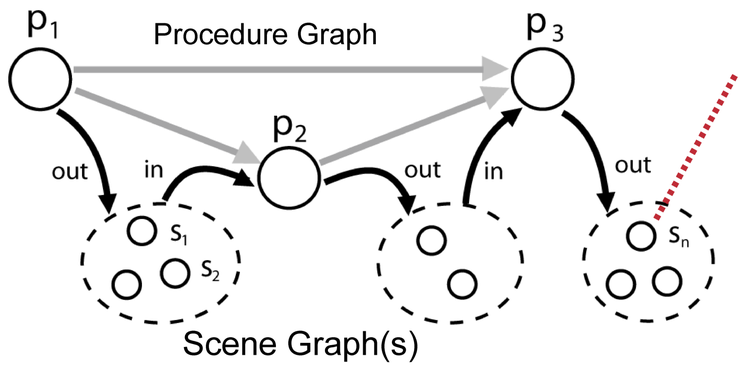
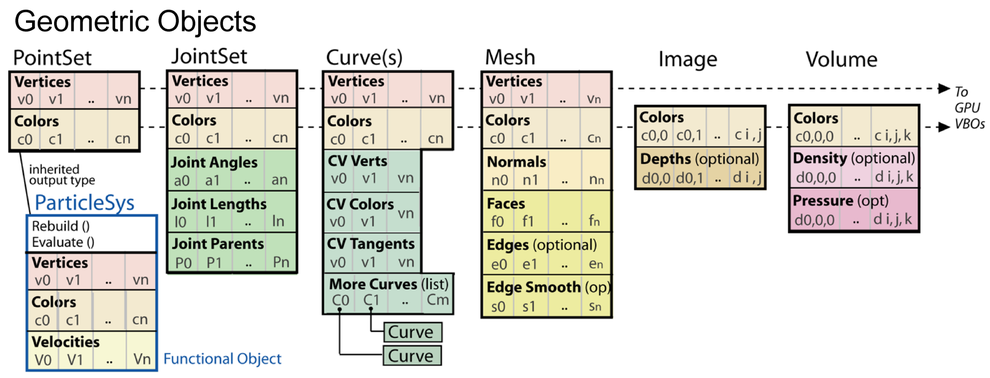
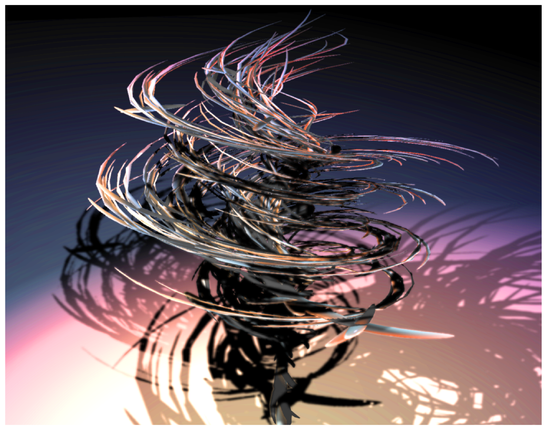
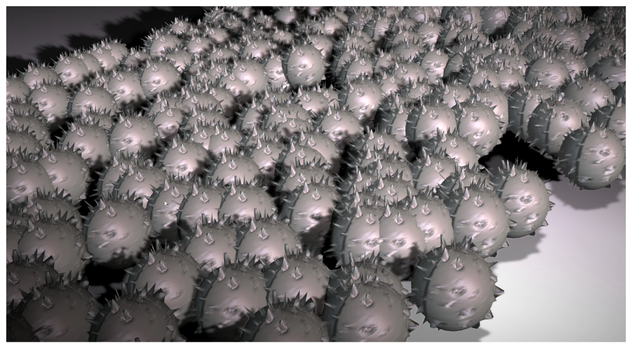
Efficient Procedural Modeling of Dynamic Systems

Rama C. Hoetzlein
Department of Architecture and Media Studies, Aalborg University
www.rchoetzlein.com

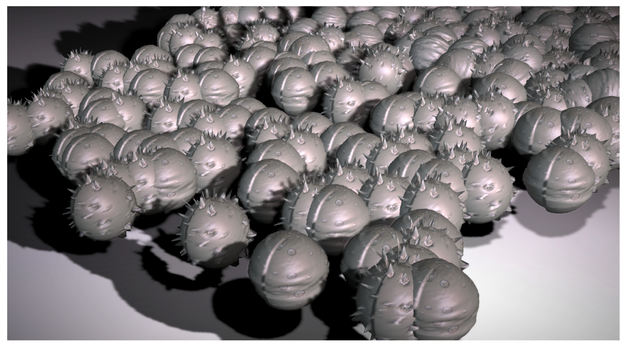




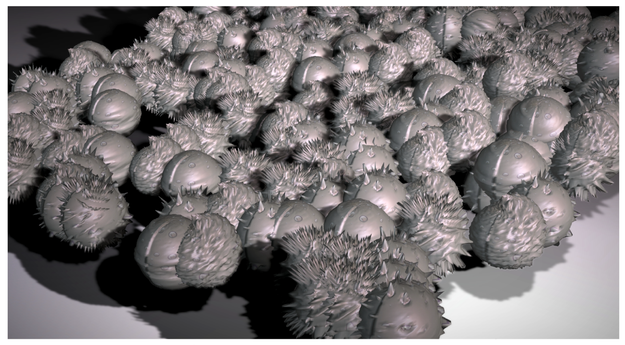




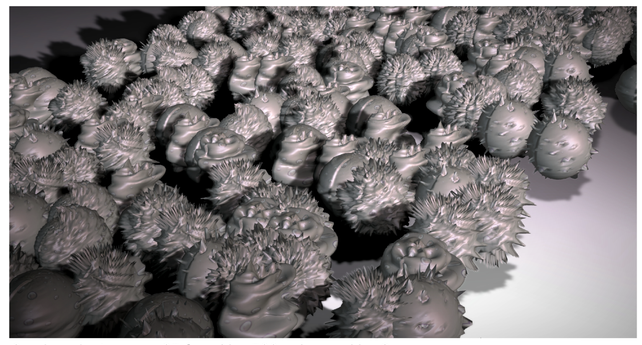
1 unique instance, 25 fps. N = 500 spheres, U = 1.
Identical to GPU-based hardware instancing.



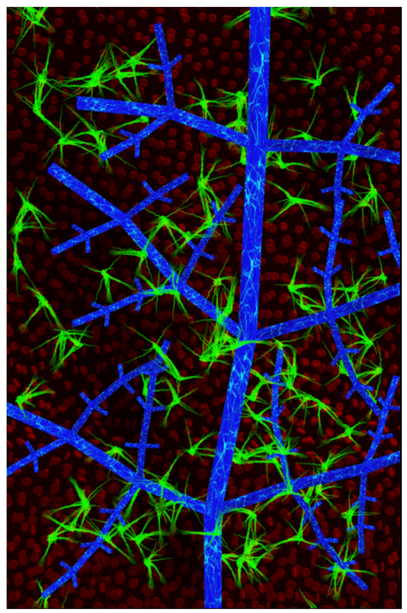
2 unique instances, 25 fps. N = 500 spheres, U = 2.
No change in frame rate, sub-graph evaluation occurs on GPU.



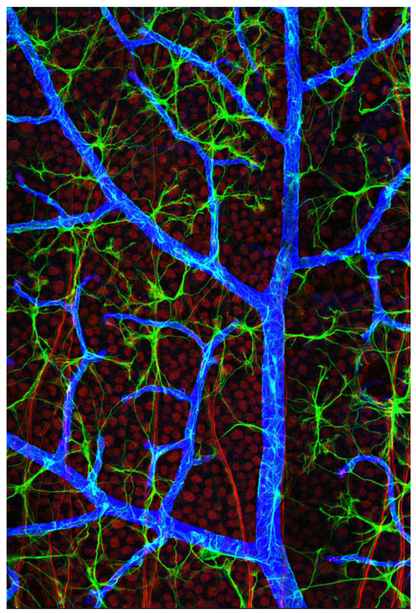
4 unique instances, 25 fps. N = 500 spheres, U = 4.
Scene appears considerably more varied. Rendering in parallel with evaluation.



8 unique instances, 25 fps. N = 500 spheres, U = 8.
Consistent frame rate up with up to 8 varieties. Depends on object complexity.



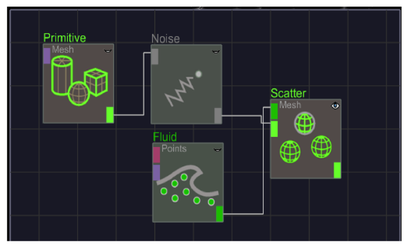
Synthetic Rendering, generated in real-time using LUNA



Original microscopic image

Synthetic Rendering: Modeling of organic structures to mimic biological imagery, used for ground truth testing of cell image segmentation methods. Blood vessels (blue), modeled as trees. Astrocytes (green), as spring systems which reach out to vessels. Image shows rabbit retinal astrocytes.

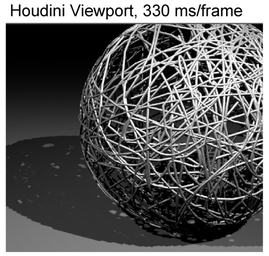
Original data courtesy Neuroscience Research Institute, University of California Santa Babara Gabe Luna, Geoffrey Lewis, and Steve Fisher (c) 2010.



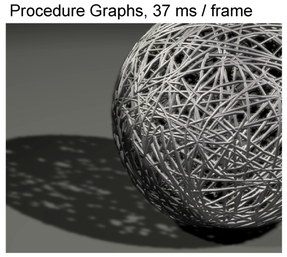
Adaptive Instancing: Sub-graphs represent the generative rules for individual objects (e.g. noisy sphere). The set of total objects N is partitioned into U unique instances, which are automatically generated by re-evaluating the subgraph. GPU buffers efficiently update these partial sets.

Performance: Performance is tested against a woven sphere reference model. Results are significantly faster than Houdini, and faster than 2x the baseline tests done directly in OpenGL/GLUT.

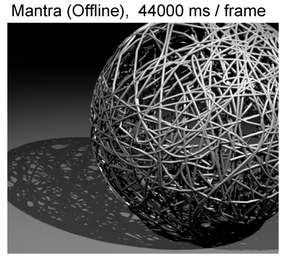
Houdini Viewport, 330 ms/frame



Procedure Graphs, 37 ms / frame



Mantra (Offline), 44000 ms / frame



$T_R N = U (T_E + T_O)$

GPU CPU

T_R = render time (0.078 ms / obj)
T_E = eval time (4.7 ms)
T_O = overhead (0.4 ms)