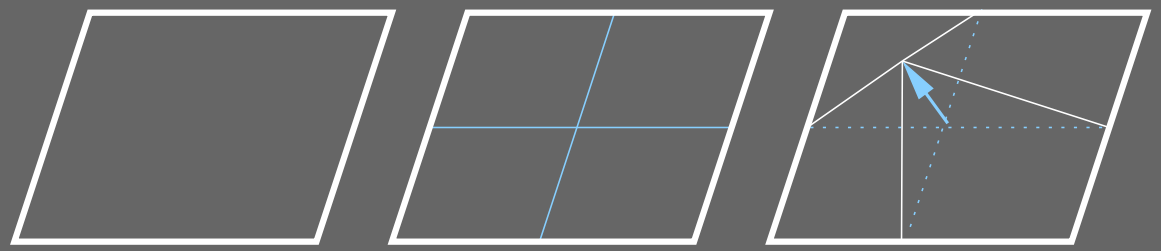
Mesh Mutation in Programmable Graphics Hardware

Graphics Hardware (2003)

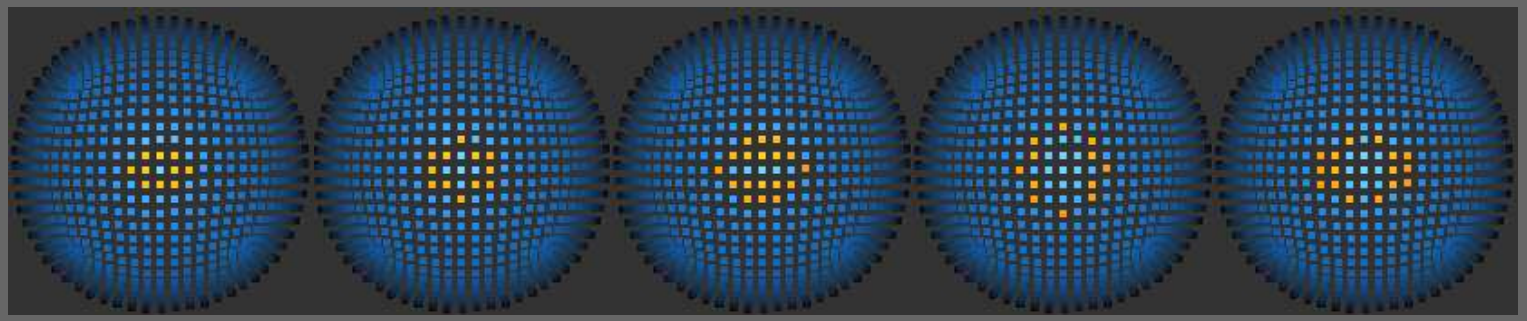
Le-Jeng Shiue, University of Florida
Vineet Goel, ATI
Jörg Peters, University of Florida

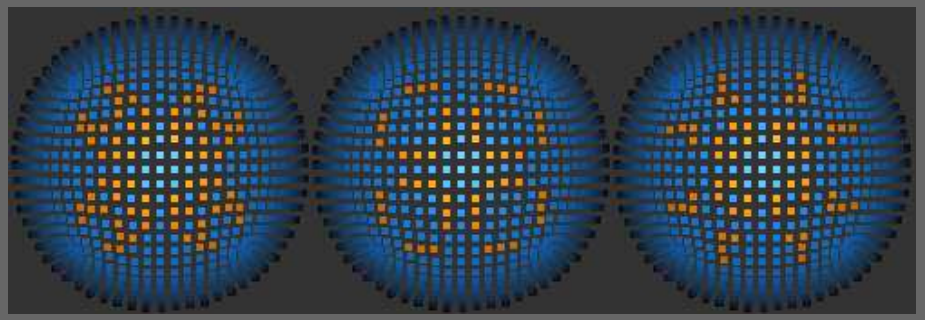
Mesh Mutation



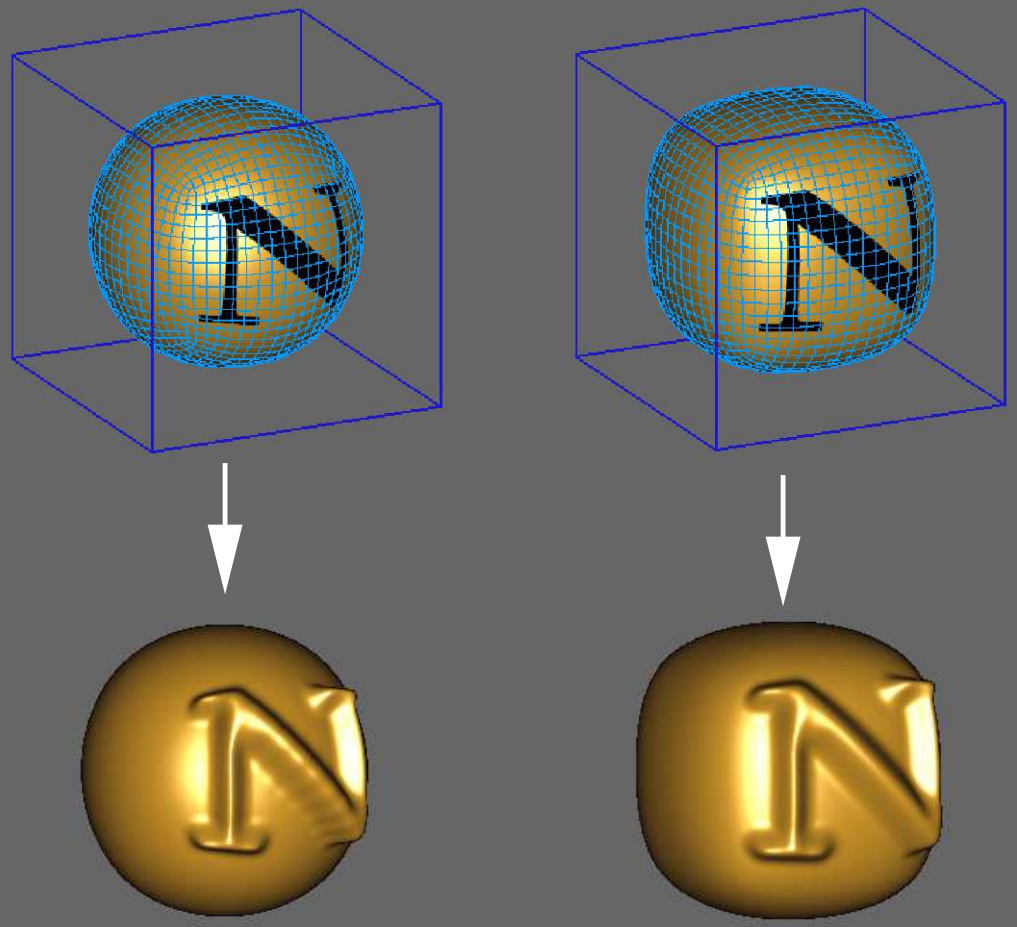
Refinement Adjustment

Mesh Mutation: No Refinement

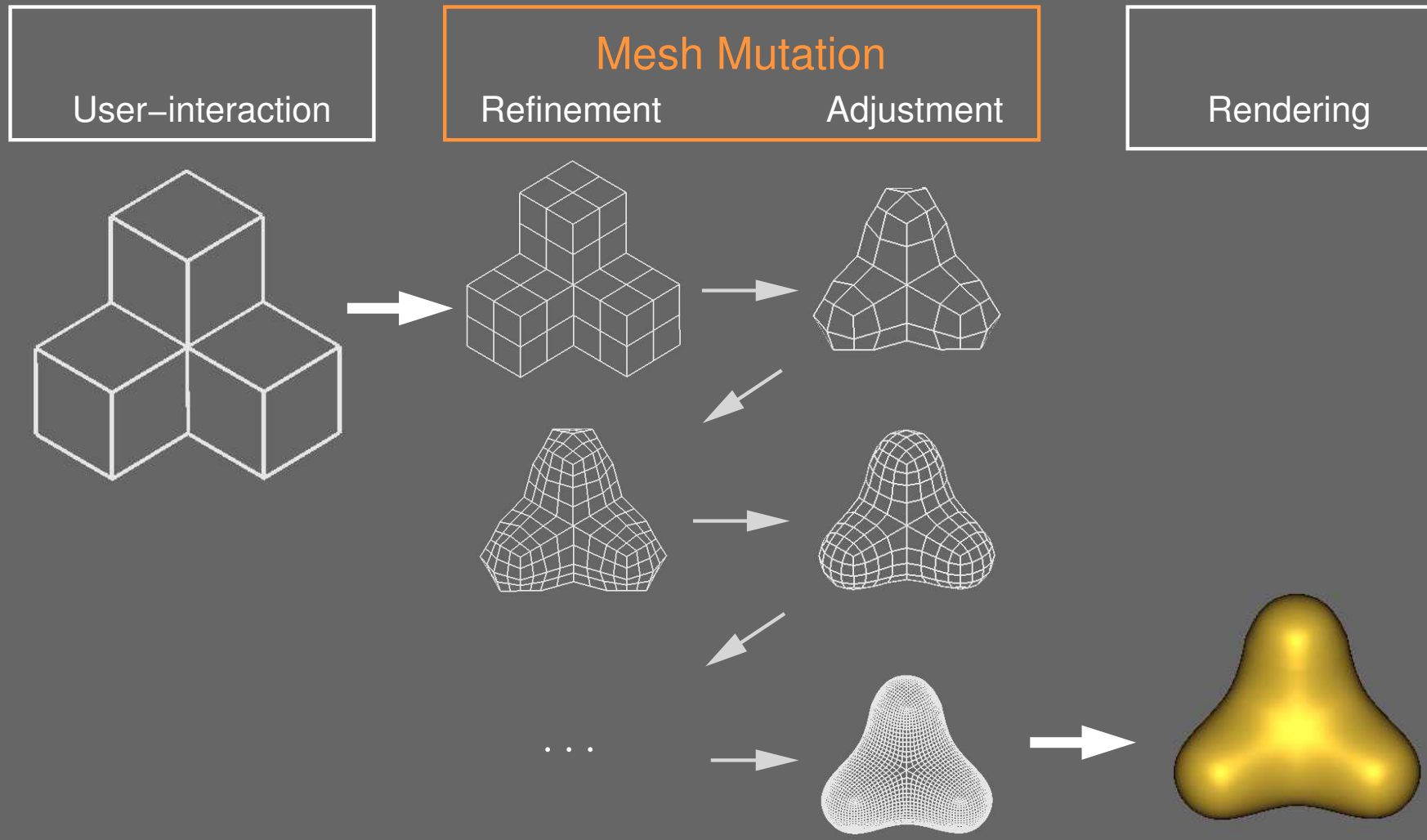




Mesh Mutation: Subdivision + Displacement Map



Mesh Modeling



Mesh Modeling

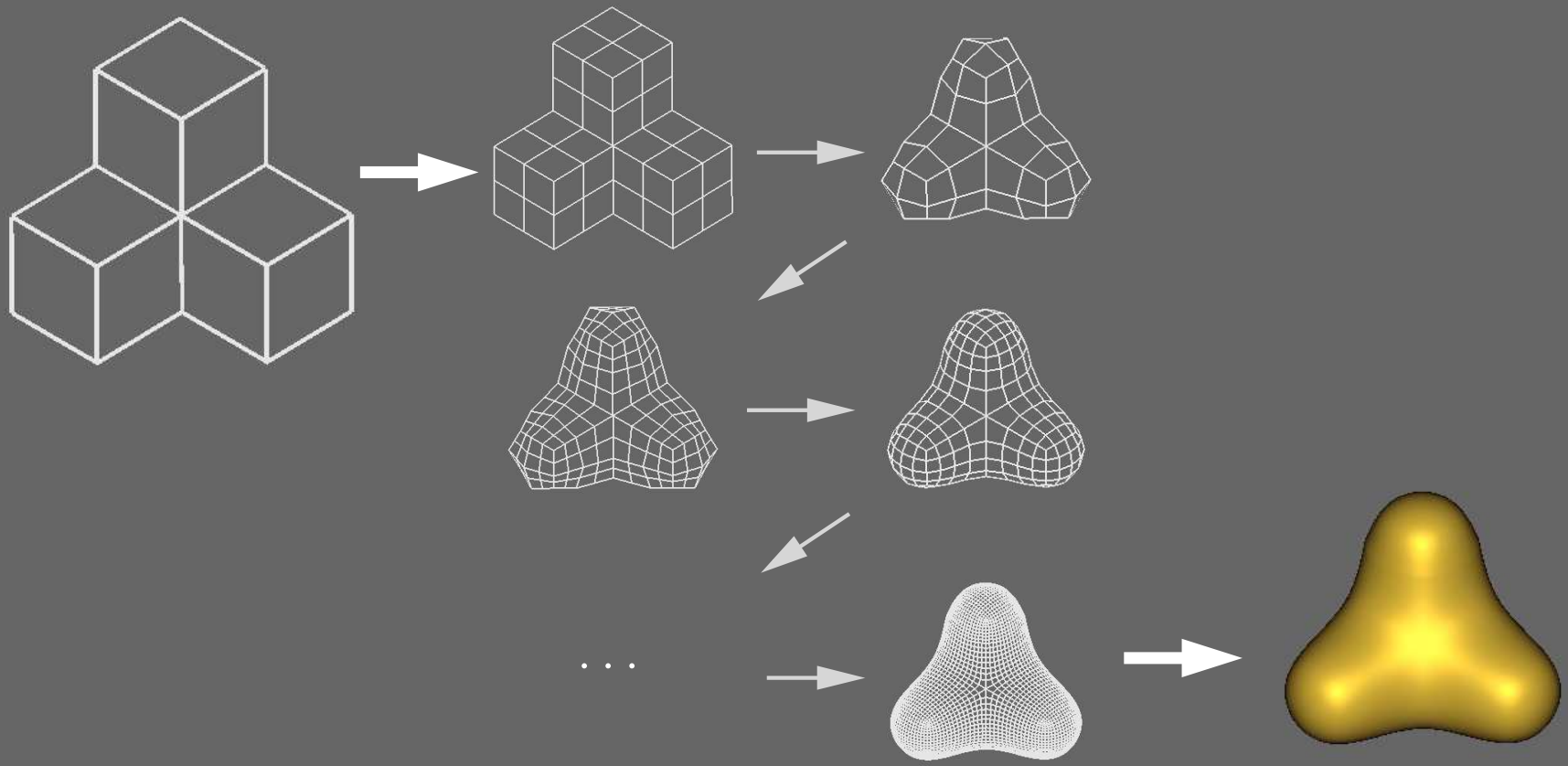
CPU

Mesh Mutation

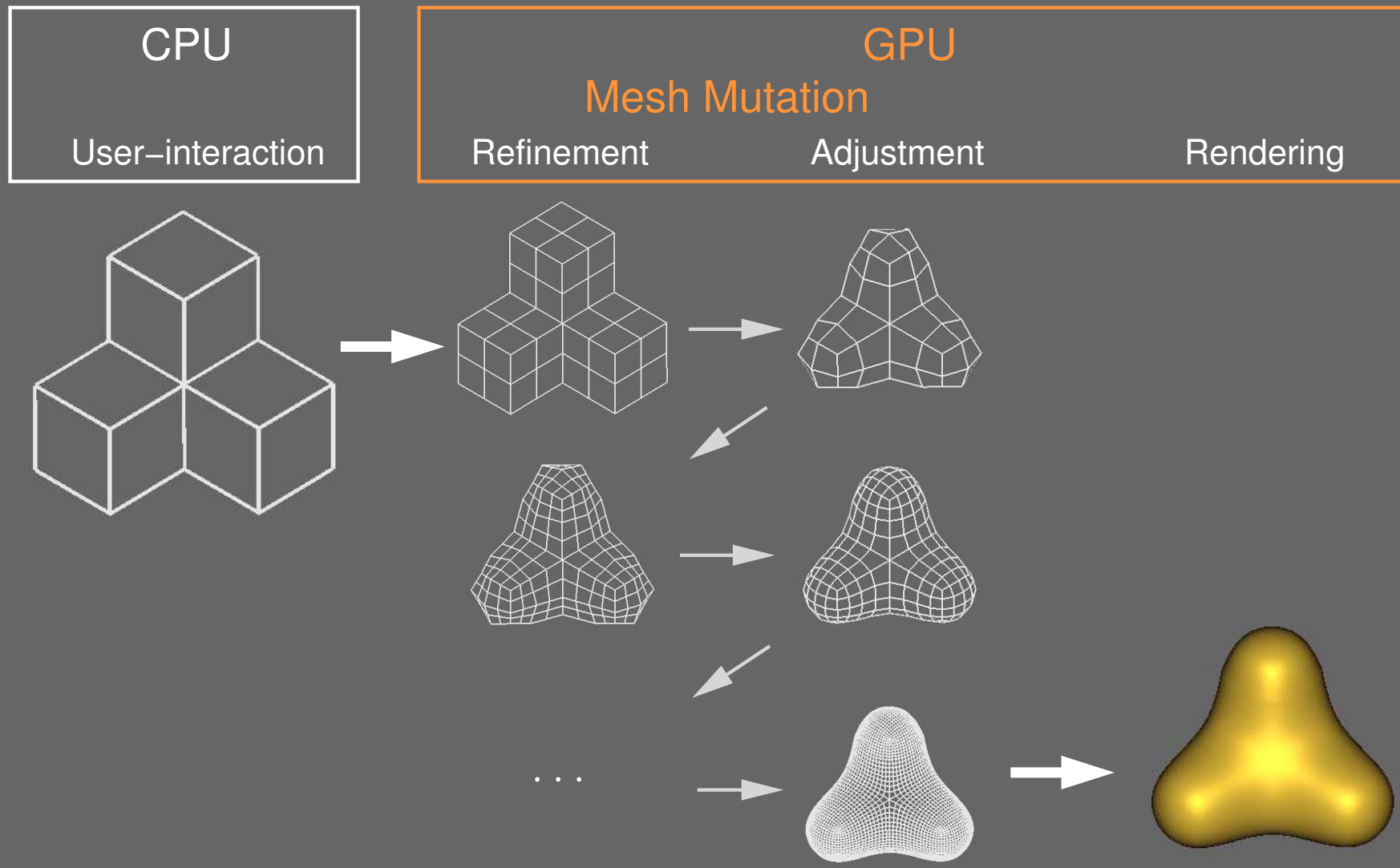
User–interaction | Refinement | Adjustment

GPU

Rendering



Mesh Modeling



Mesh in Graphics Hardware: Goals & Tools

- Reduce CPU→GPU traffic (data and rendering instructions)
- Use parallel computing on the GPU
- Improve the rendering quality for meshes
- Increase the level of abstraction for the programmer

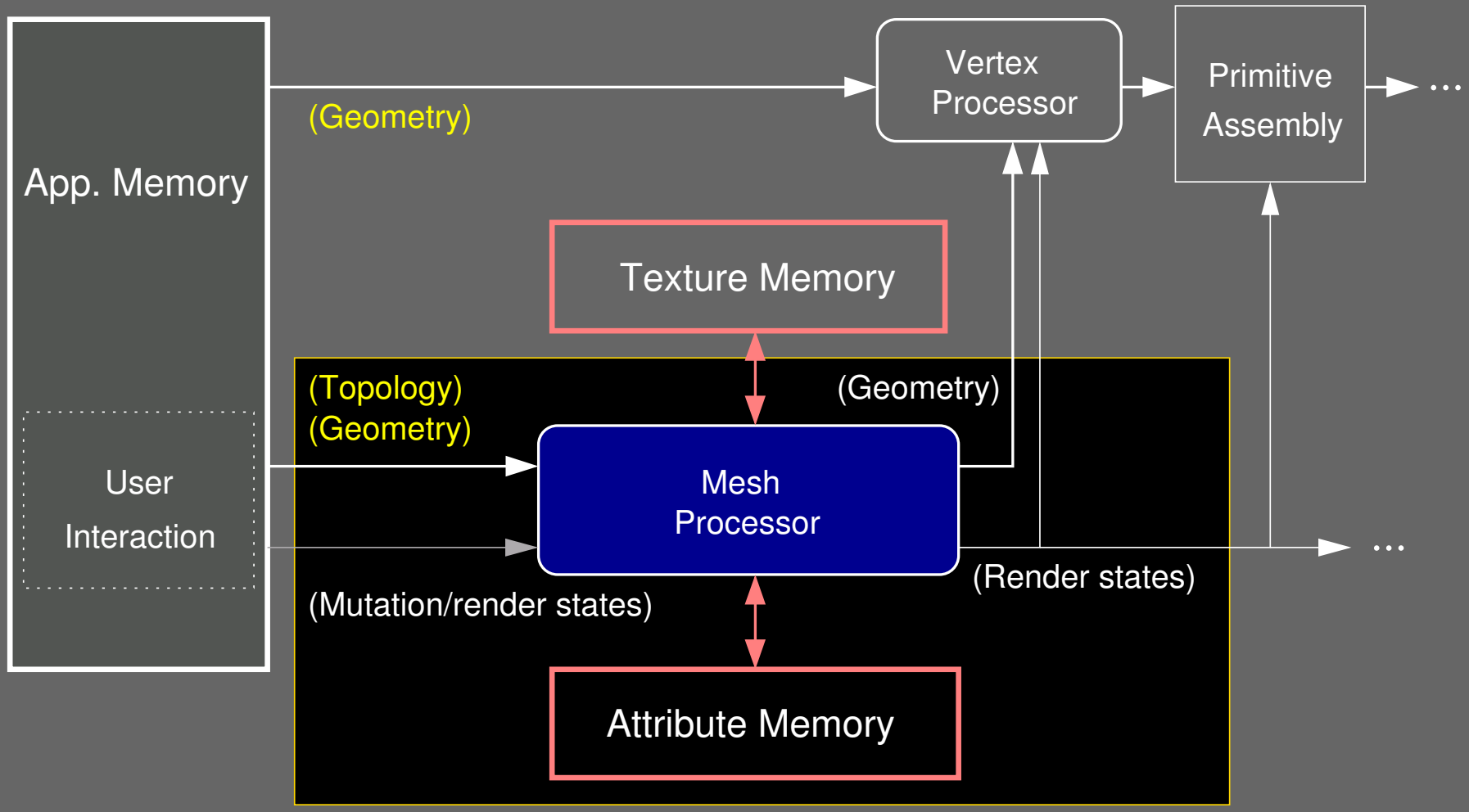
- Separate connectivity and attributes (structure and operations)
- Avoid adjacency pointers where possible
- Maximize algorithmic parallelism

Mesh in Graphics Hardware: Goals & Tools

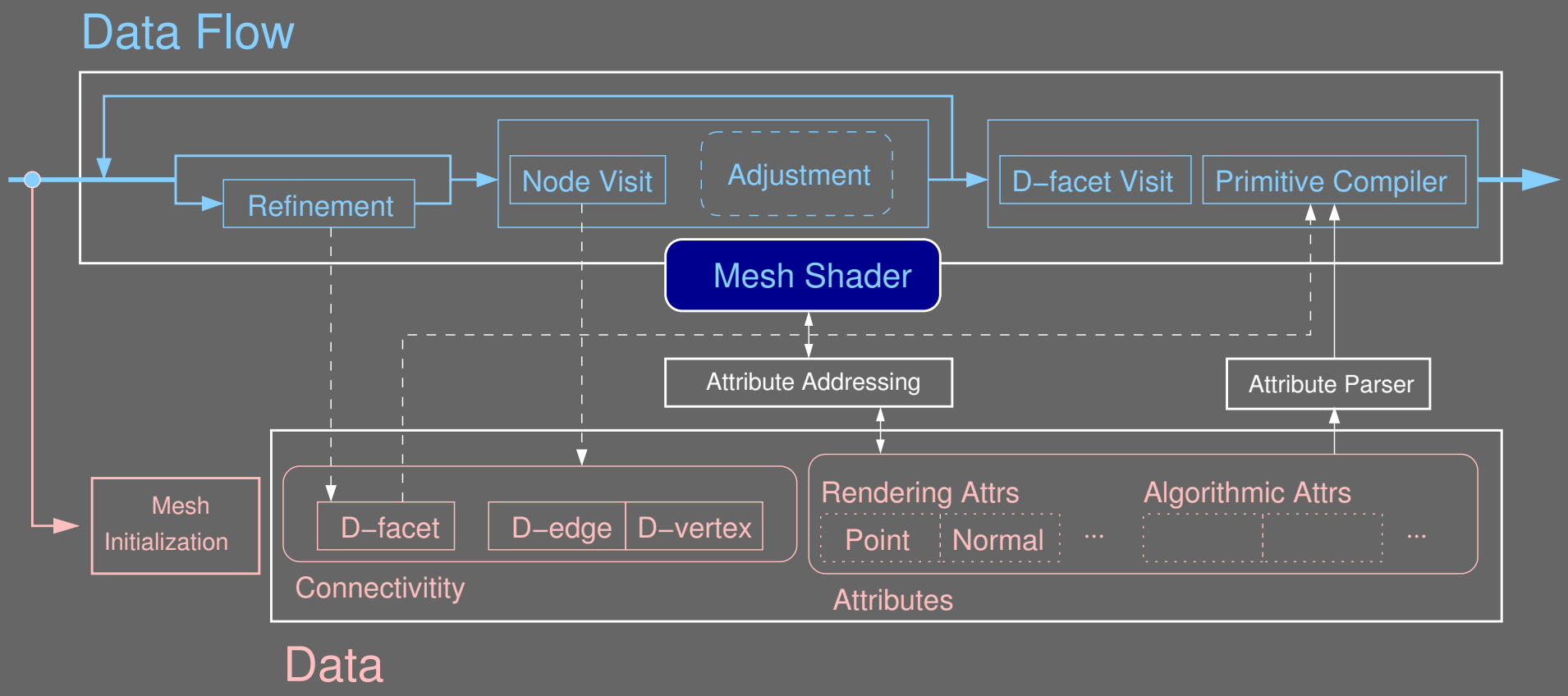
- Reduce CPU→GPU traffic (data and rendering instructions)
- Use parallel computing on the GPU
- Improve the rendering quality for meshes
- Increase the level of abstraction for the programmer

- Separate connectivity and attributes (structure and operations)
- Avoid adjacency pointers where possible
- Maximize algorithmic parallelism

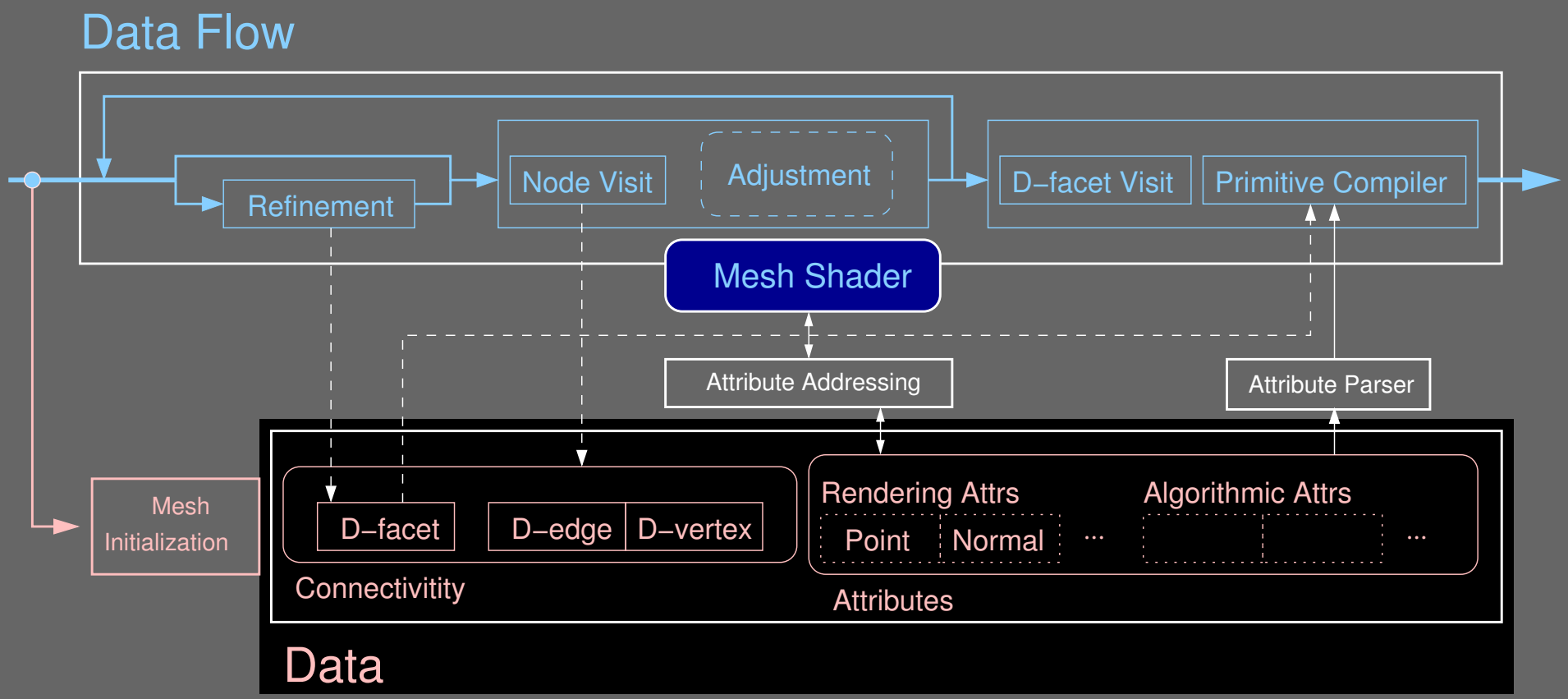
Mesh Processor



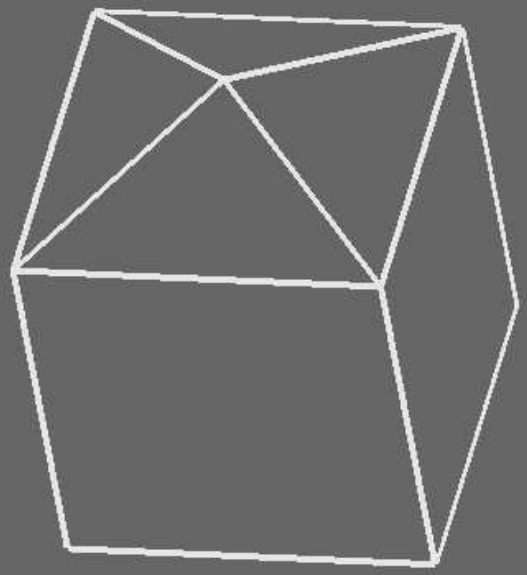
Mesh Processor



Mesh and Refinement Structure

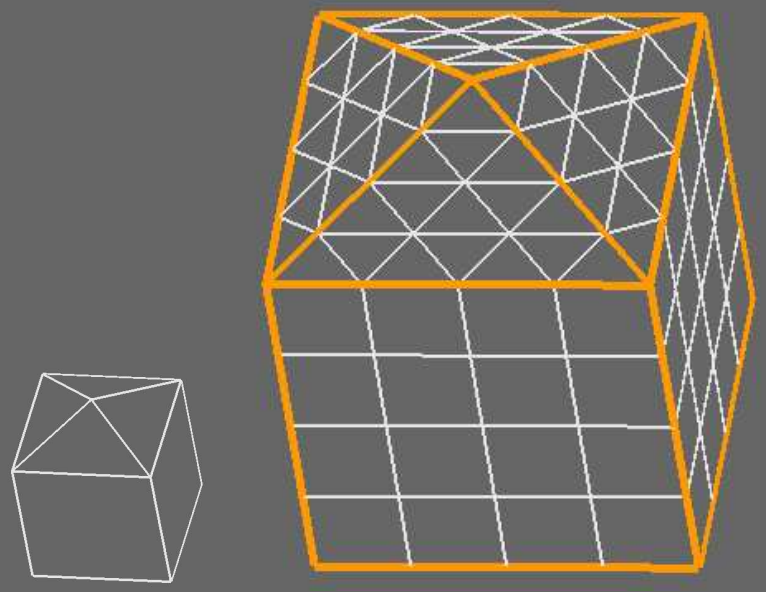


Atlas, Charts and Attributes

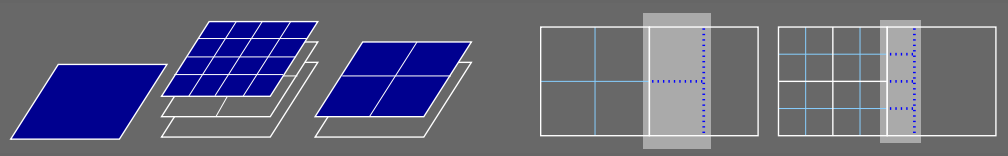


- Static domain connectivity
- Index-based halfedge structure: no adjacency pointers
- Pure connectivity structure, no attributes associated

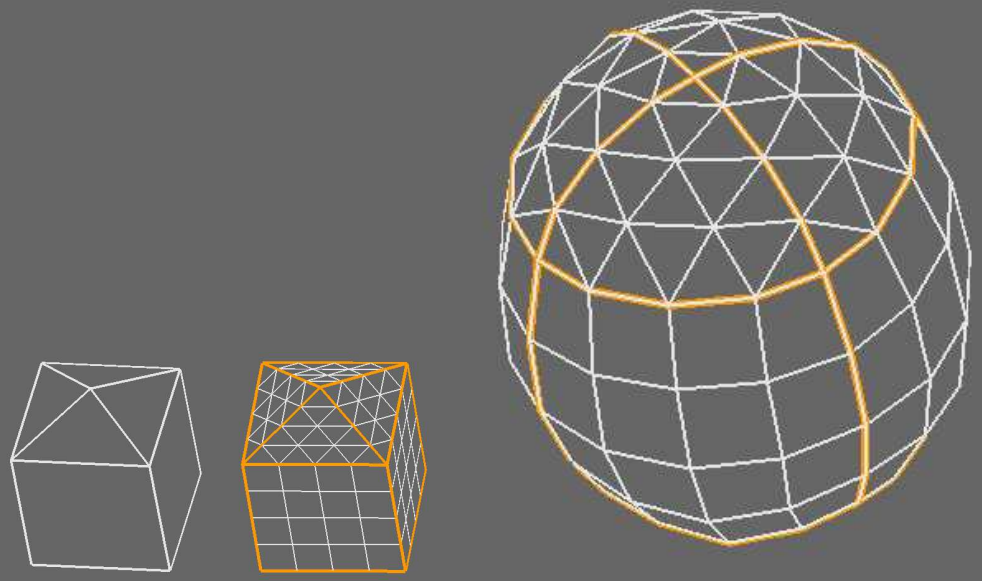
Atlas, **Charts** and Attributes



- Dynamic and regular refinement connectivity
- Index enumeration: not a real structure
- Chart-based adaptive refinement using the auxiliary layer

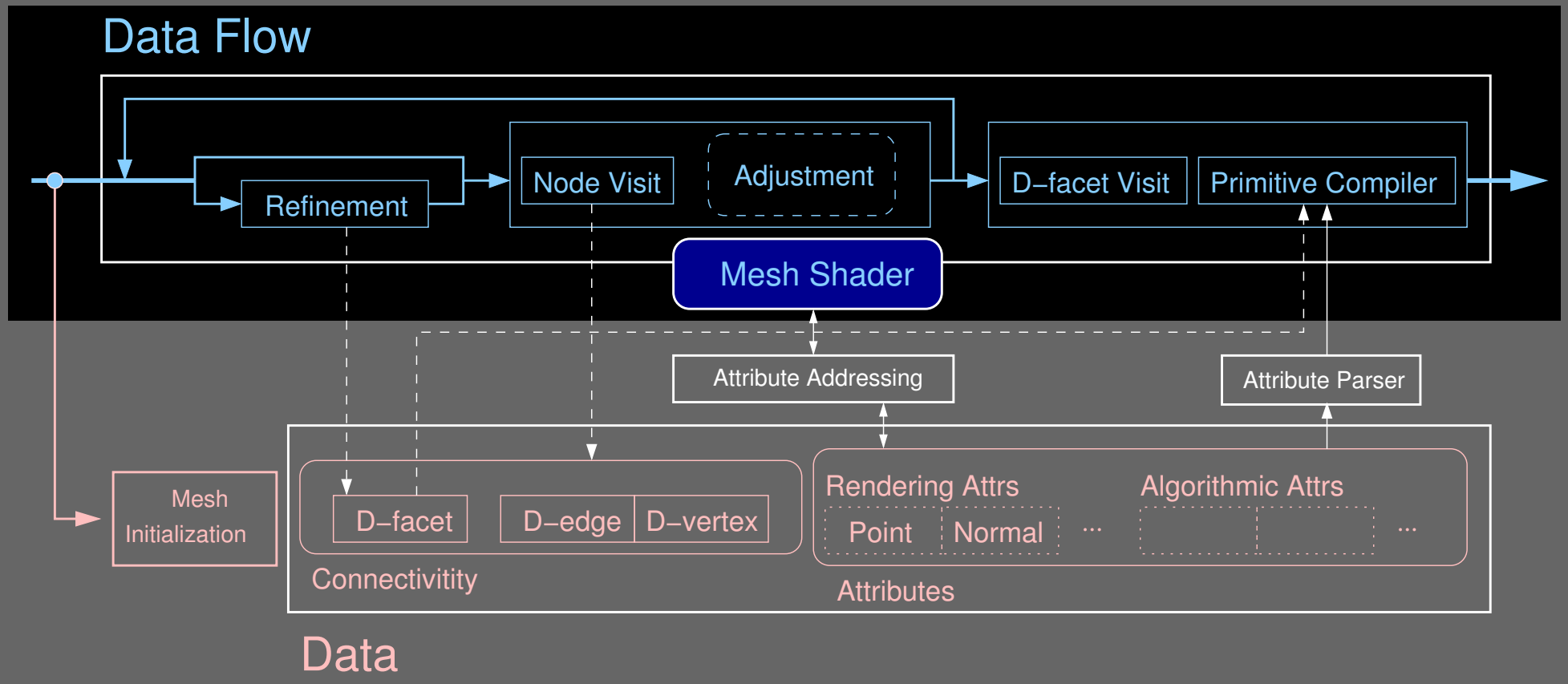


Atlas, Charts and **Attributes**

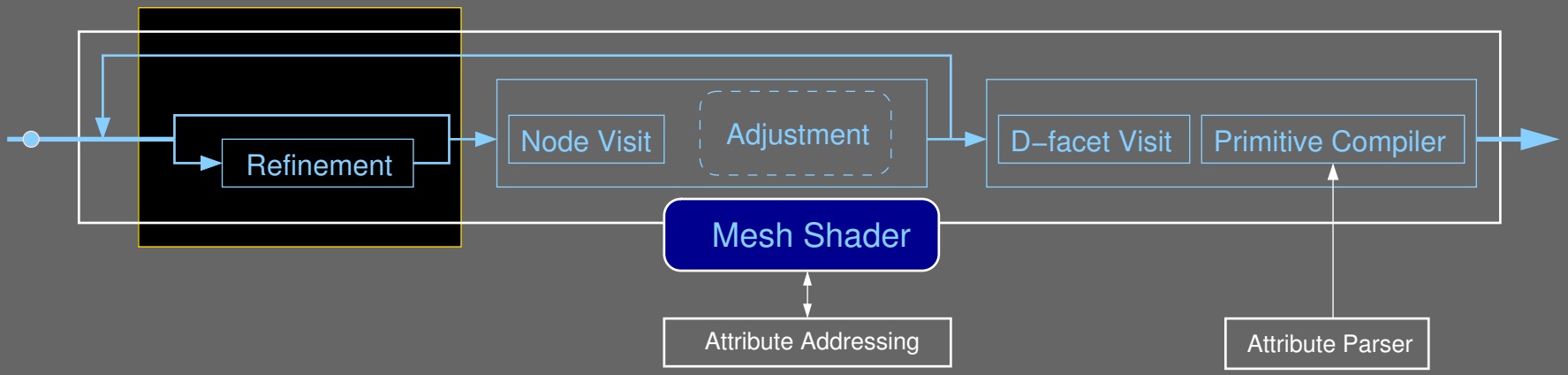


- Generic: instantiated with memory base and attribute size
- User-definable: attribute types and addressing modes
- Computed access

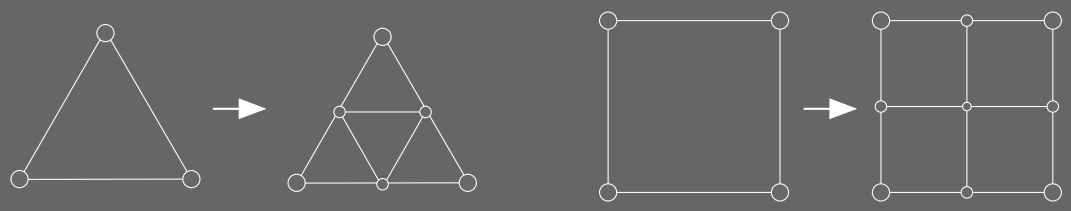
Mesh in Programmable Graphics Hardware (MiPGH)



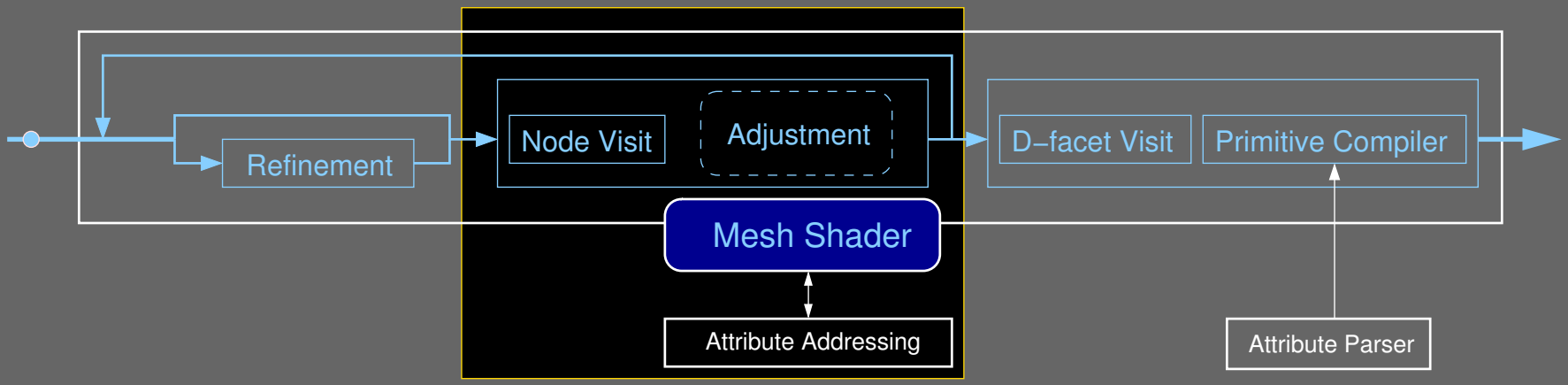
MiPGH: Refinement



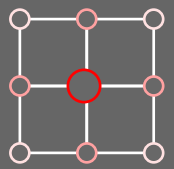
- User-transparent
- Support quadrisection on quad or triangle polygons



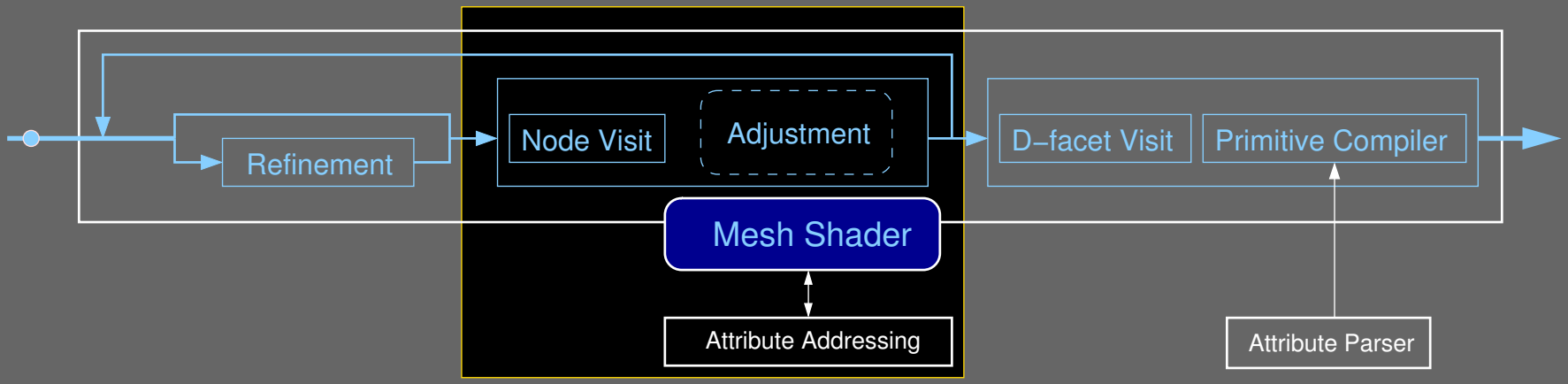
MiPGH: Adjustment



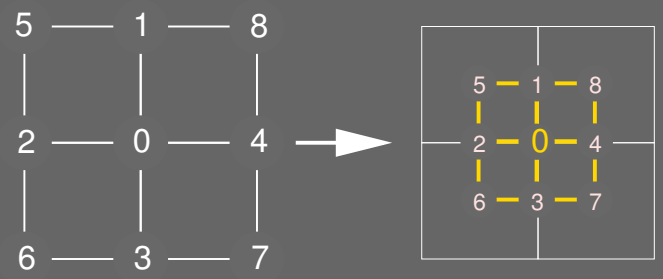
- User-transparent global mesh traversal
- User-programmable mesh shader (adjustment rule) on user-defined attributes
- 1-disk local access in the mesh shader



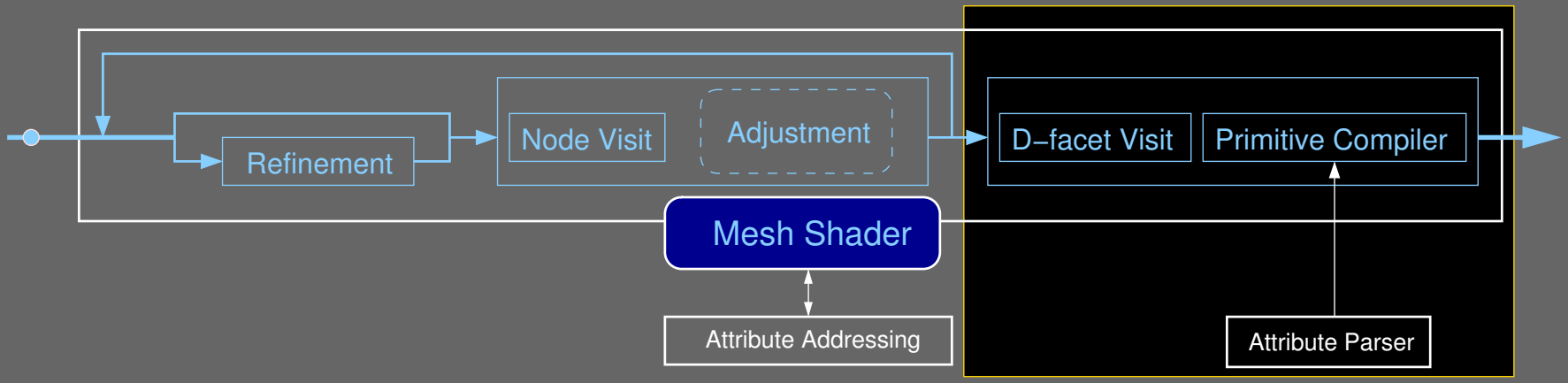
MiPGH: Mesh Shader




```
...
const float mask[9] = {36/64.0,6/64.0,6/64.0,6/64.0,6/64.0,1/64.0,1/64.0,1/64.0,1/64.0};
readAttribute4(IPositionHandle, IVertexIndices, mesh, 9);
Position = inner_product(mesh, mask, 9);
...
```

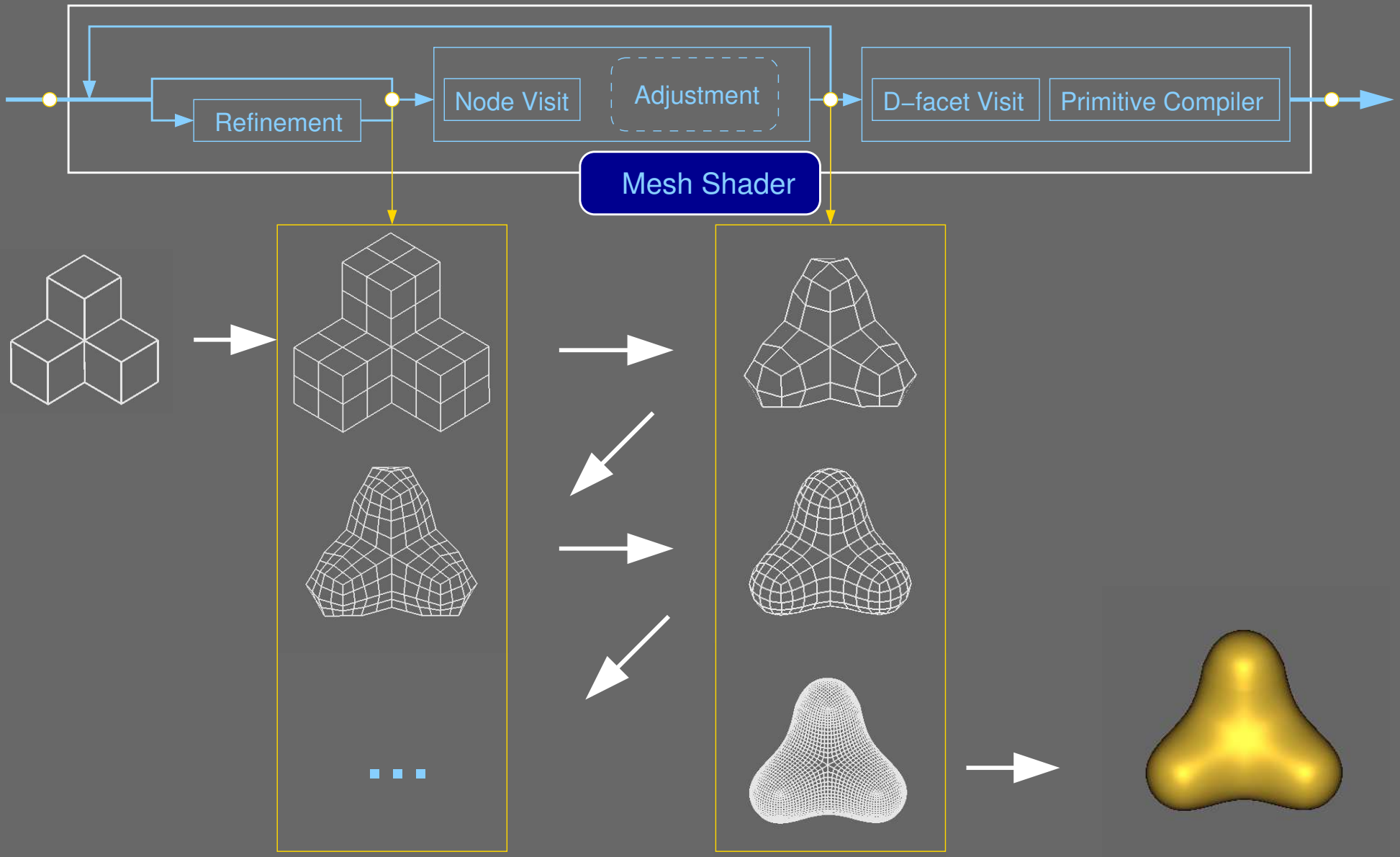


MiPGH: Rendering

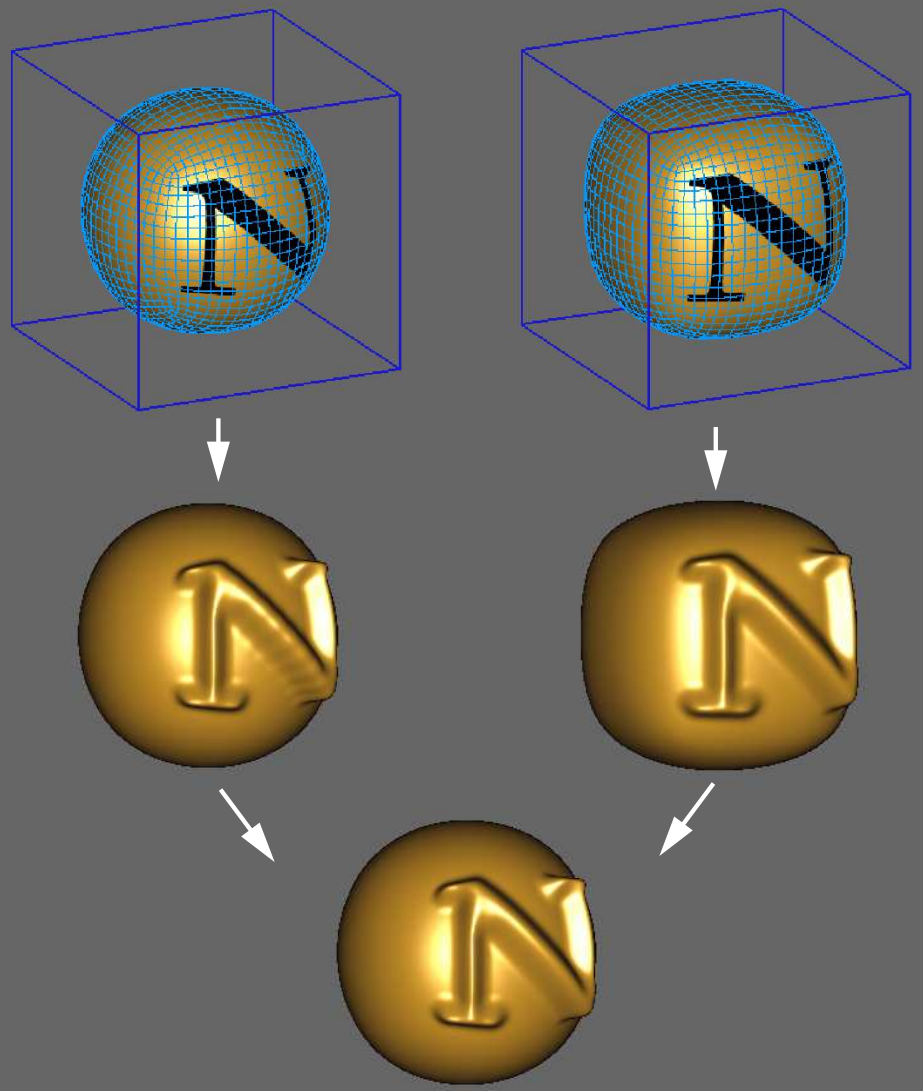


- User-defined rendering attributes
- Chart is the rendering primitive

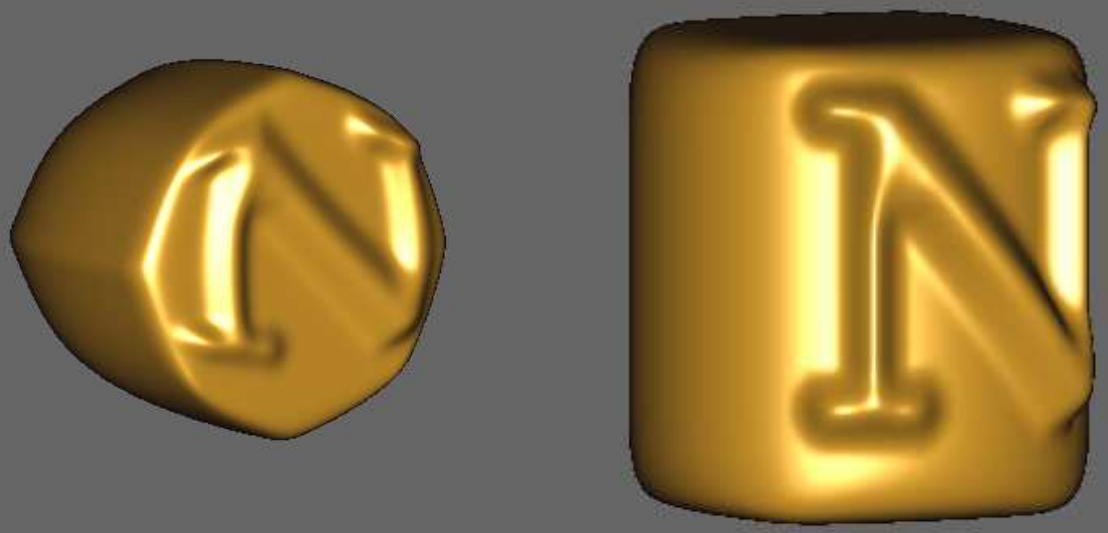
MiPGH


Refinement
Node Visit
Adjustment
D–facet Visit
Primitive Compiler
Mesh Shader

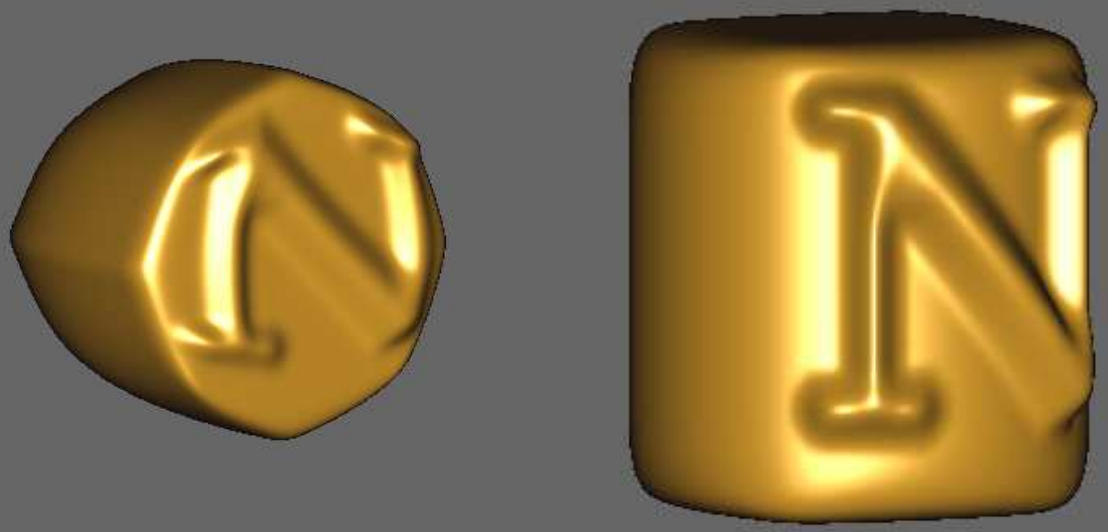
Subdivision + Displacement Map



Extended Attributes: Crease Value



Extented Attributes: Crease Value



Thank You!