
current level  
set field



previous level  
set field

subtract

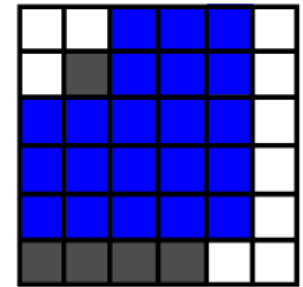
-



voxels changing  
in time



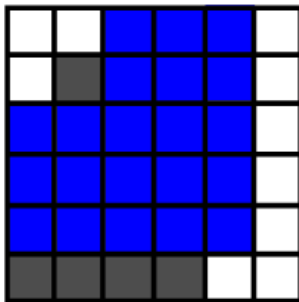
dilate



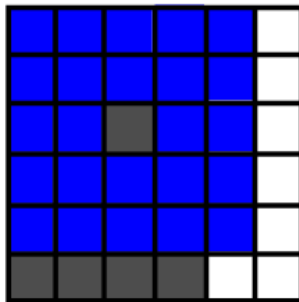
neighbors of voxels  
changing in time



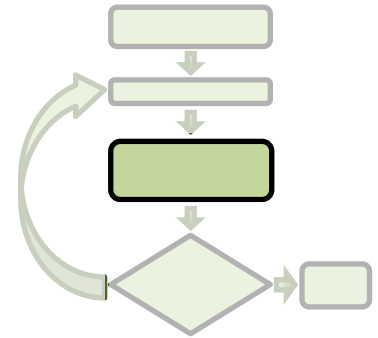
Find the voxels that are changing in  
*space* and *time*



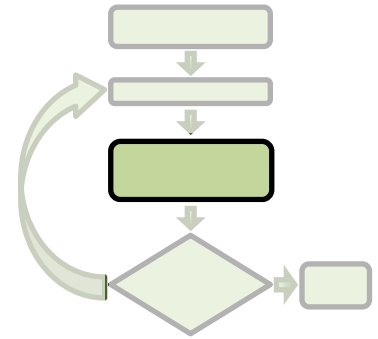
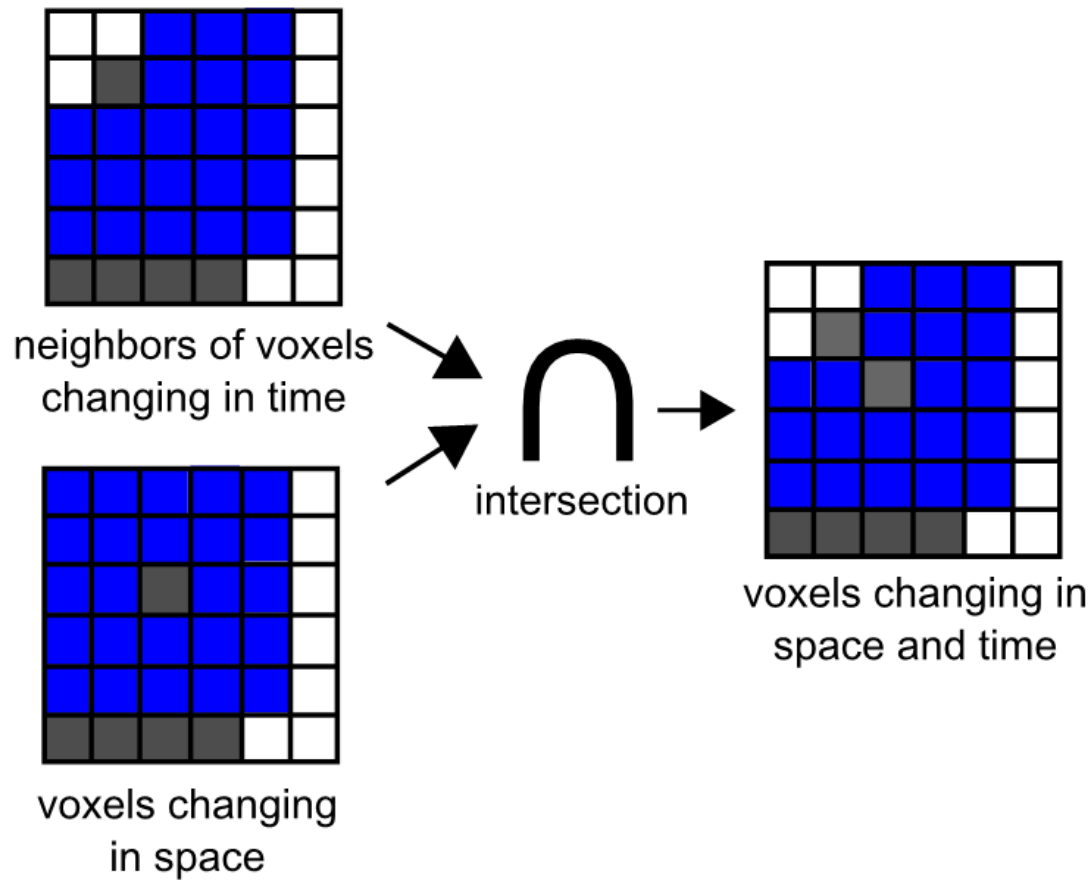
neighbors of voxels  
changing in time



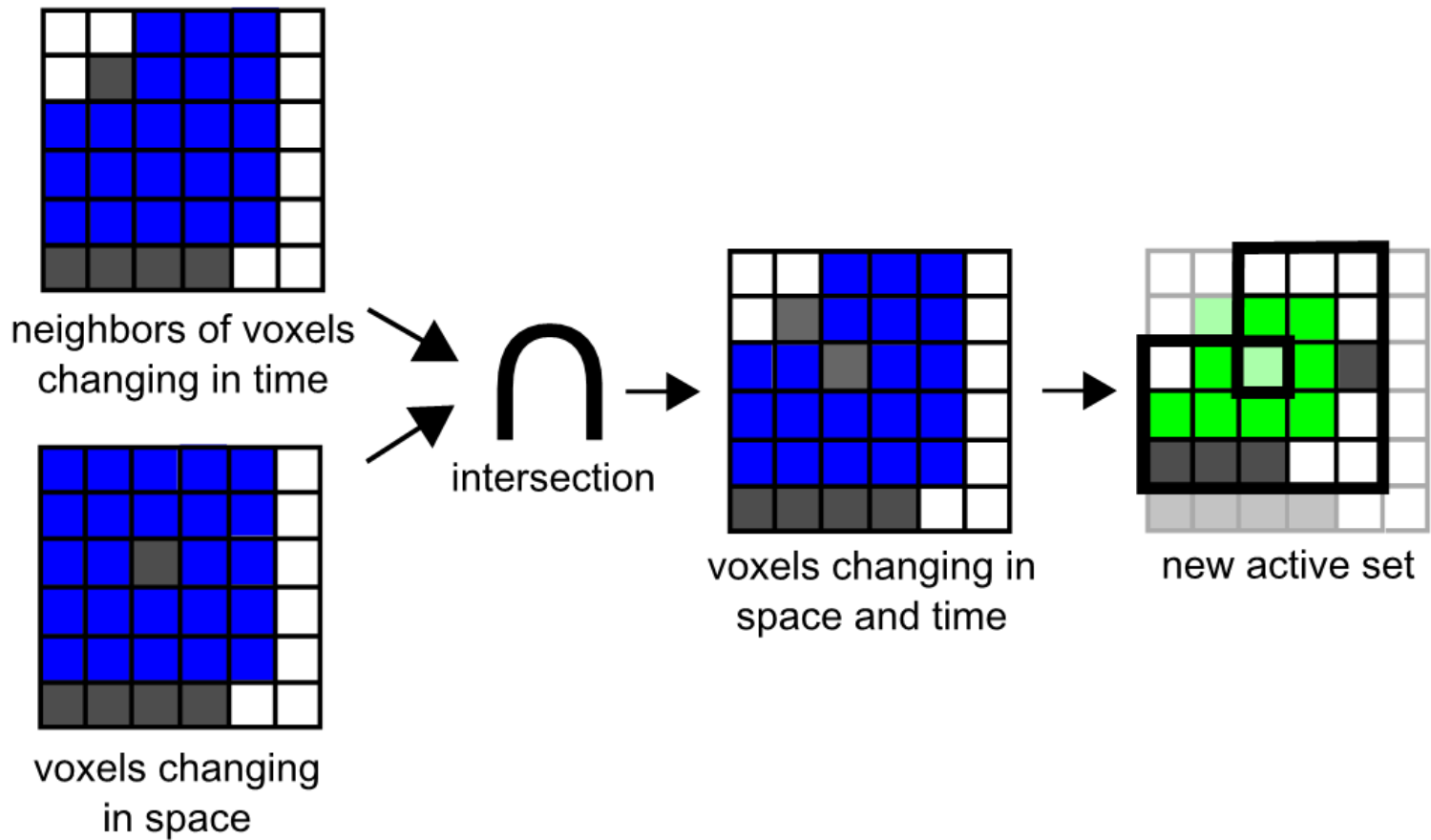
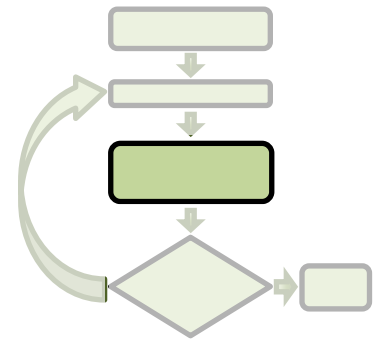
voxels changing  
in space



Find the voxels that are changing in  
*space* and *time*



Find the voxels that are changing in  
*space* and *time*

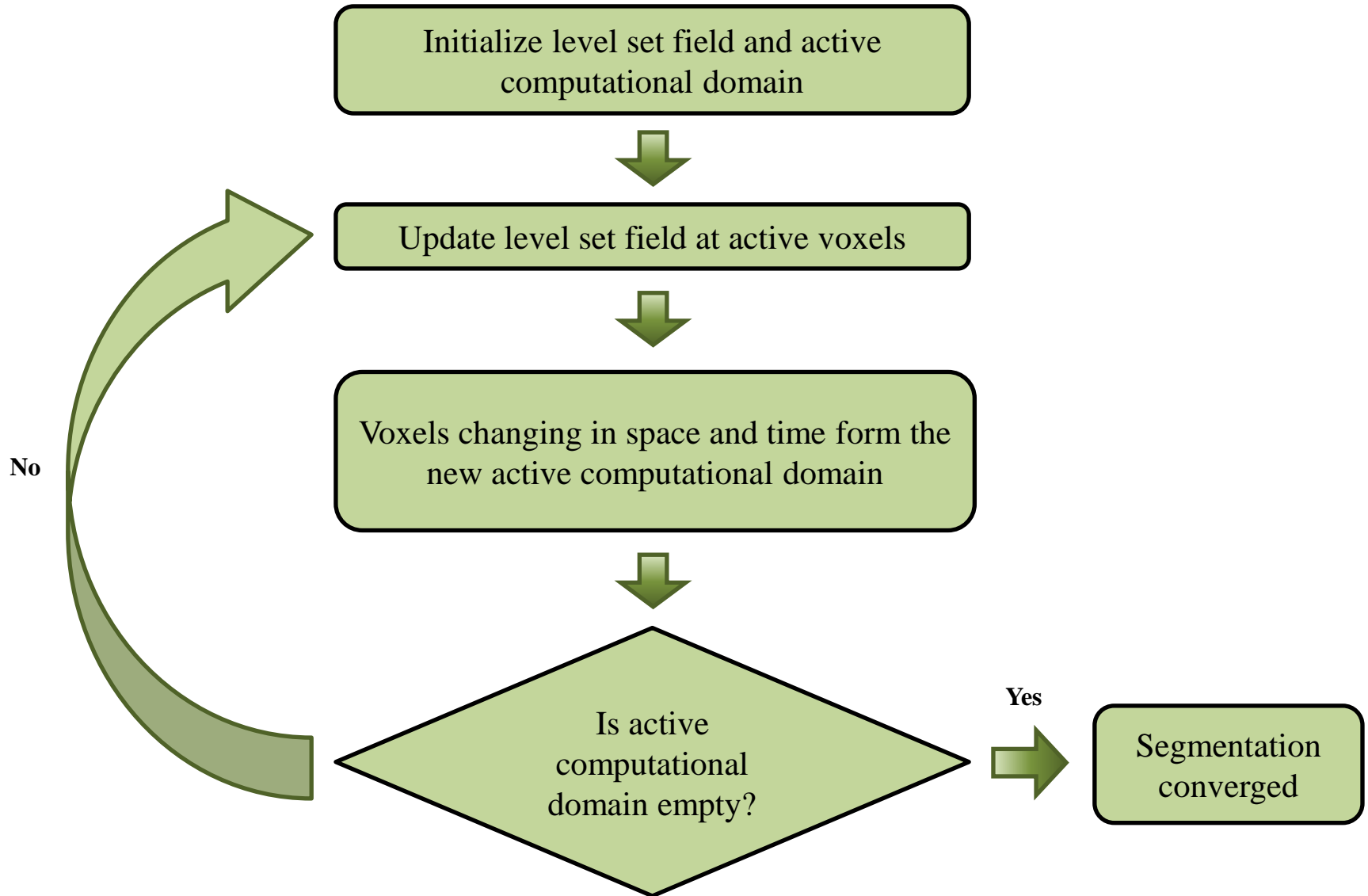


# GPU Algorithm

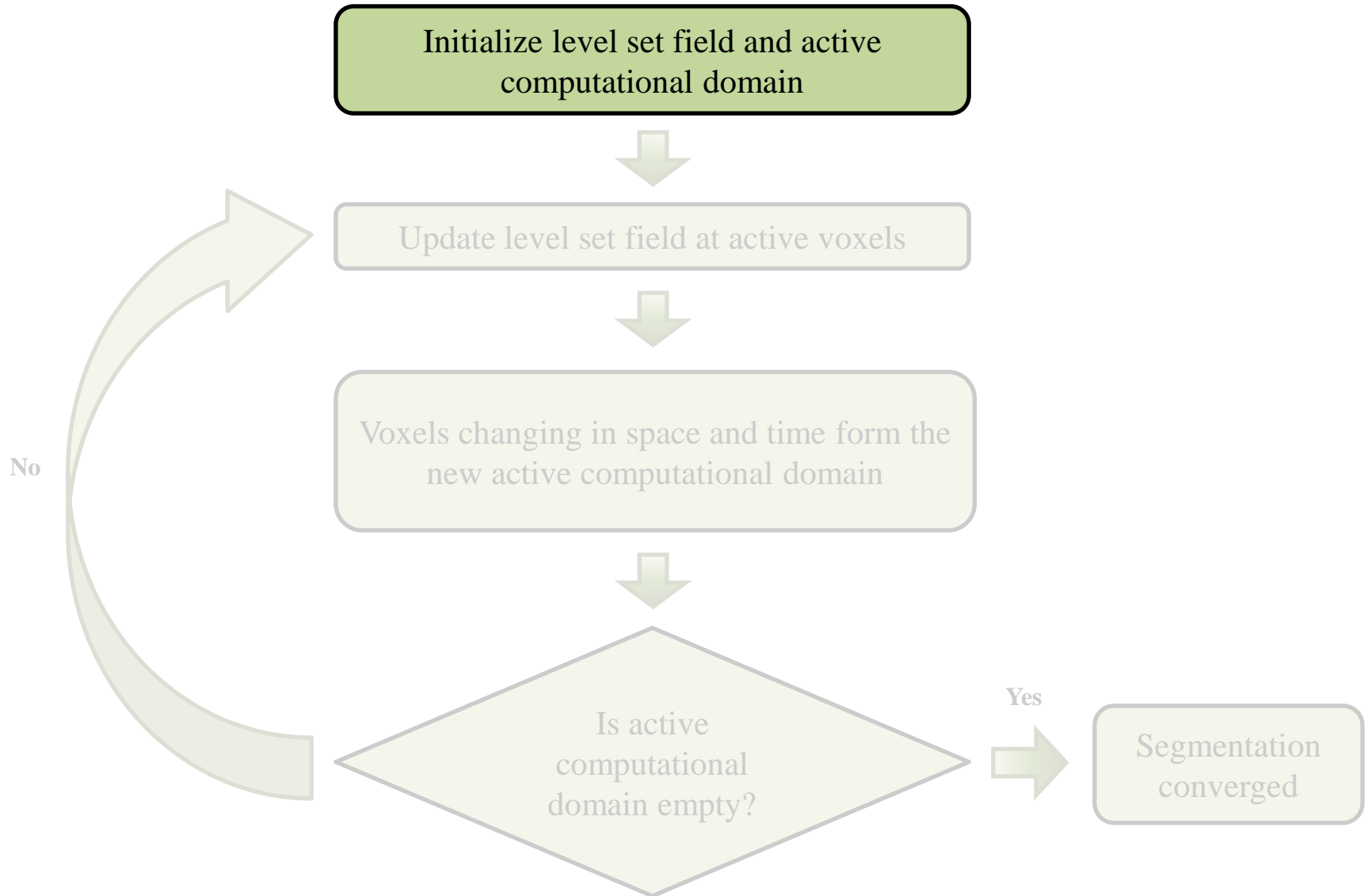
# GPU Algorithm

Intuition: efficiently maintain a dense list of active coordinates

# GPU Algorithm



# GPU Algorithm



# GPU Algorithm

1. Initialize level set field
2. Initialize dense list of active voxels

Update level set field at active voxels

Voxels changing in space and time form the new active computational domain

Is active  
computational  
domain empty?

Yes

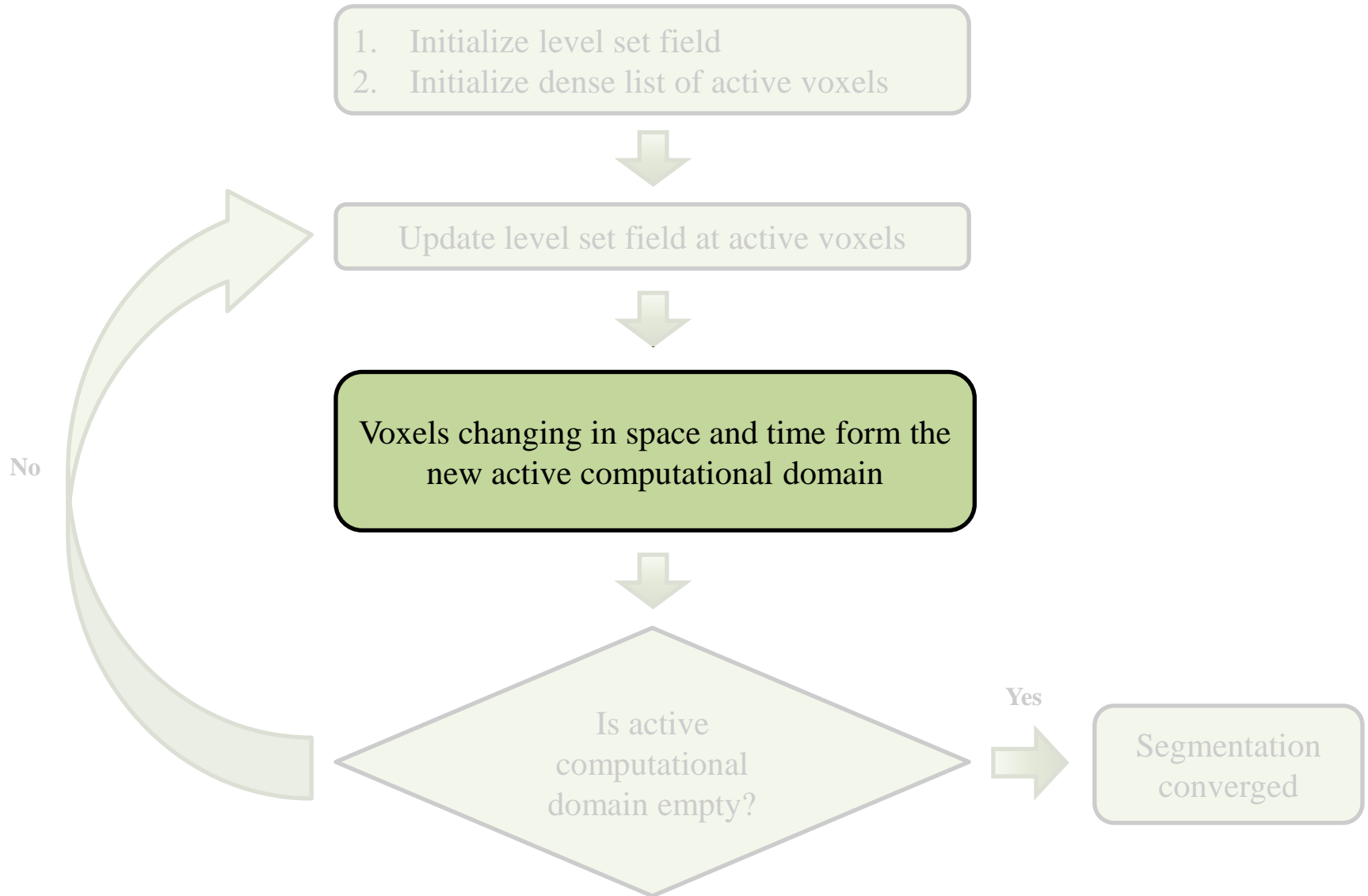
Segmentation  
converged

No

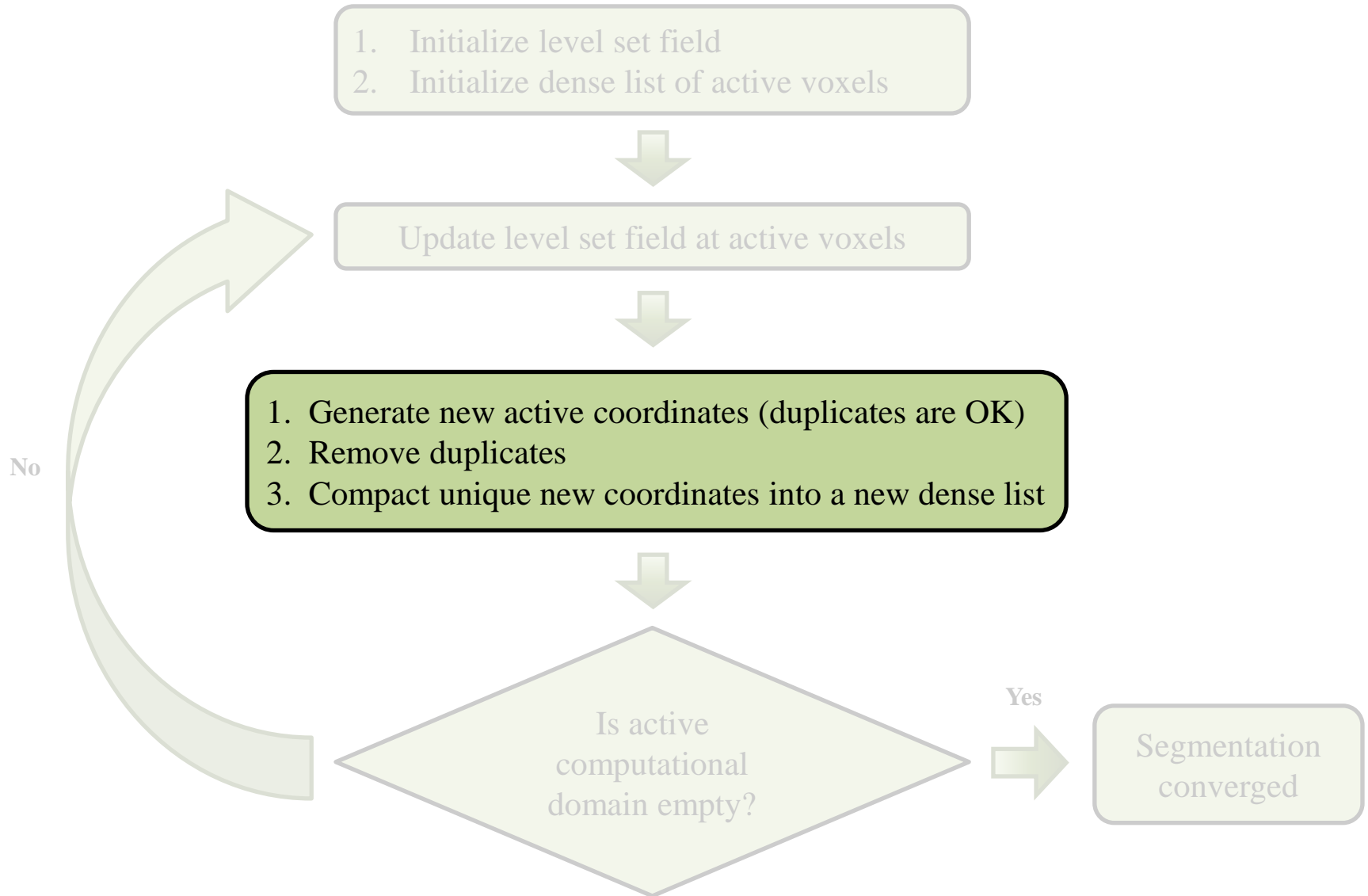




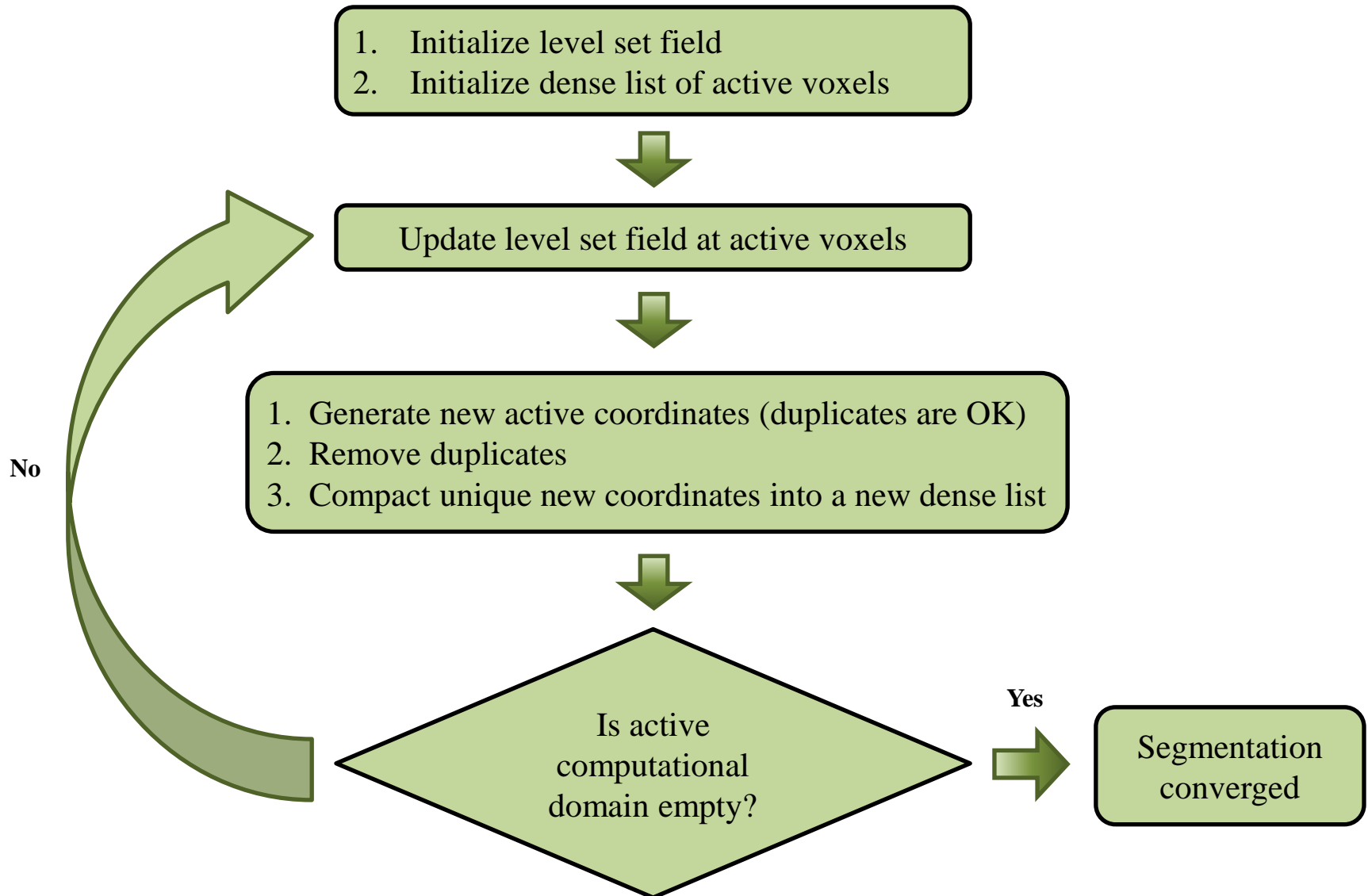
# GPU Algorithm



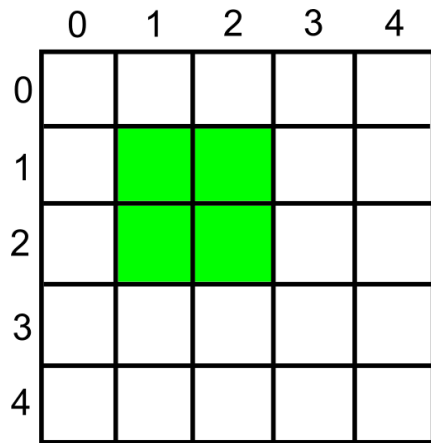
# GPU Algorithm



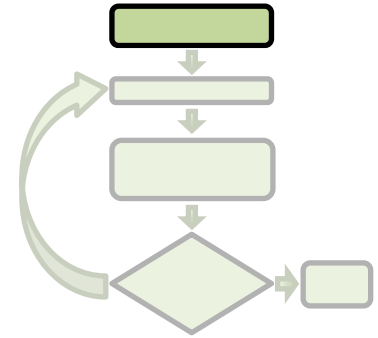
# GPU Algorithm



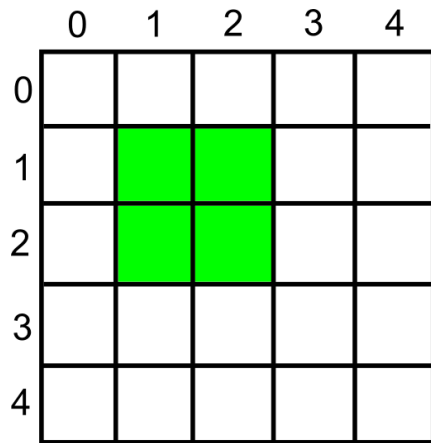
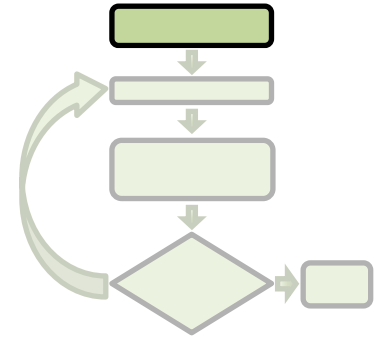
Initialize level set field and dense list of active voxels



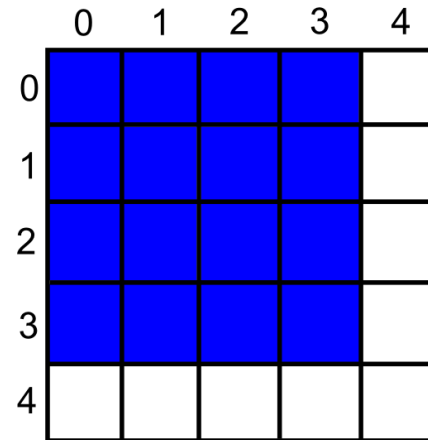
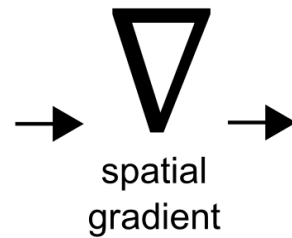
initial level set field



Initialize level set field and dense list of active voxels

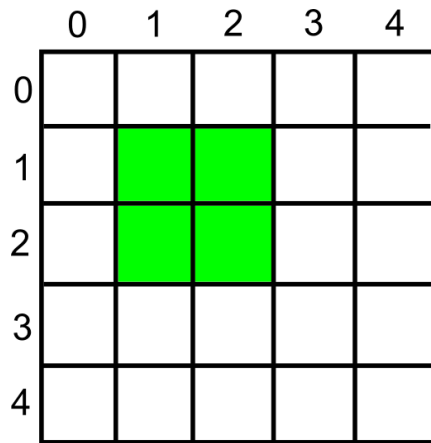
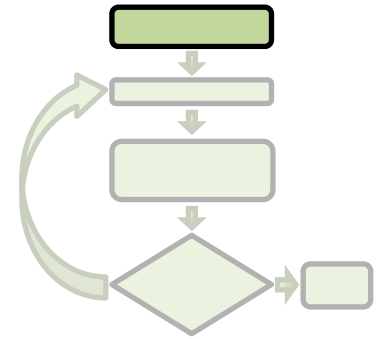


initial level set field

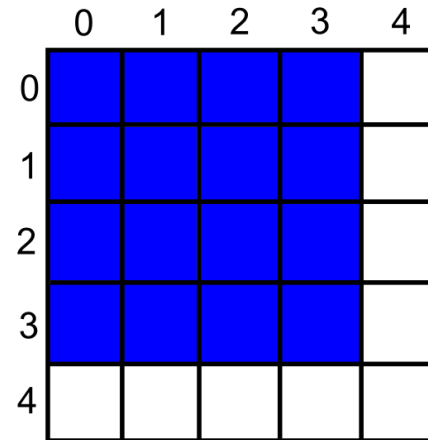
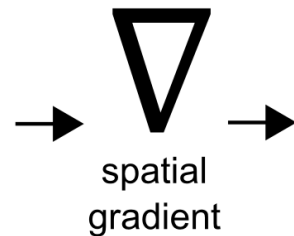


initial active coordinates

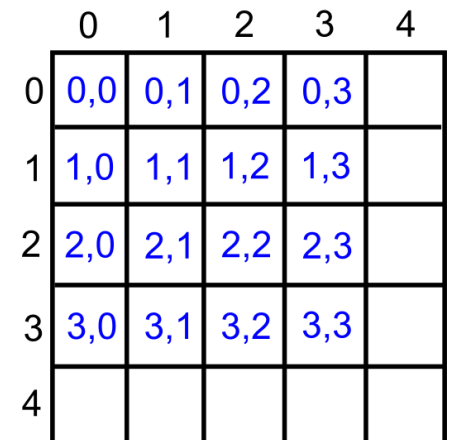
# Initialize level set field and dense list of active voxels



initial level set field

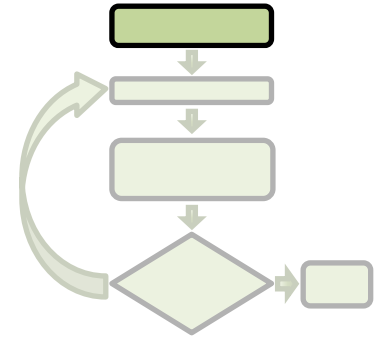


initial active coordinates



write active coordinates  
to scratchpad buffer

# Interpret scratchpad as 1D

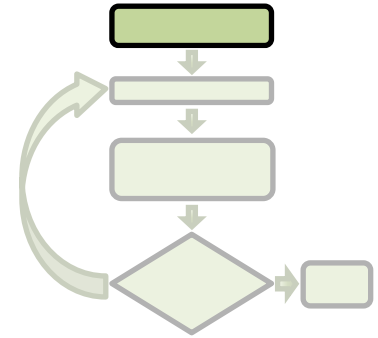


	0	1	2	3	4
0	0,0	0,1	0,2	0,3	
1	1,0	1,1	1,2	1,3	
2	2,0	2,1	2,2	2,3	
3	3,0	3,1	3,2	3,3	
4					

=

0,0	0,1	0,2	0,3		1,0	1,1	1,2	1,3		2,0	2,1	2,2	2,3		3,0	3,1	3,2	3,3								
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24		

# Compact the scratchpad



scratchpad buffer

0,0	0,1	0,2	0,3		1,0	1,1	1,2	1,3		2,0	2,1	2,2	2,3		3,0	3,1	3,2	3,3								
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24		



Stream Compaction

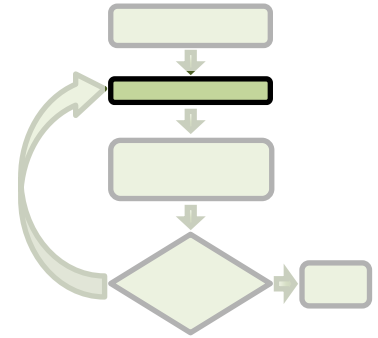


coordinate buffer

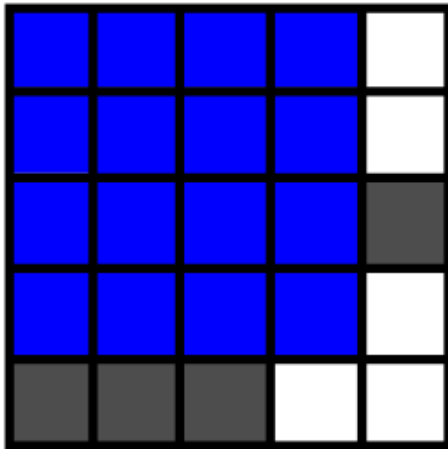
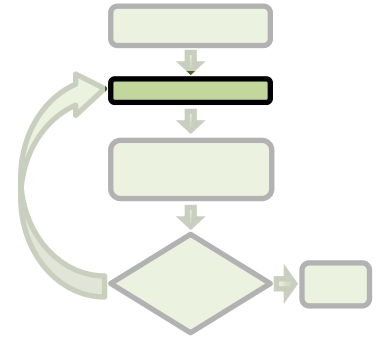
0,0	0,1	0,2	0,3	1,0	1,1	1,2	1,3	2,0	2,1	2,2	2,3	3,0	3,1	3,2	3,3												
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24			



Update the level set field at active  
coordinates

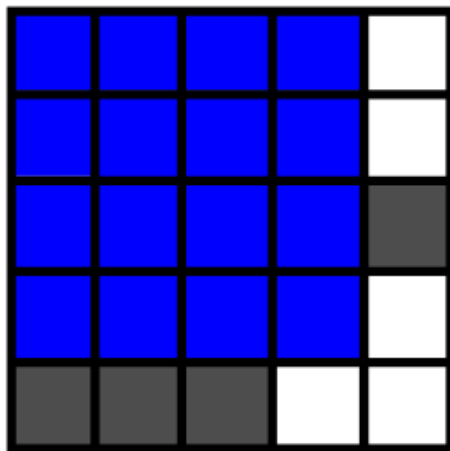
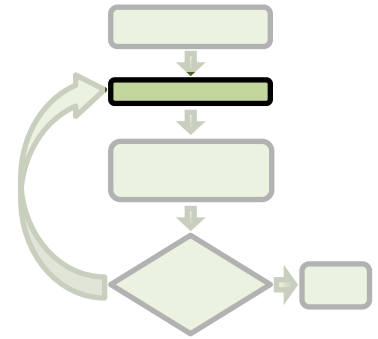


Update the level set field at active  
coordinates



active coordinates

Update the level set field at active coordinates

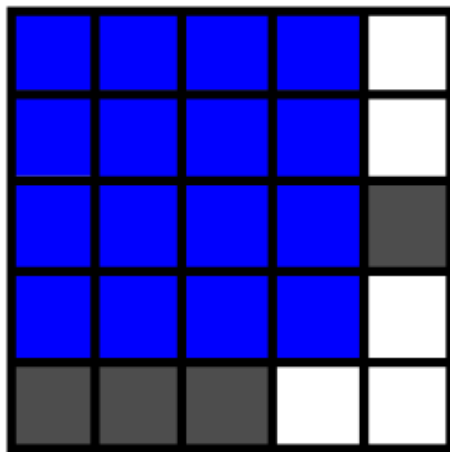
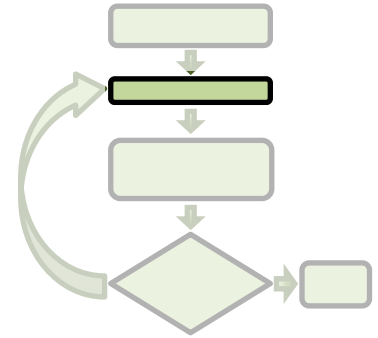


active coordinates



old level set field

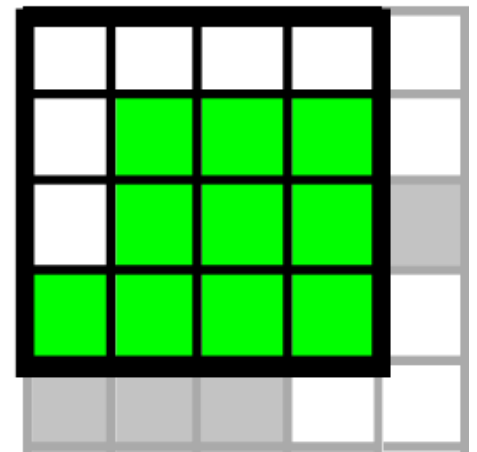
Update the level set field at active coordinates



active coordinates

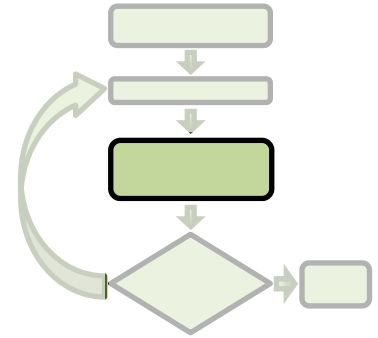


old level set field



new level set field

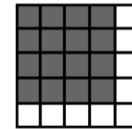
Generate new active coordinates  
(duplicates are OK)



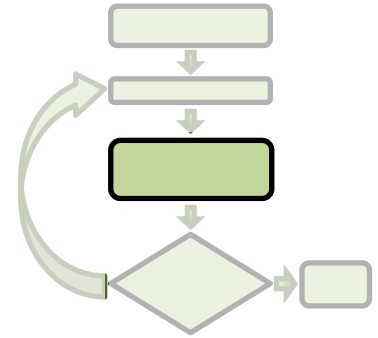
Generate new active coordinates  
(duplicates are OK)

coordinate buffer

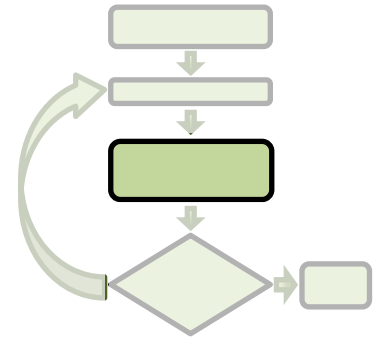
0,0	0,1	0,2	0,3	1,0	1,1	1,2	1,3	2,0	2,1	2,2	2,3	3,0	3,1	3,2	3,3
-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----



old active set

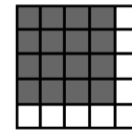


# Generate new active coordinates (duplicates are OK)

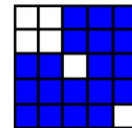


coordinate buffer

0,0	0,1	0,2	0,3	1,0	1,1	1,2	1,3	2,0	2,1	2,2	2,3	3,0	3,1	3,2	3,3
-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----

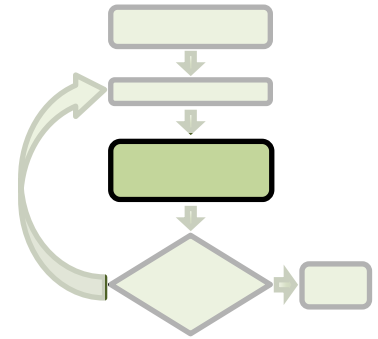
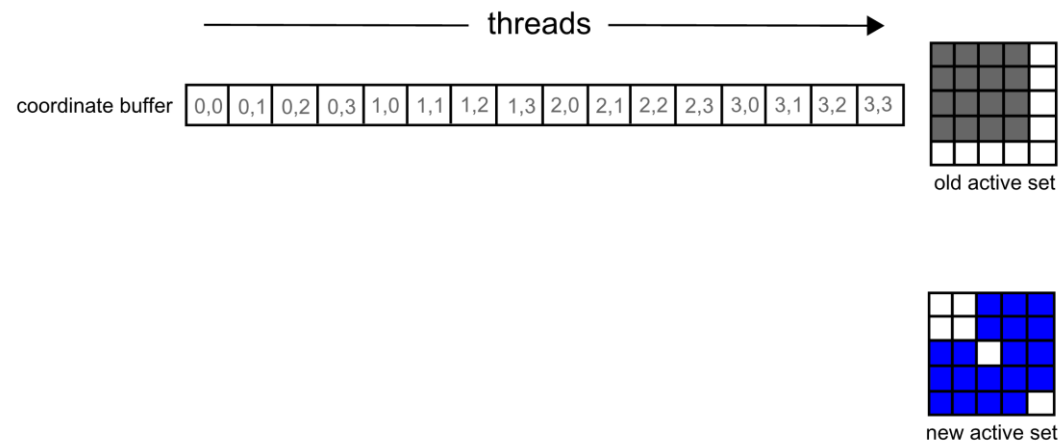


old active set



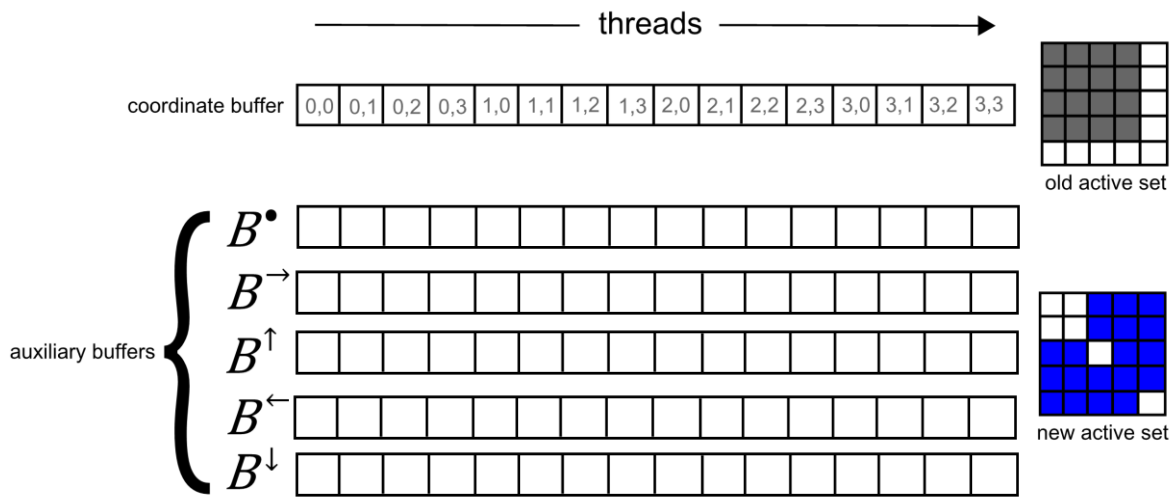
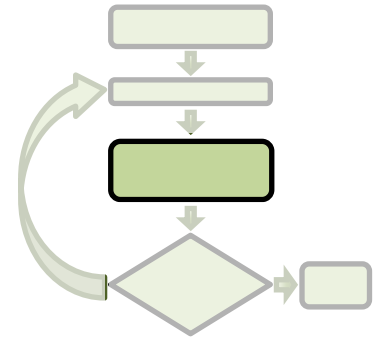
new active set

# Generate new active coordinates (duplicates are OK)

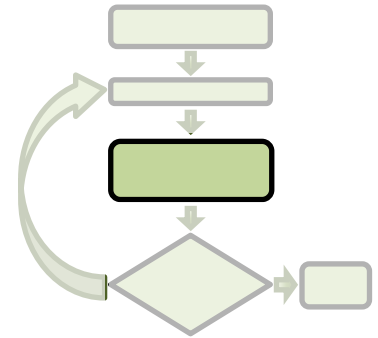




# Generate new active coordinates (duplicates are OK)



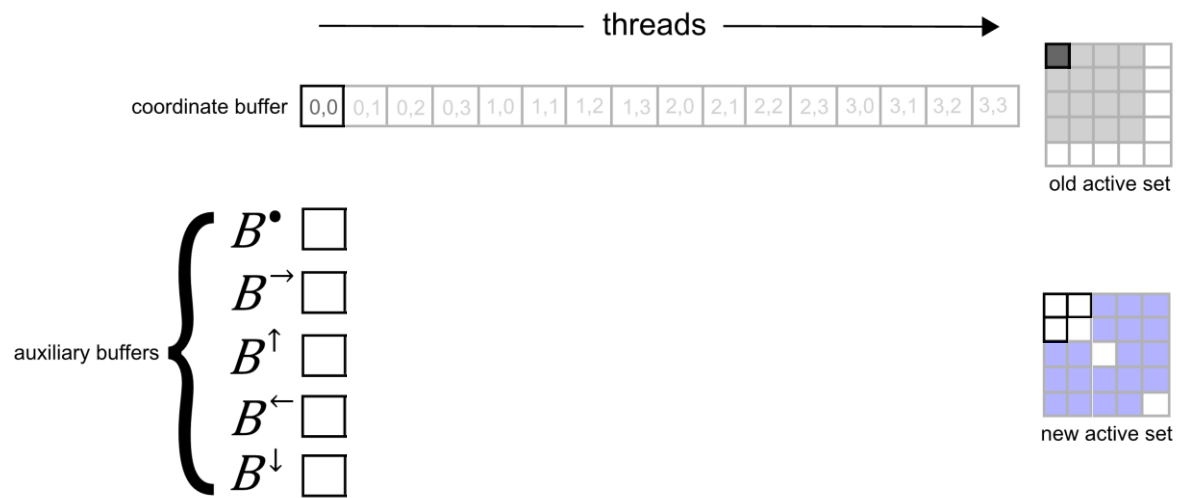
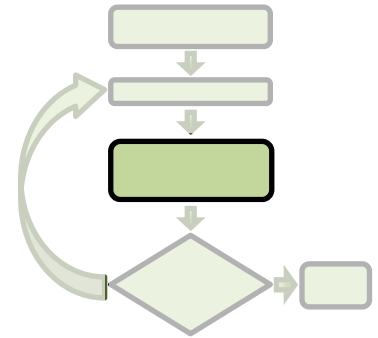
# Generate new active coordinates (duplicates are OK)



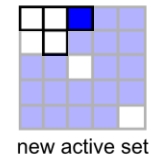
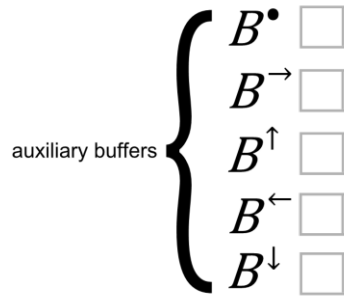
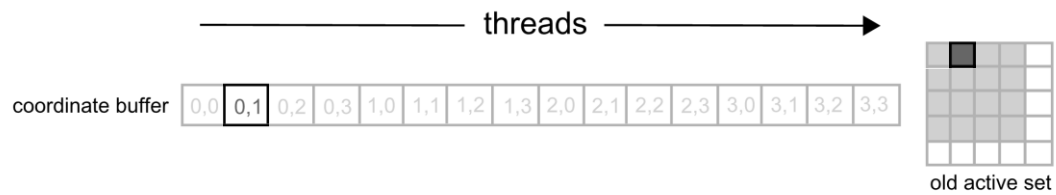
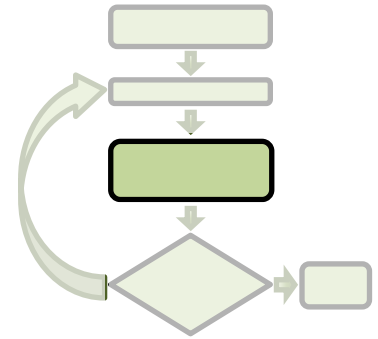
auxiliary buffers

$$\left\{ \begin{array}{l} B^\bullet \\ B^\rightarrow \\ B^\uparrow \\ B^\leftarrow \\ B^\downarrow \end{array} \right.$$

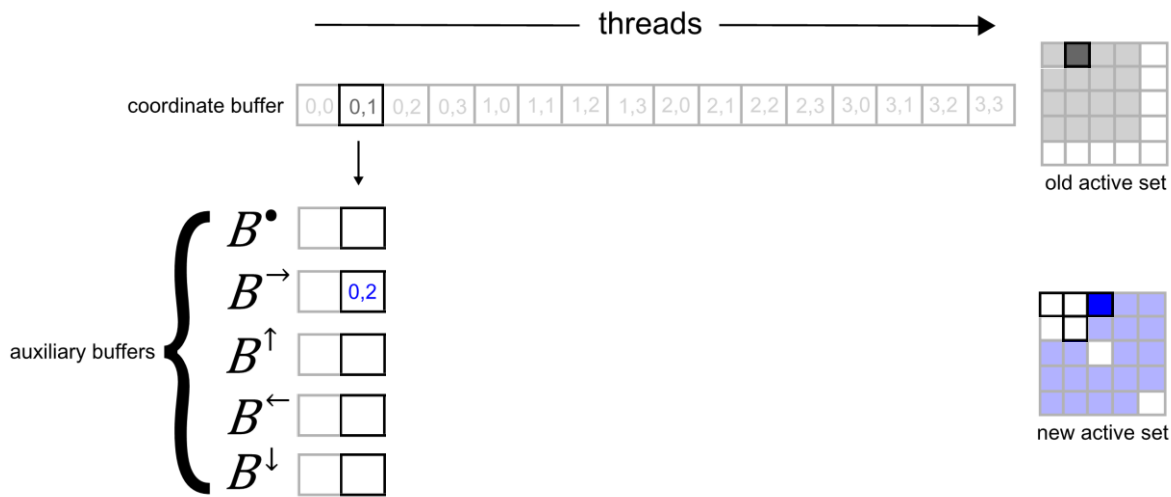
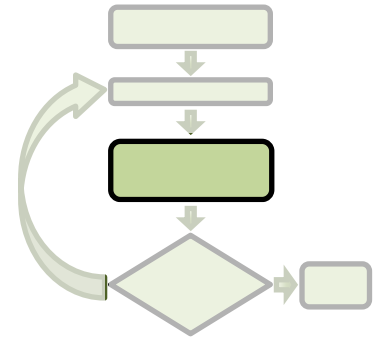

Generate new active coordinates  
(duplicates are OK)



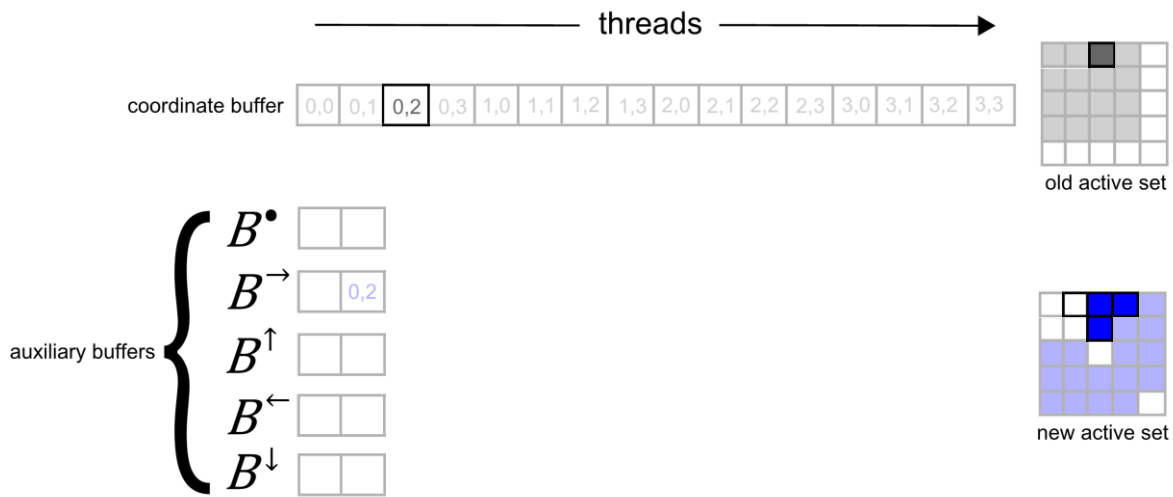
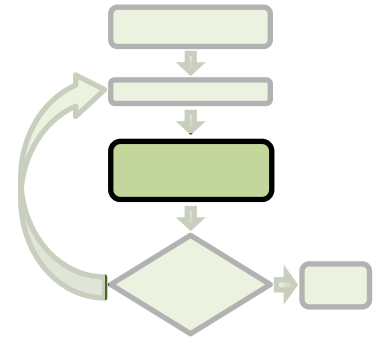
# Generate new active coordinates (duplicates are OK)



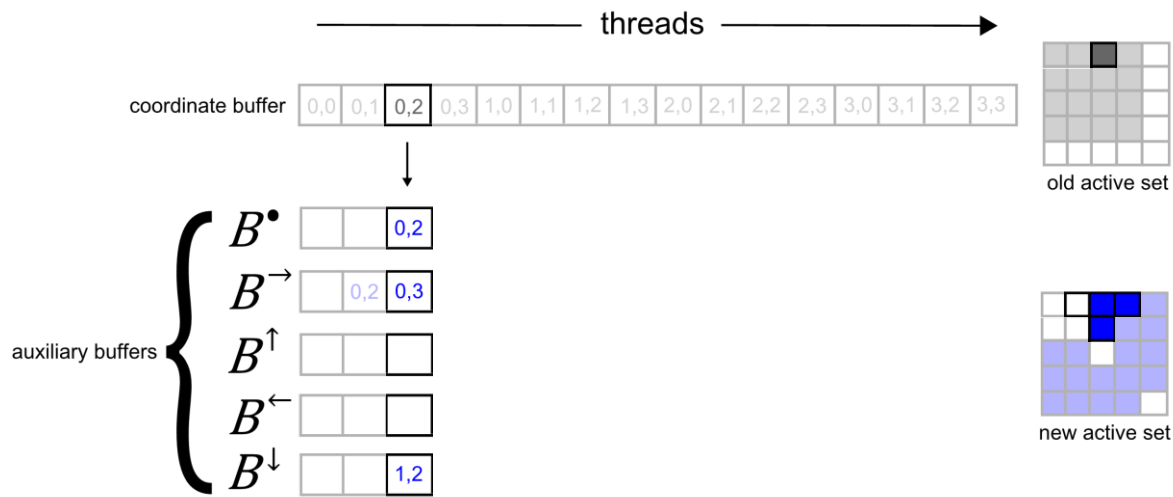
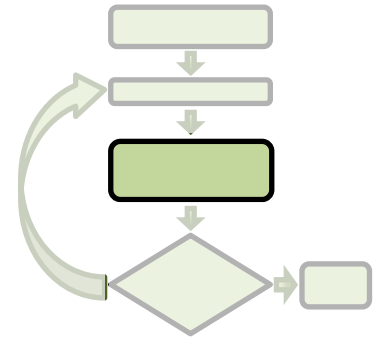
# Generate new active coordinates (duplicates are OK)



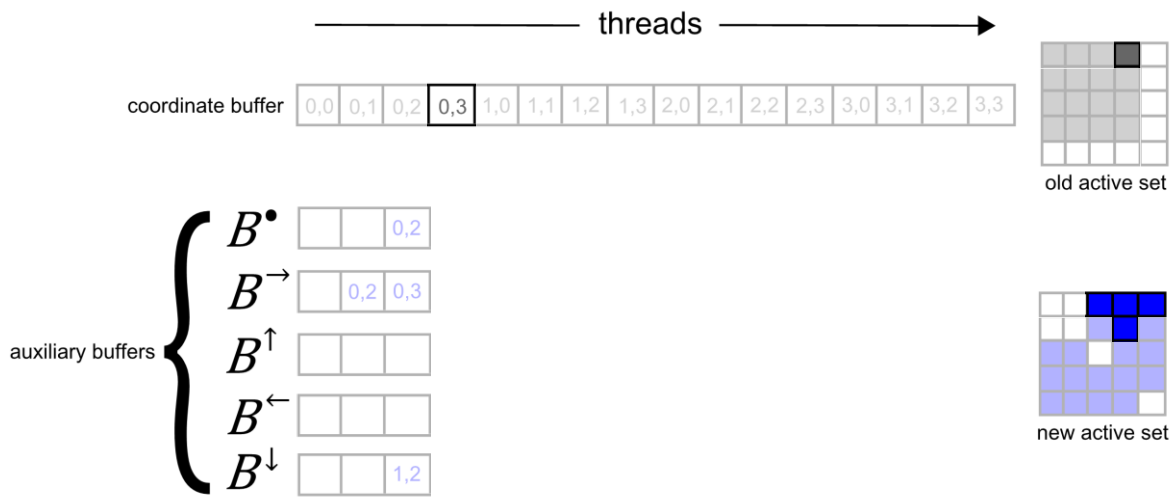
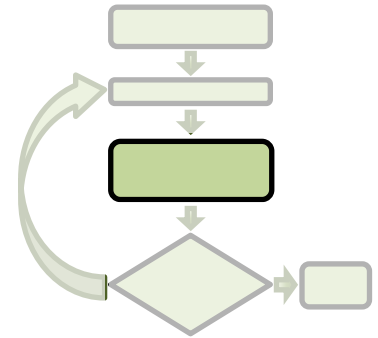
# Generate new active coordinates (duplicates are OK)



# Generate new active coordinates (duplicates are OK)

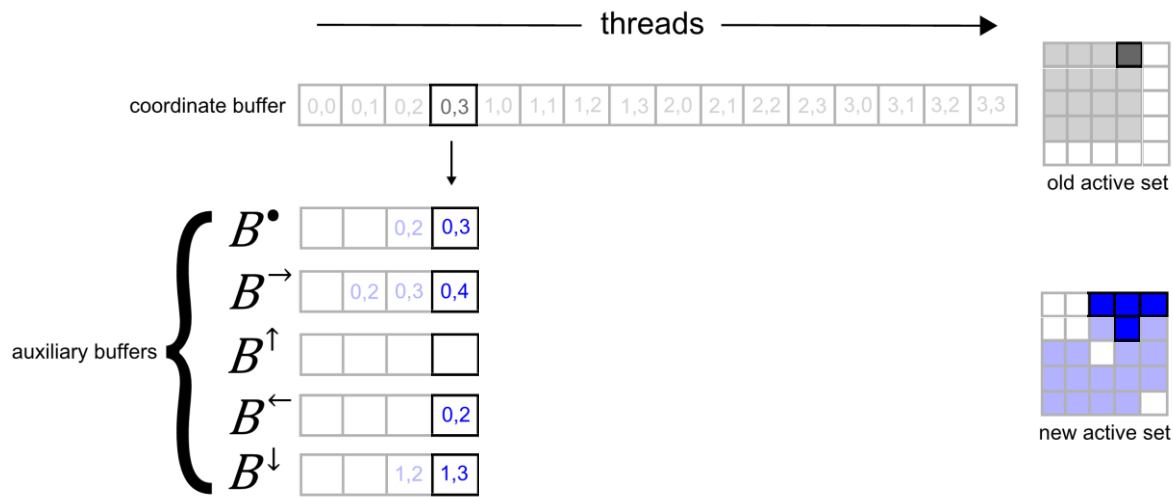
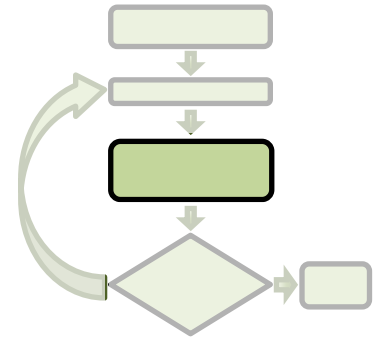


# Generate new active coordinates (duplicates are OK)

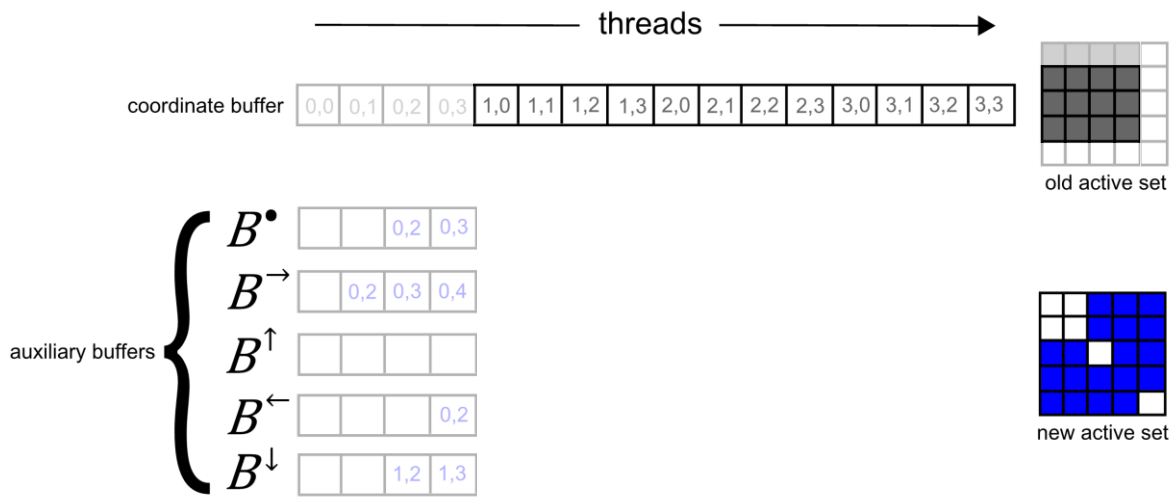
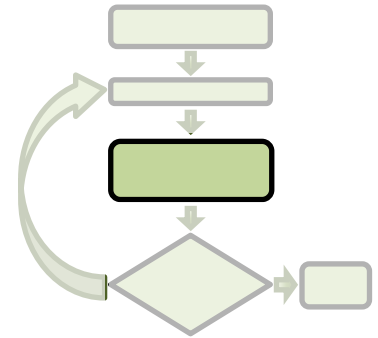




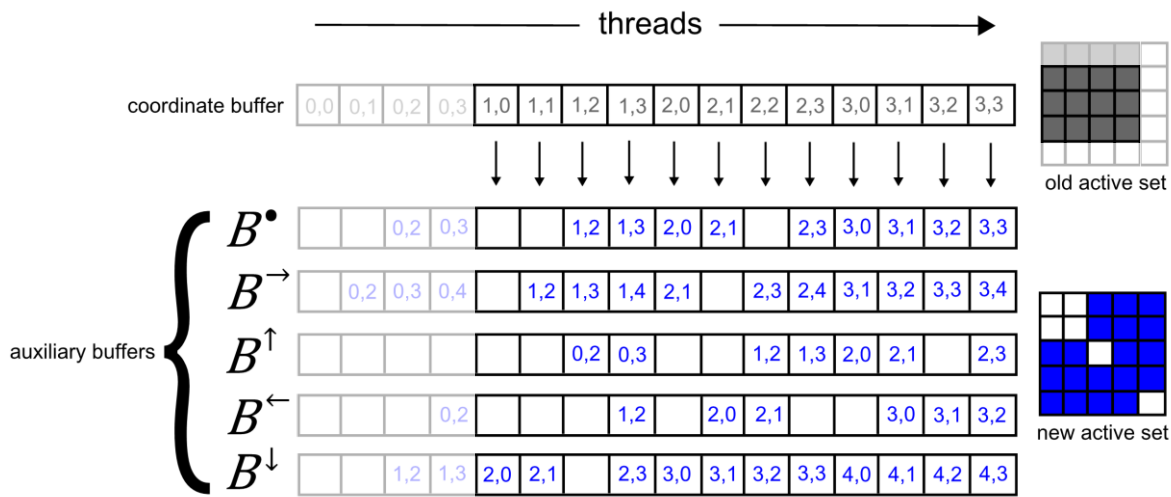
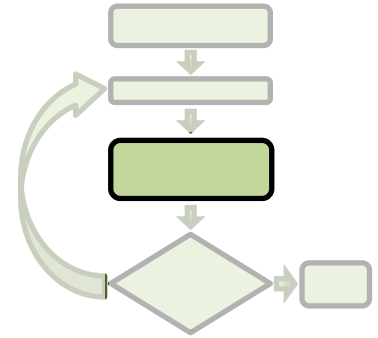
# Generate new active coordinates (duplicates are OK)



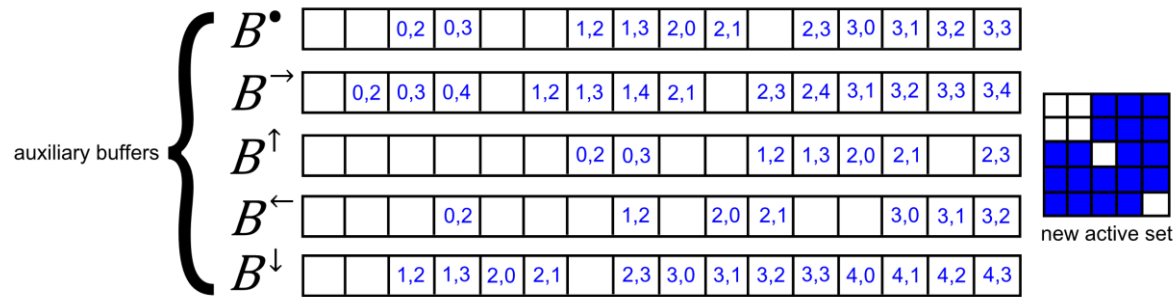
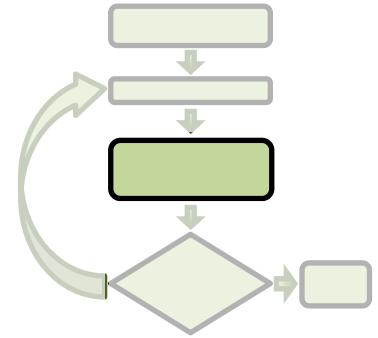
# Generate new active coordinates (duplicates are OK)



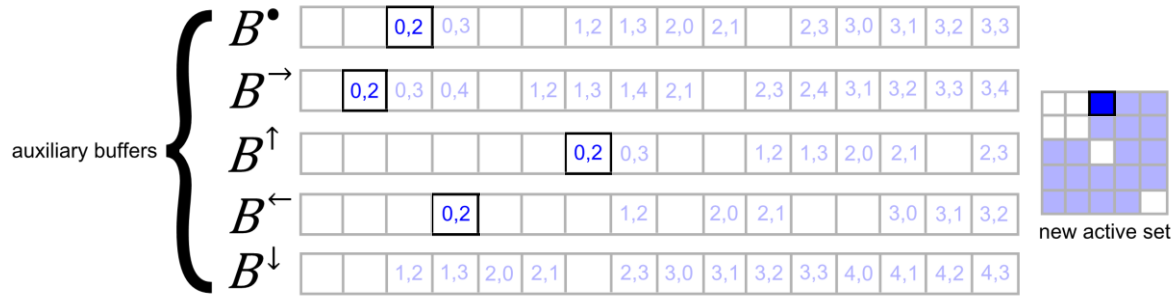
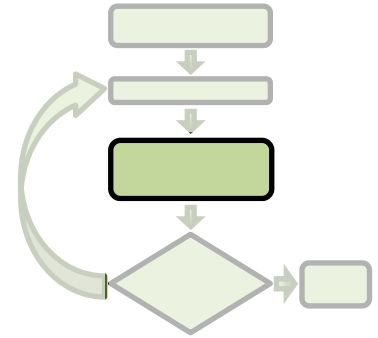
# Generate new active coordinates (duplicates are OK)



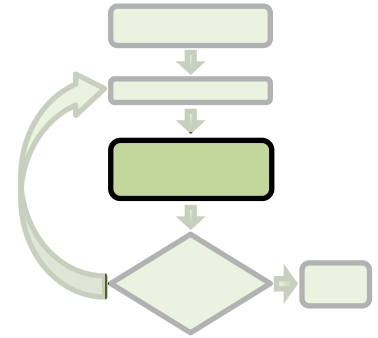
# Generate new active coordinates (duplicates are OK)



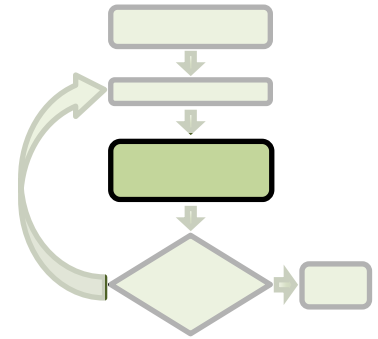
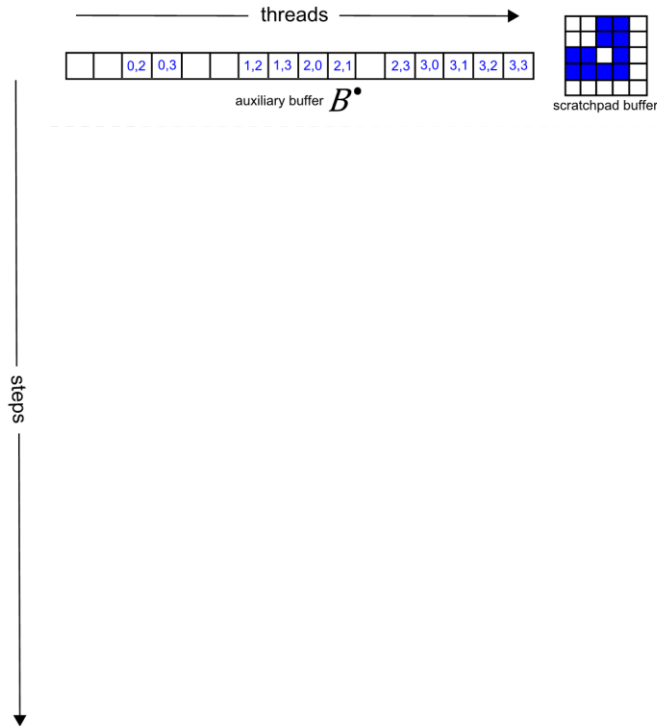
# Generate new active coordinates (duplicates are OK)



# Remove duplicates



# Remove duplicates



----- global synchronization

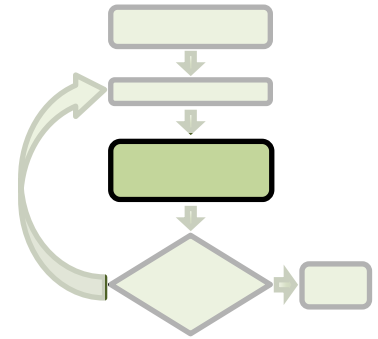
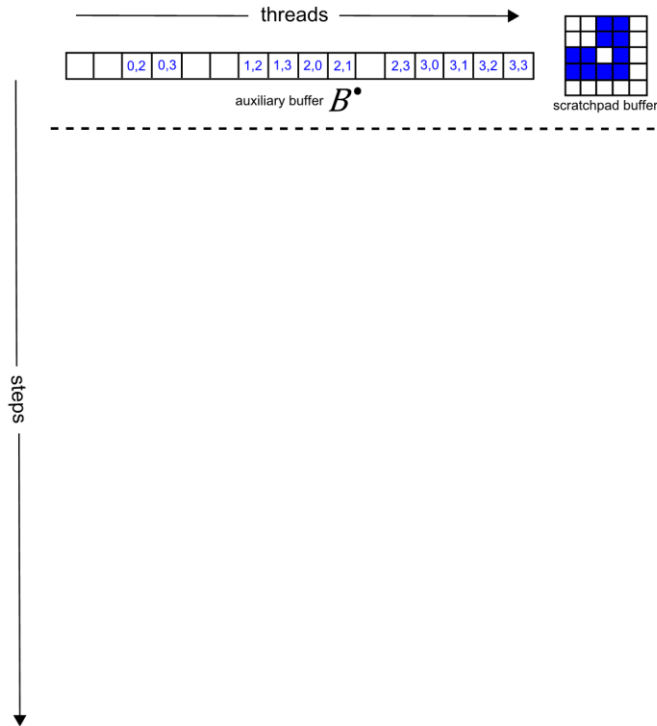


not previously tagged → tag in scratchpad



previously tagged → remove from auxiliary

# Remove duplicates



----- global synchronization

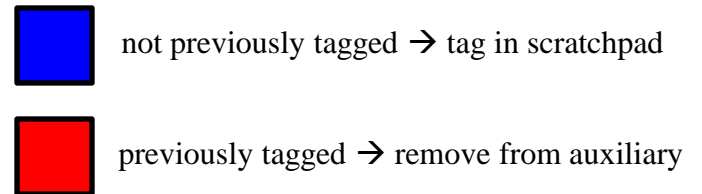
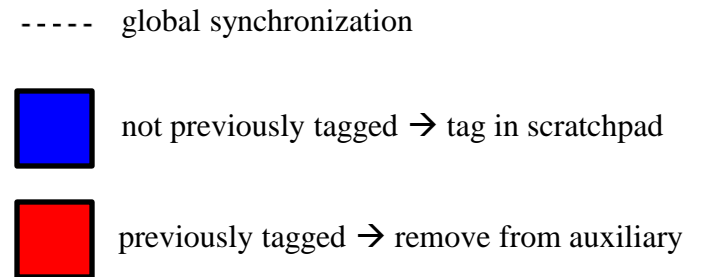


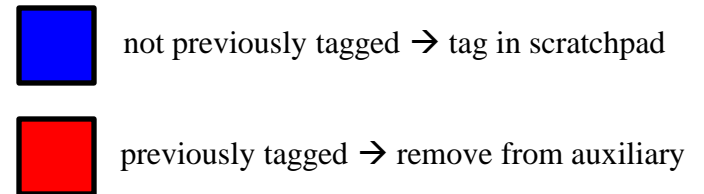
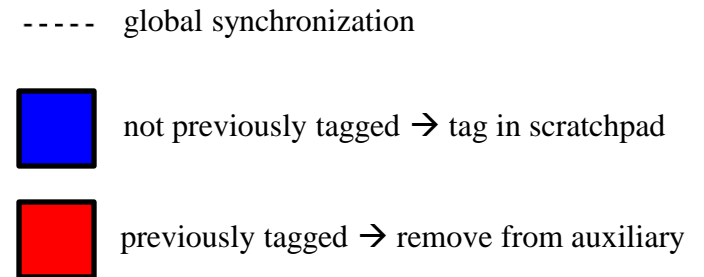
not previously tagged → tag in scratchpad



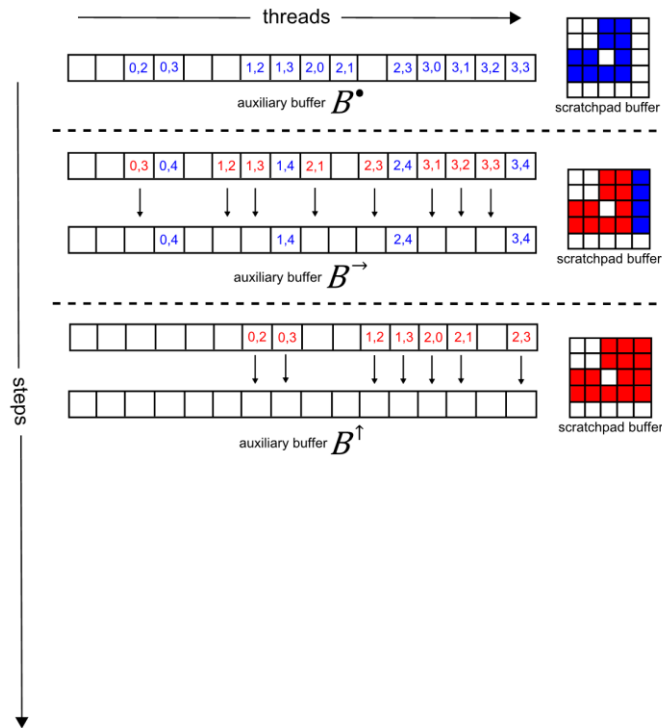
previously tagged → remove from auxiliary







# Remove duplicates



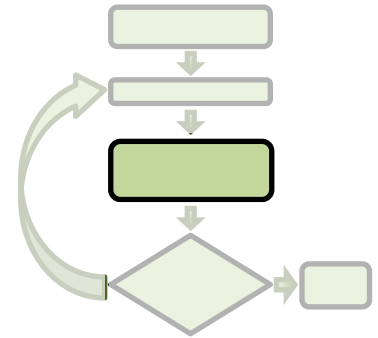
----- global synchronization



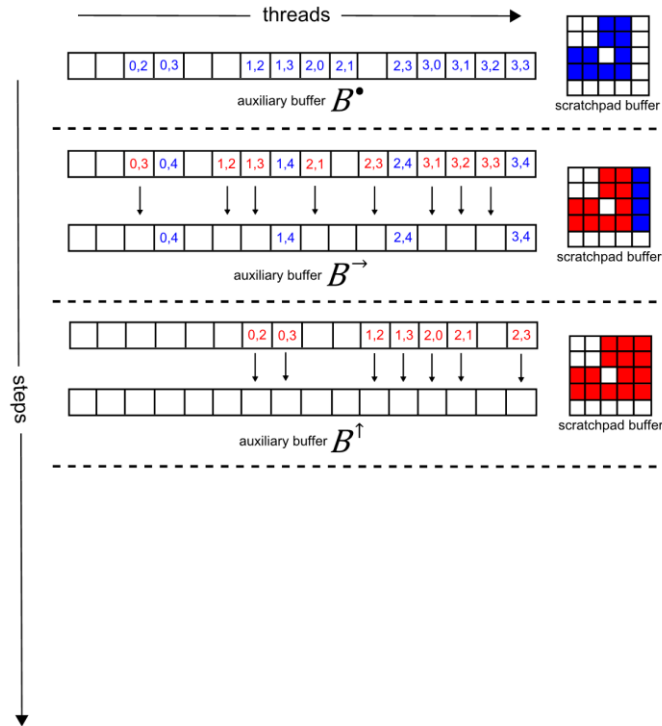
not previously tagged → tag in scratchpad



previously tagged → remove from auxiliary



# Remove duplicates



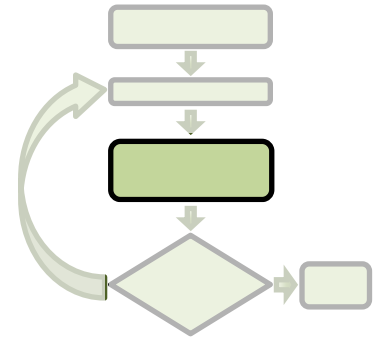
----- global synchronization



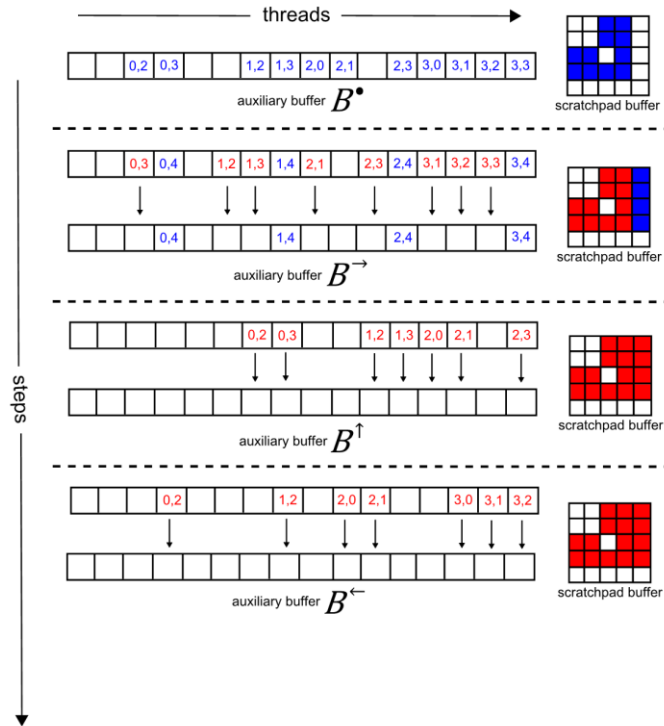
not previously tagged → tag in scratchpad



previously tagged → remove from auxiliary



# Remove duplicates



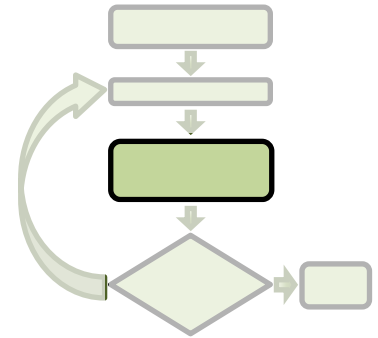
----- global synchronization



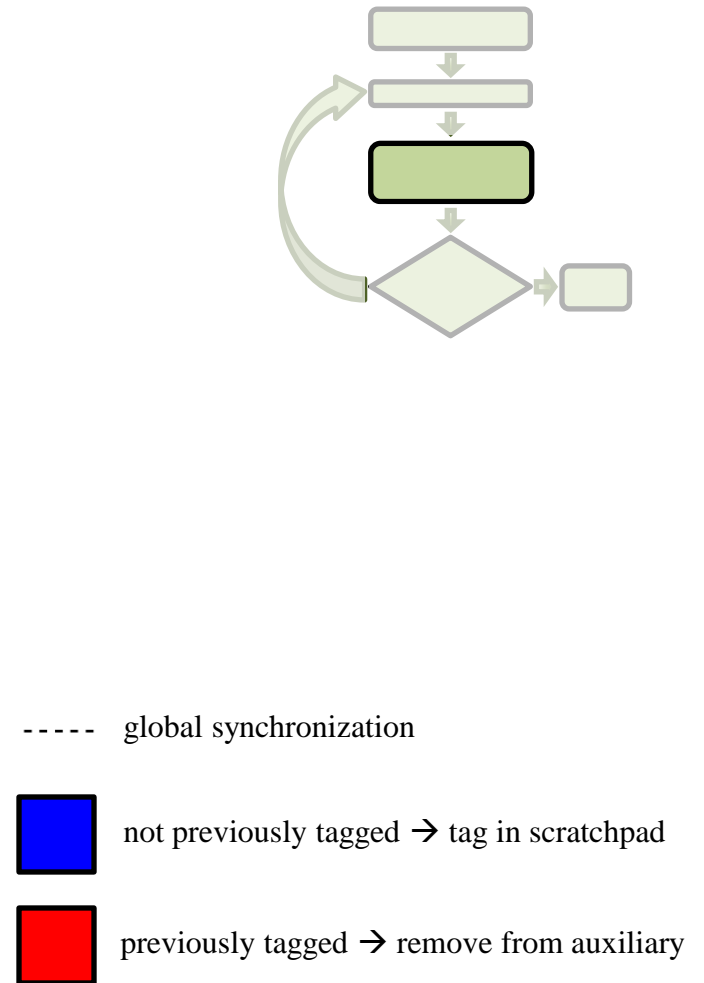
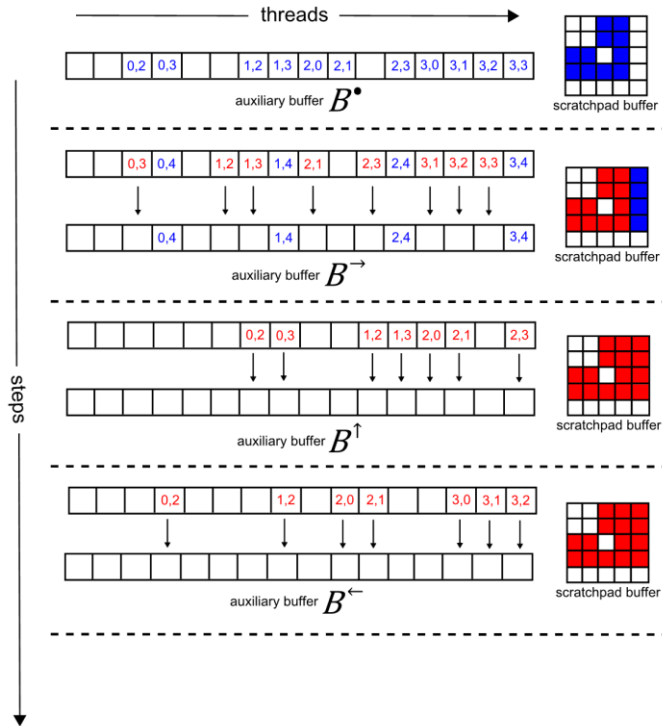
not previously tagged → tag in scratchpad



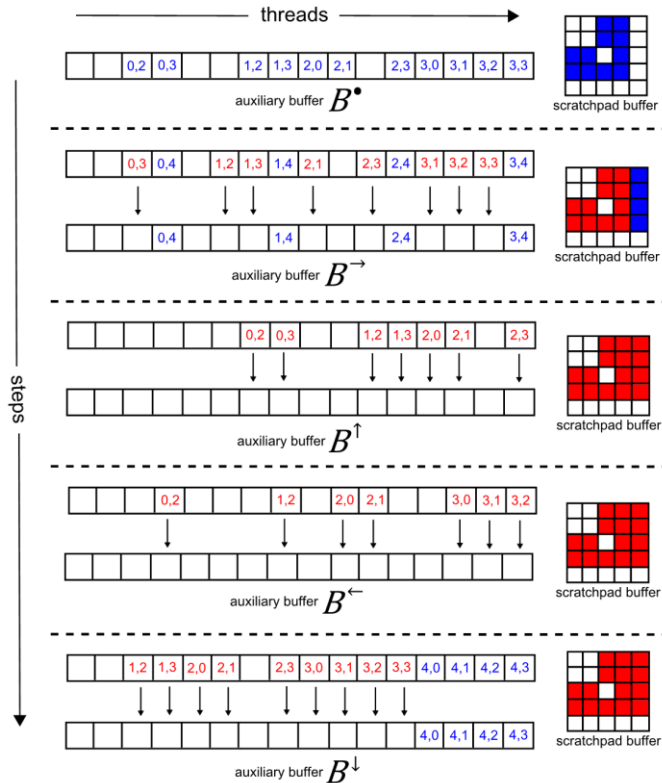
previously tagged → remove from auxiliary



# Remove duplicates



# Remove duplicates



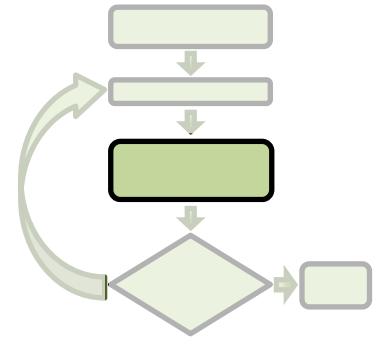
----- global synchronization



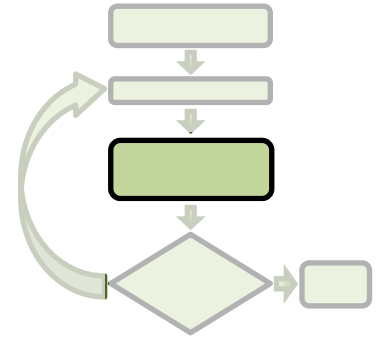
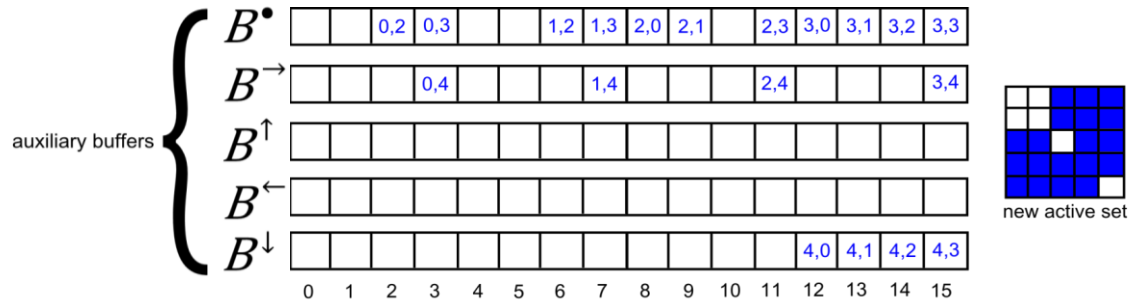
not previously tagged → tag in scratchpad



previously tagged → remove from auxiliary

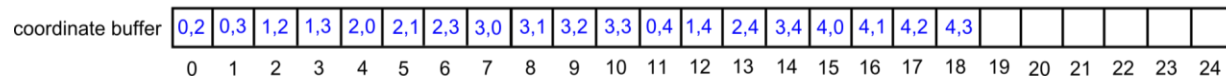
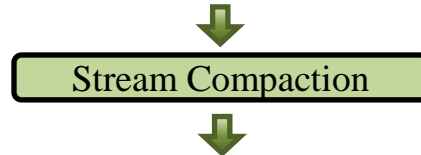
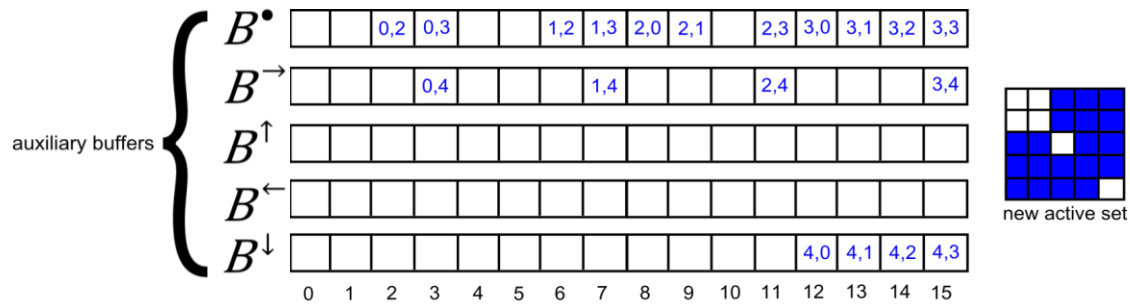
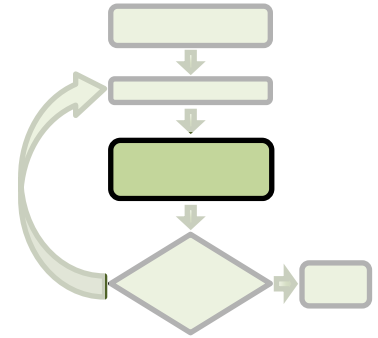


# Compact the auxiliary buffers





# Compact the auxiliary buffers



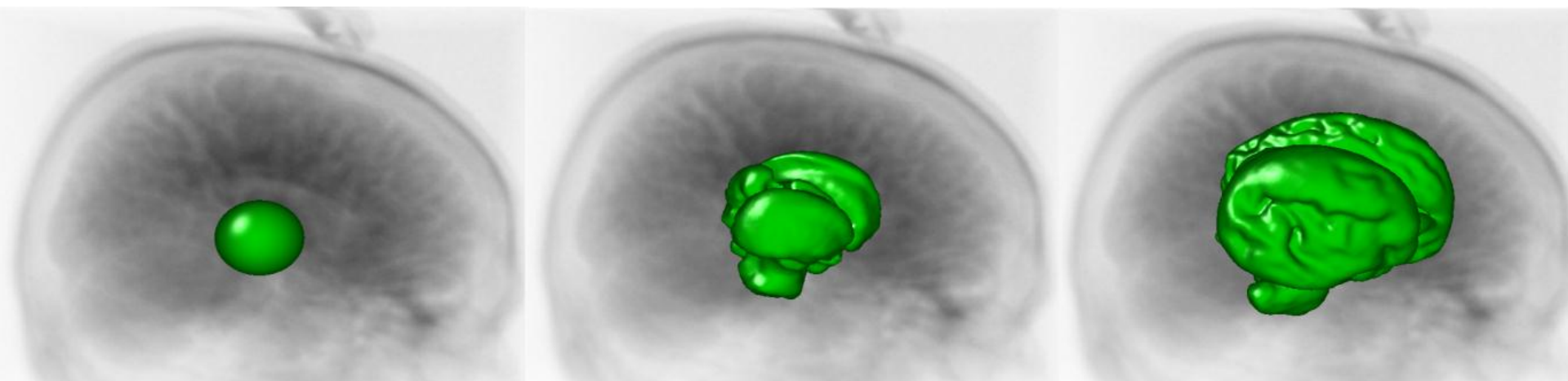
# Asymptotic Complexity

Previous methods:  $\mathbf{O}(n)$  steps per level set update

# Asymptotic Complexity

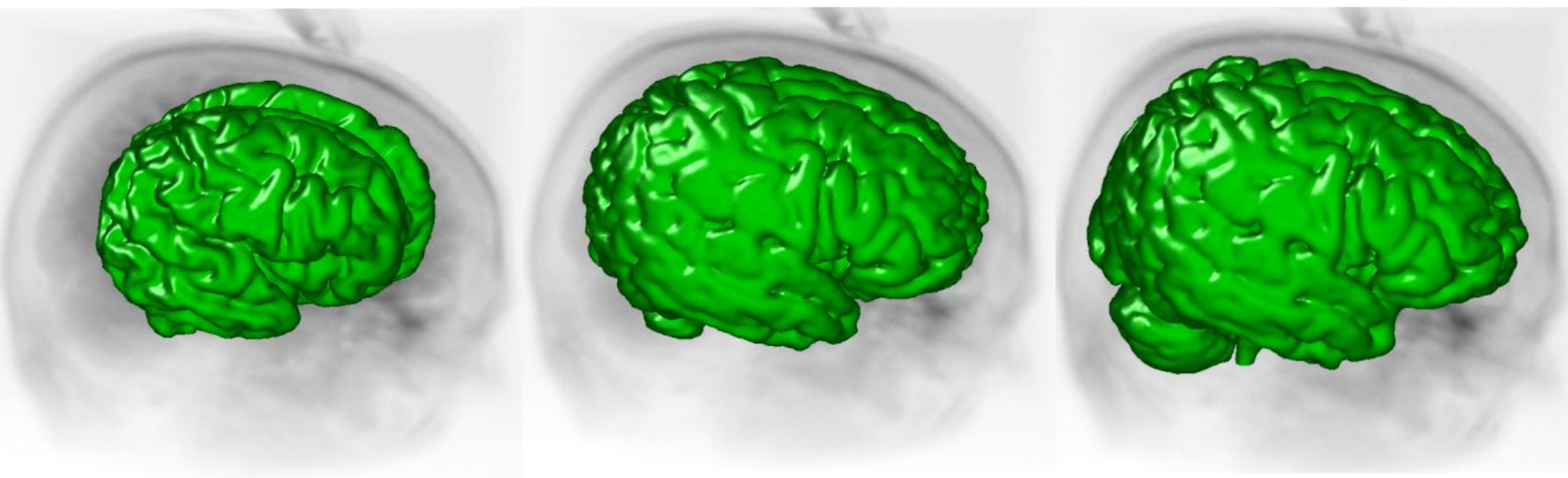
Our method:

**$O(\log n)$**  steps per level set update

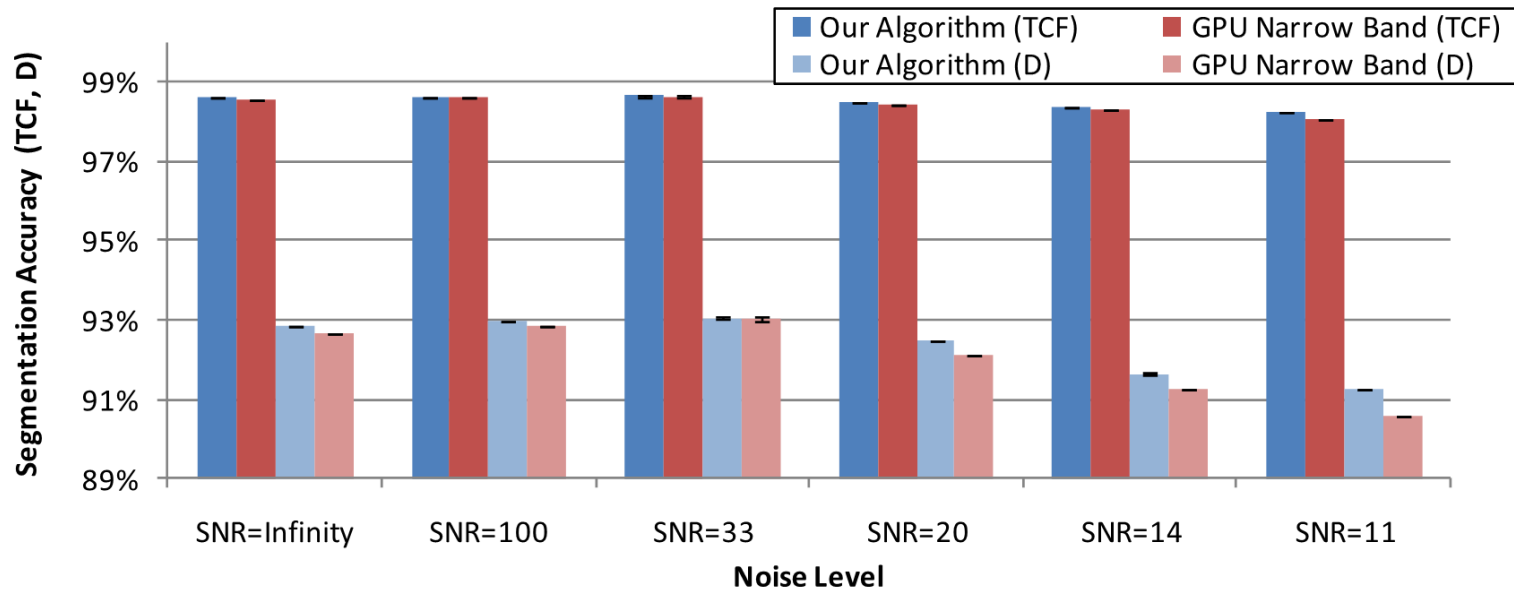


# Experimental Methodology

$256^3$  head MRI



# Our algorithm is slightly more accurate

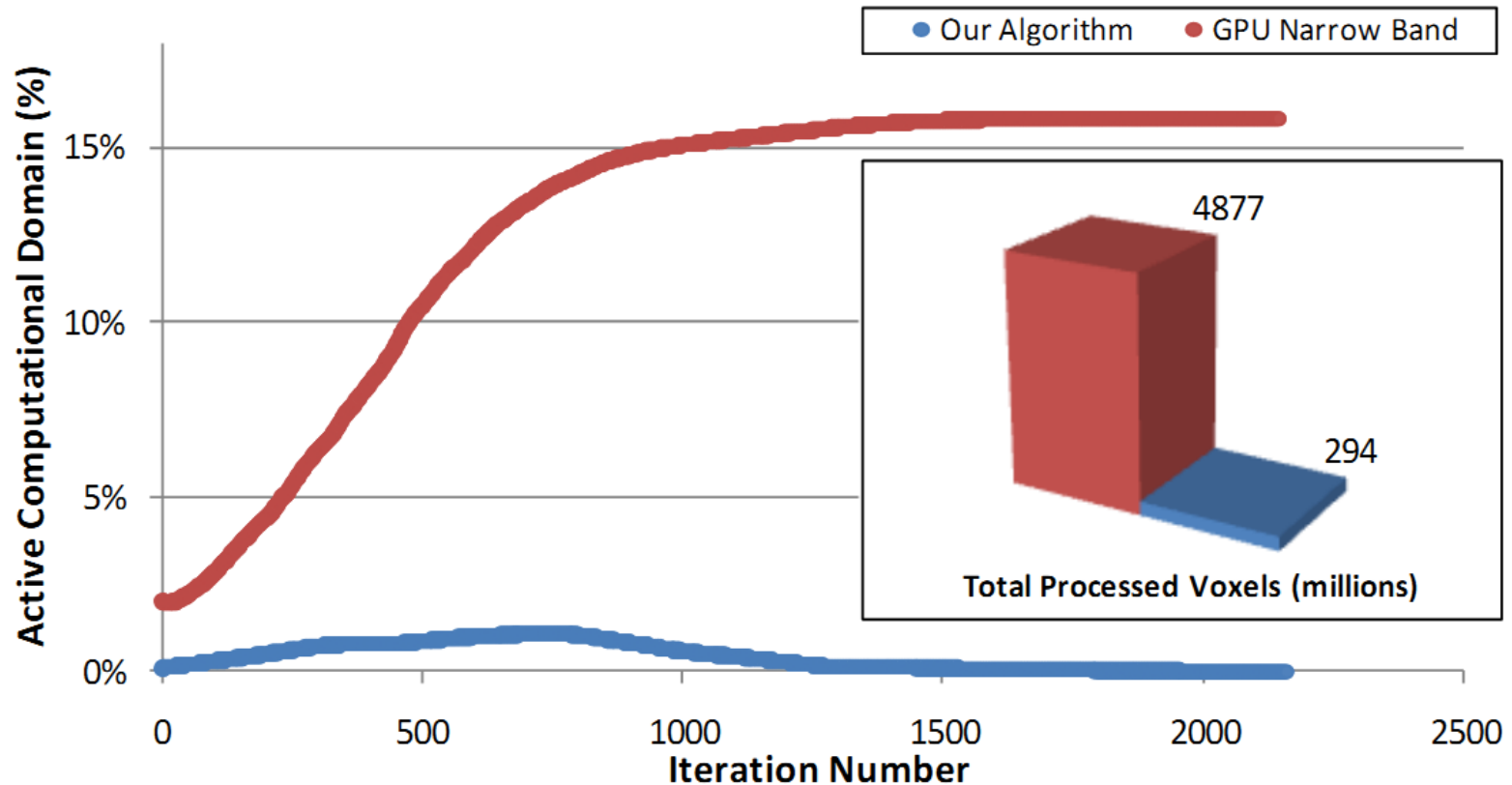


SNR = Signal-to-noise Ratio

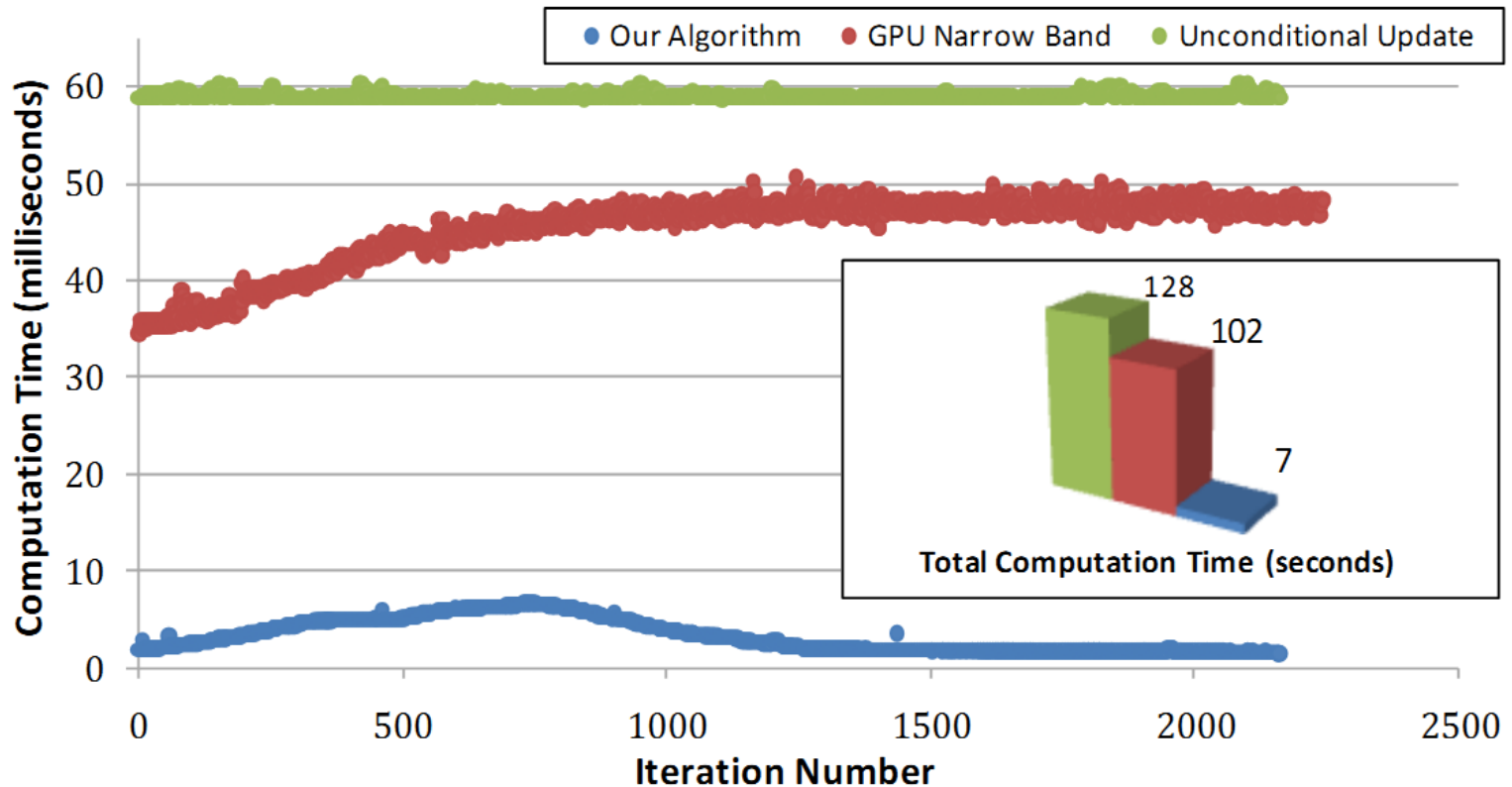
D = Dice Coefficient

TCF = Total Correct Fraction of Labeled Voxels

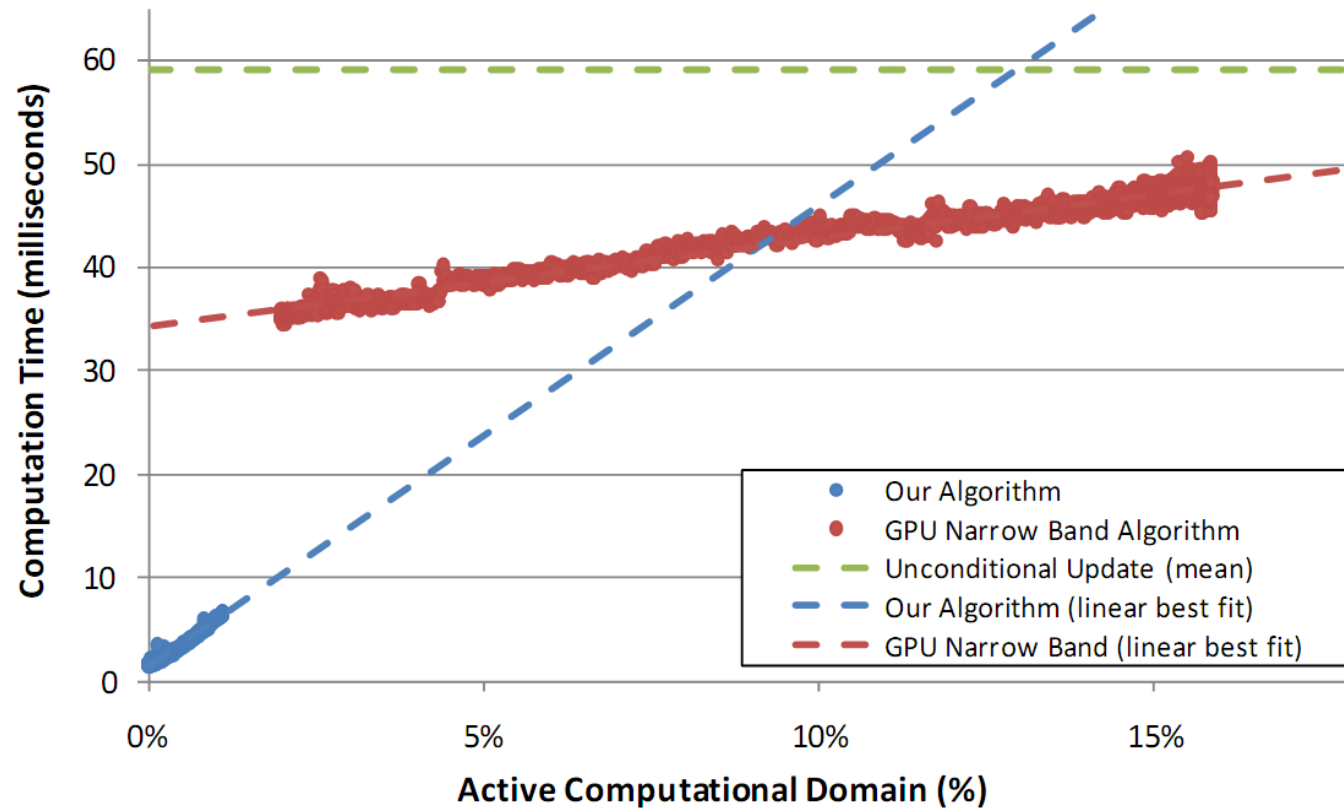
# Our algorithm performs $16\times$ less GPU work



# Our algorithm is $14\times$ faster

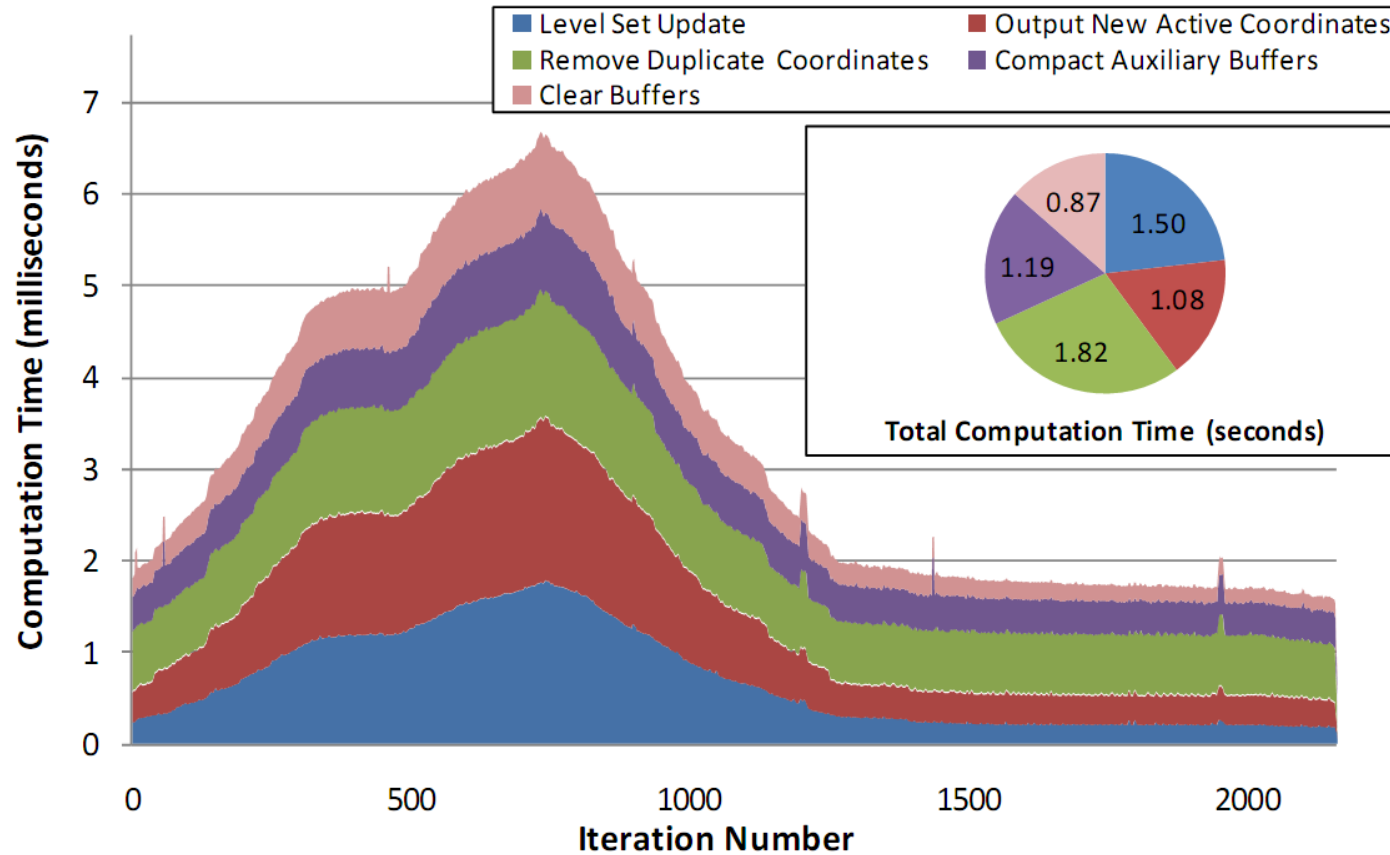


# Linear relationship between computation time and GPU work

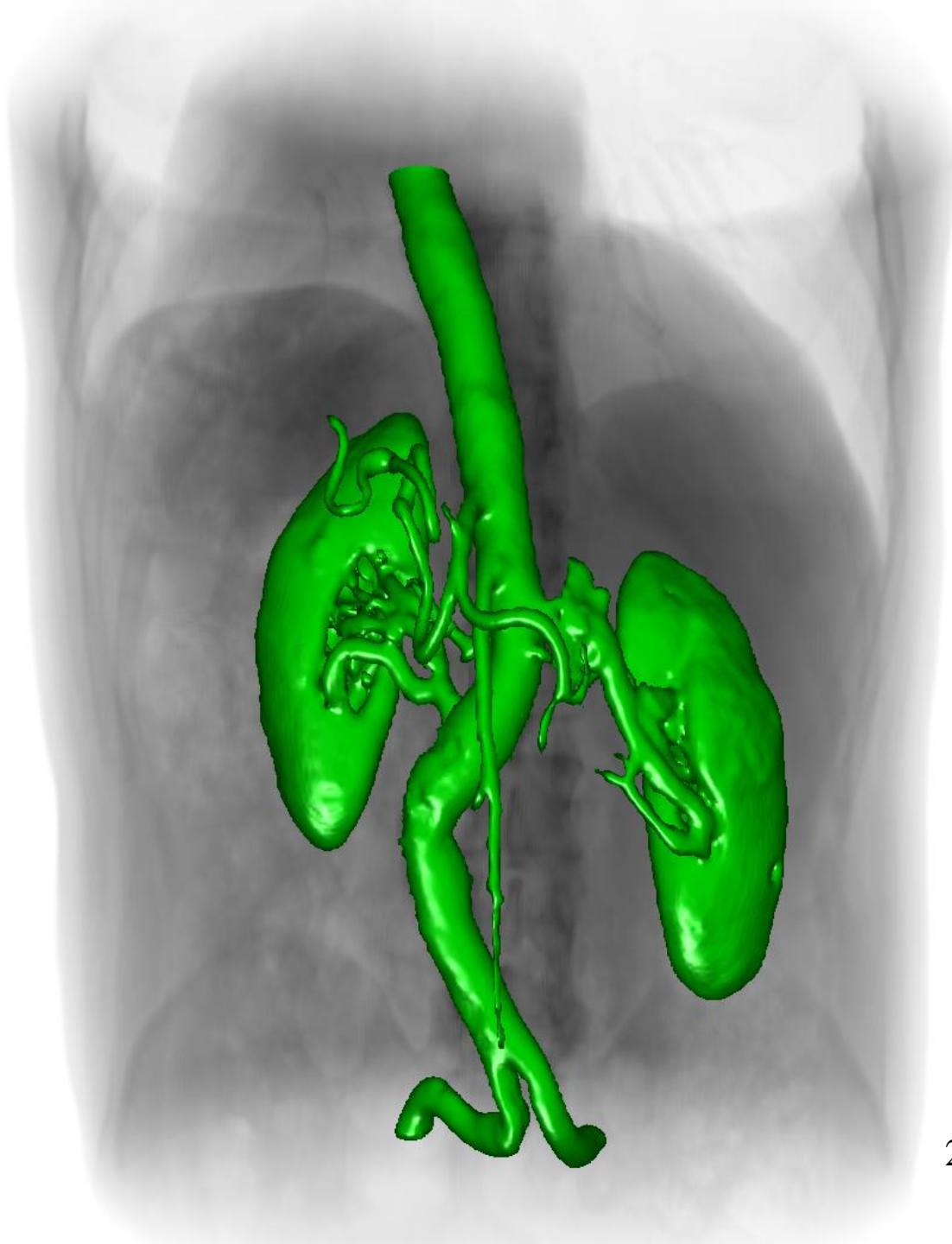




# Most time is spent updating our sparse data structure

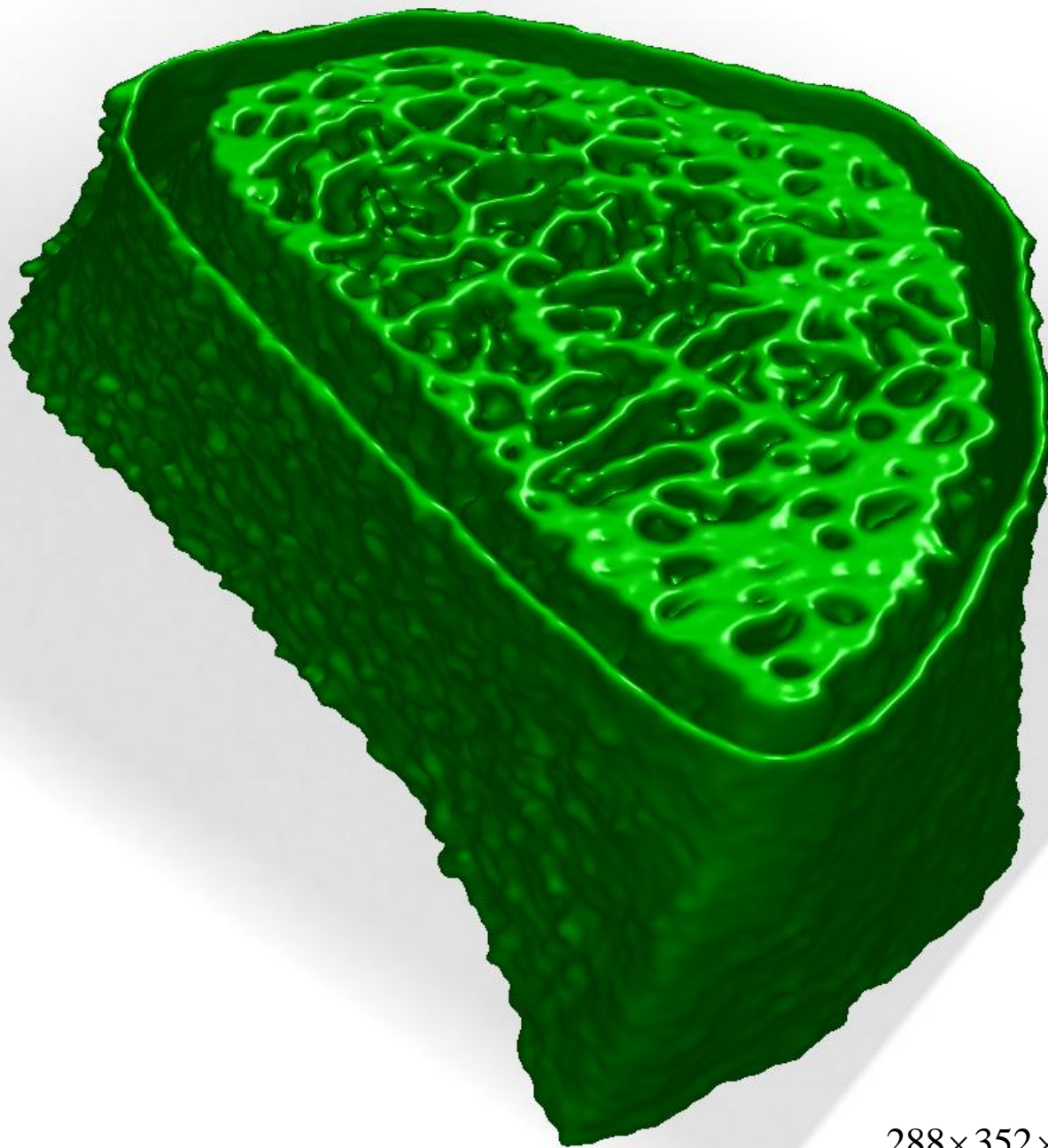


# Results



256×256×272 abdominal CT

# Results



288×352×112 wrist CT

**Goals:** fast, interactive, accurate

 **Goals:** fast, interactive, accurate

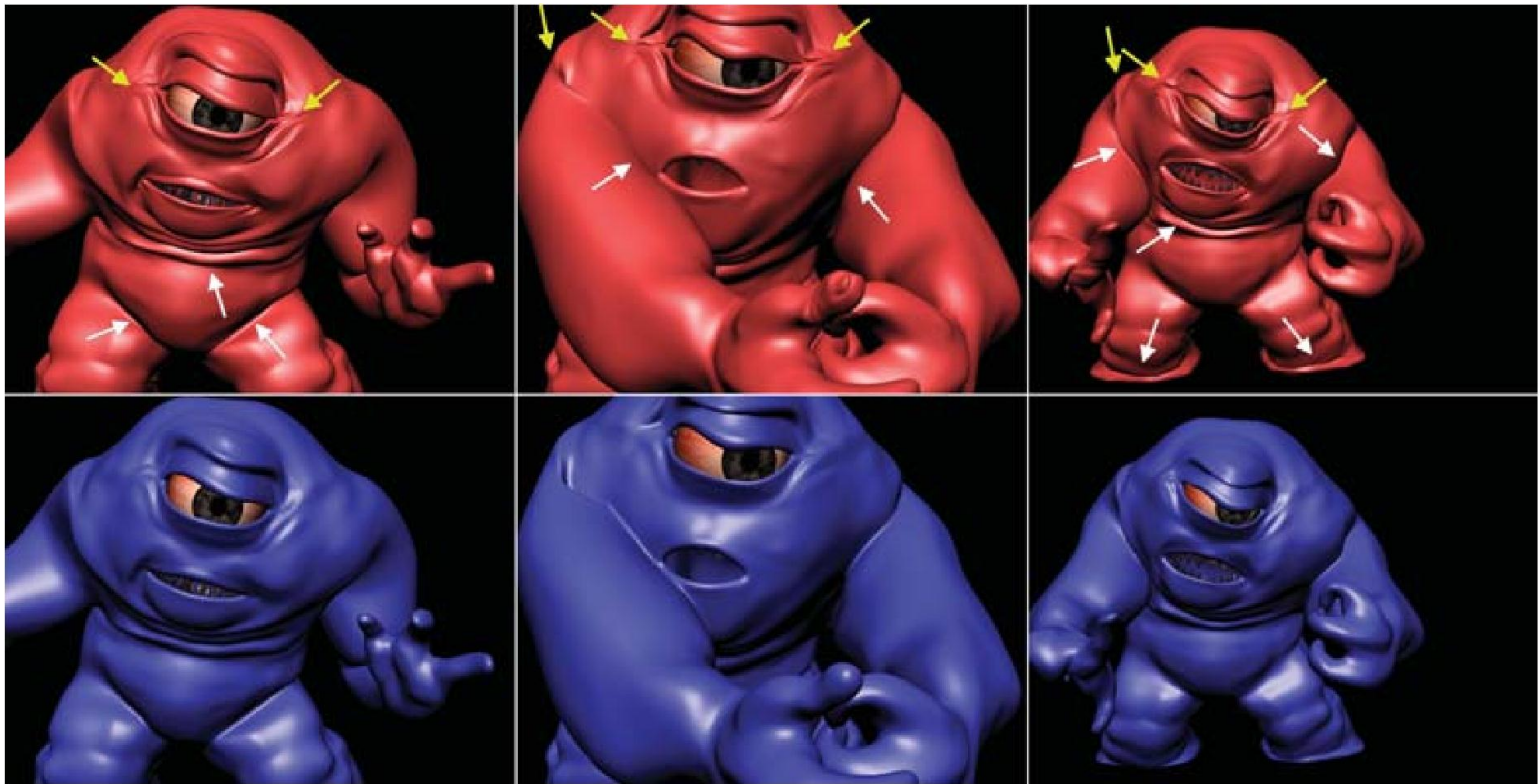
**Goals:**  fast,  interactive, accurate

**Goals:**  fast,  interactive,  accurate

**Goals:**  fast,  interactive,  accurate,  **compact**



*“overall memory consumption proportional to the area of the geometric surface”*



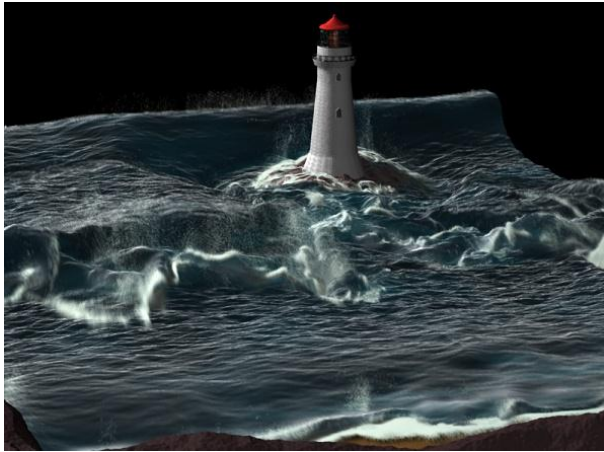
Hierarchical RLE Level Set: A Compact and Versatile Deformable Surface  
Representation  
[Houston et al. 2006]

*“overall memory consumption proportional to the area of the geometric surface”*

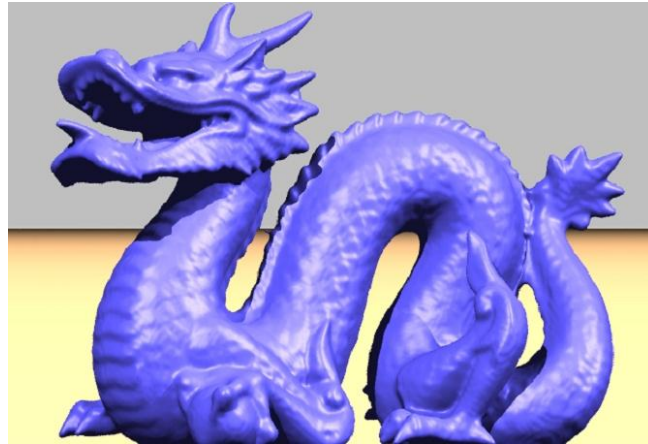
*“fast sequential traversal with  $O(1)$  access time to elements”*



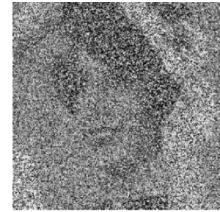
Hierarchical RLE Level Set: A Compact and Versatile Deformable Surface  
Representation  
[Houston et al. 2006]



[Losasso et al. 2008]



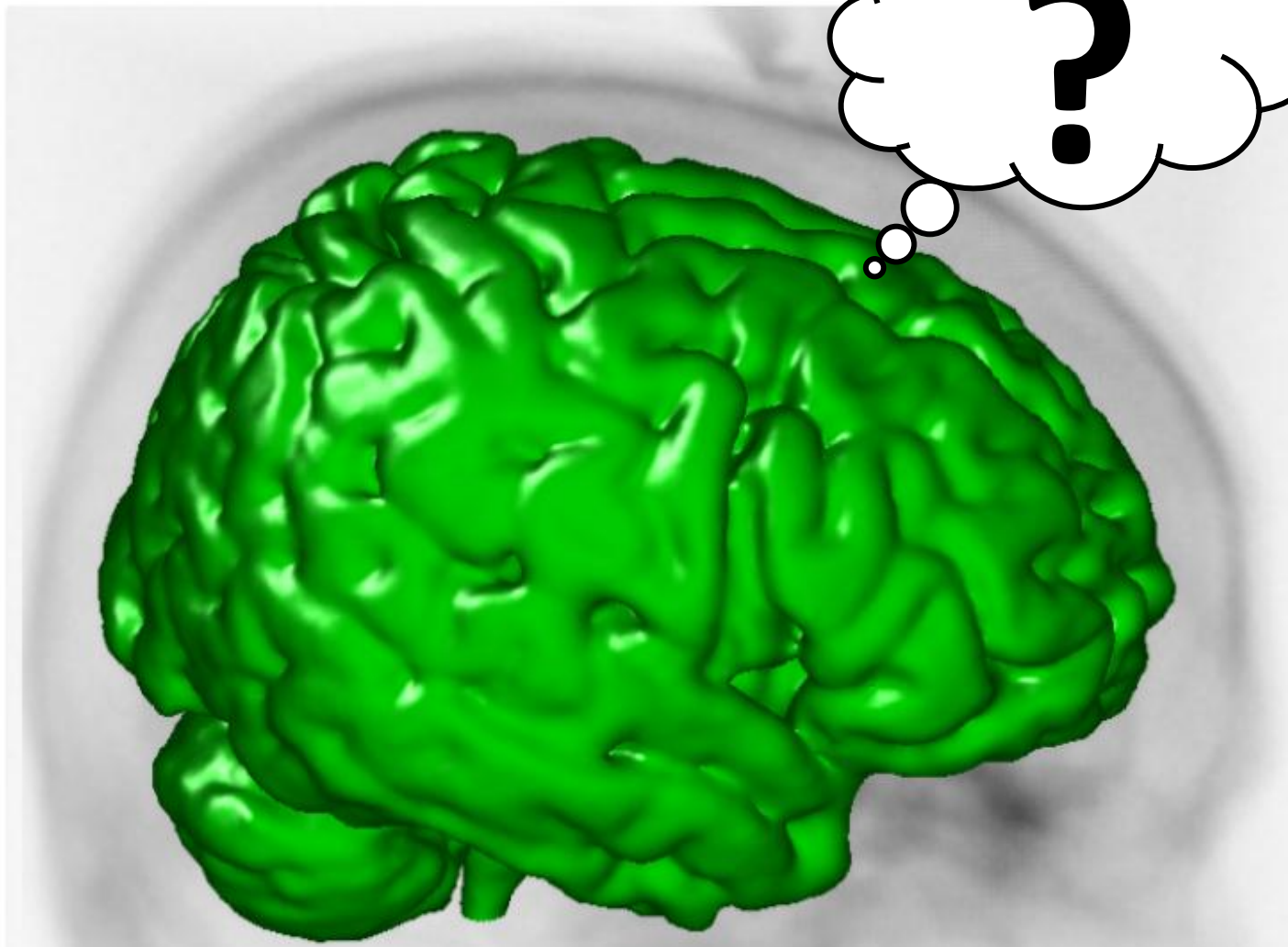
[Zhao et al. 2001]



[Darbon and Sigelle 2006]

?





## **A Work-Efficient GPU Algorithm for Level Set Segmentation**

Mike Roberts, Jeff Packer, Mario Costa Sousa, Joseph Ross Mitchell

*High Performance Graphics 2010*





# Bonus Slides

What do I mean by work-efficient?



What do I mean by work-efficient?

$$\mathbf{O}(\text{parallel work}) = \mathbf{O}(\text{sequential work})$$