FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Diplomarbeitspräsentation

Micropolygon Rendering on the GPU

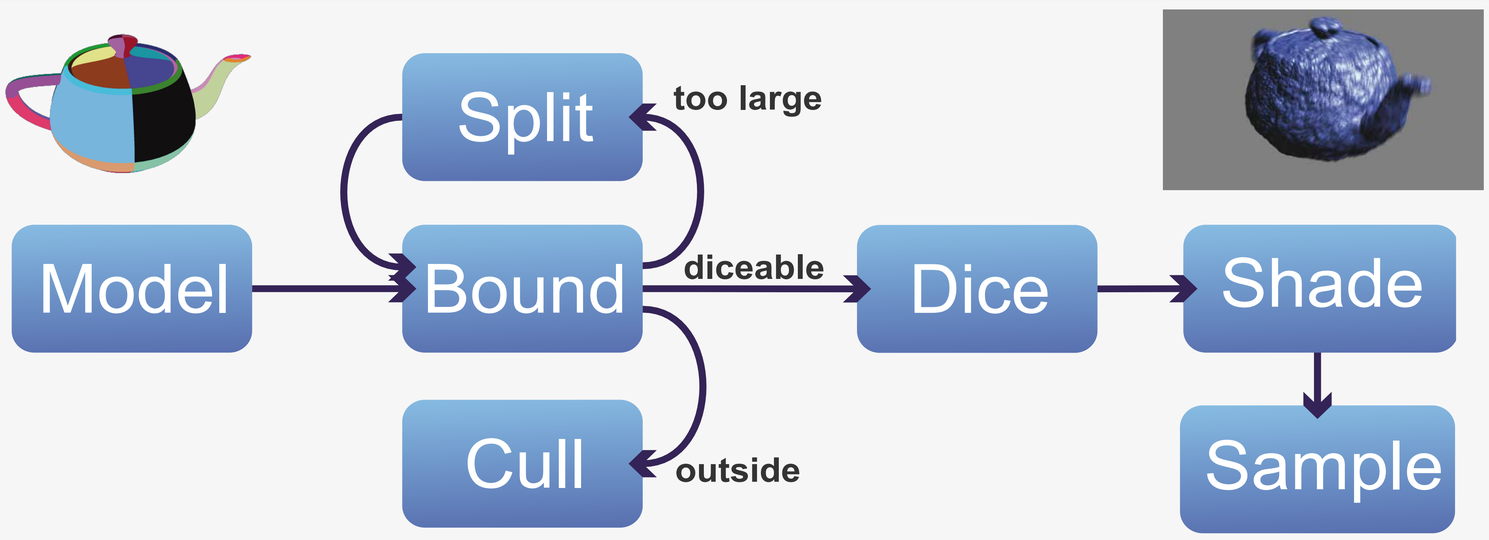
Masterstudium:
Visual Computing

Thomas Weber

Technische Universität Wien
Institut für Computergraphik und Algorithmen
Arbeitsbereich: Computergraphik
Betreuer: Prof. Michael Wimmer
Mitwirkung: Prof. John D. Owens, UC Davis

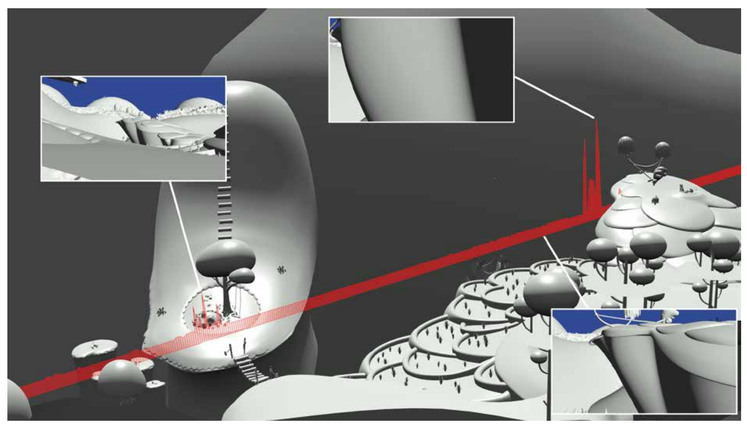
Problem Statement/Motivation

Reyes is a powerful rendering algorithm that works by tessellating surfaces into polygons that are smaller than the screen's pixels (so-called micropolygons). Surfaces are first recursively bound and split until the result is smaller than a predefined screen-bound. After this, the surface is evaluated into a grid of polygons, which are then shaded and rasterized.
This allows artifact-free rendering of curved surfaces with optional displacement. These quality advantages of Reyes makes it desirable to implement it on the GPU.



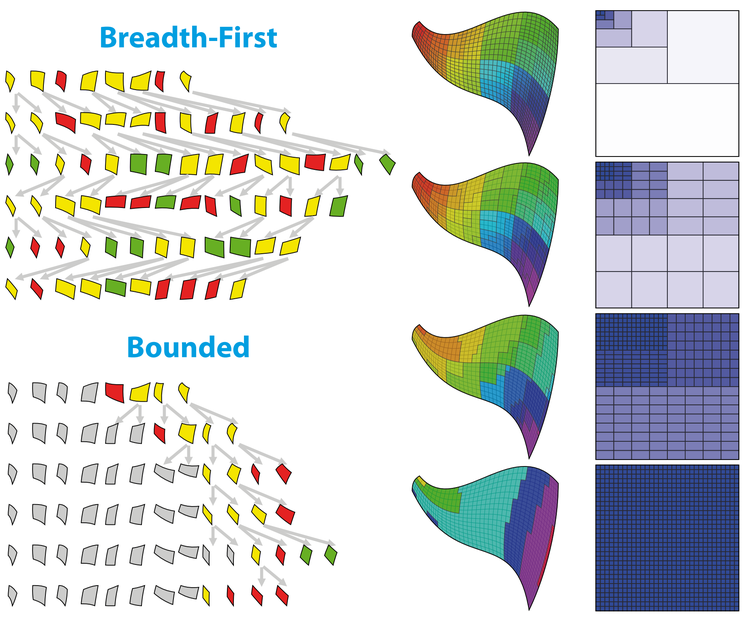
Subdivision Memory Usage

While the recursive bound & split operation (essentially a depth-first tree traversal) can be parallelized for the GPU by turning it into a breadth-first traversal, this is limited by its high worst-case memory consumption.
The image on the right shows the memory necessary for breadth-first subdivision as a camera moves through a scene. The highest spike requires over 1GB of memory.



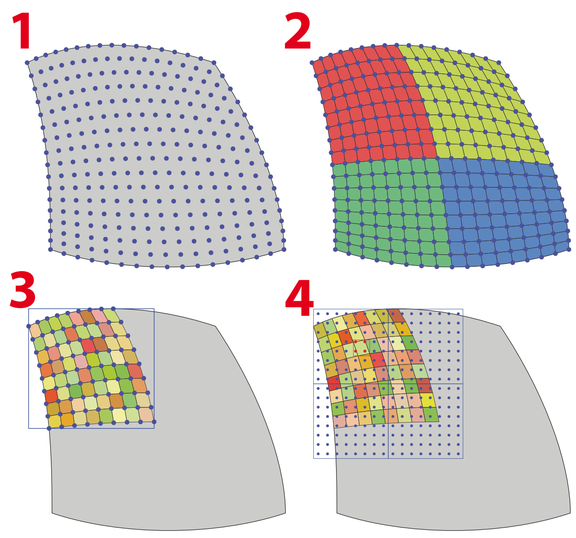
Parallel Subdivision with Bounded Memory

We present a generalization of breadth-first subdivision, which allows controlling the maximal memory consumption by limiting the number of surfaces that are processed concurrently. This gives the user a smooth transition between memory usage and performance.
The column on the very right shows the effect that changing this parameter has on the worst case number of surfaces in memory.
Our method also preserves locality as shown in the middle column. Sub-patches sharing the same color are processed together.



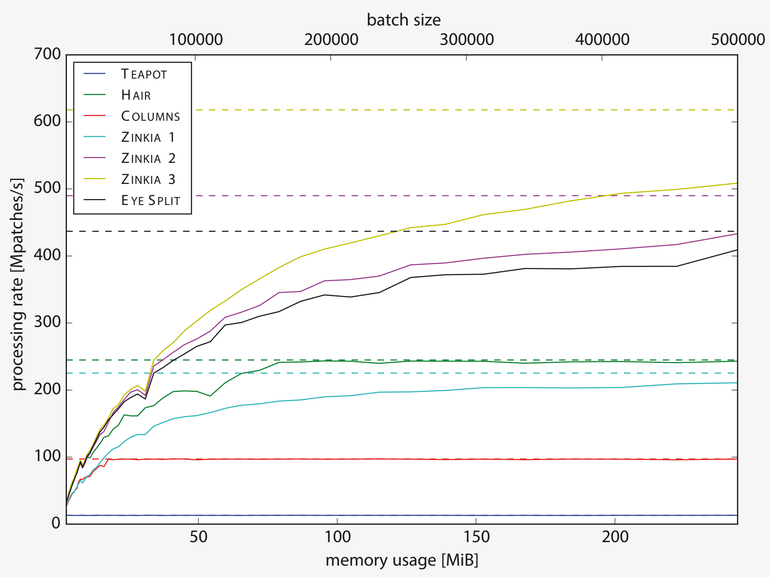
Shading and Rasterization

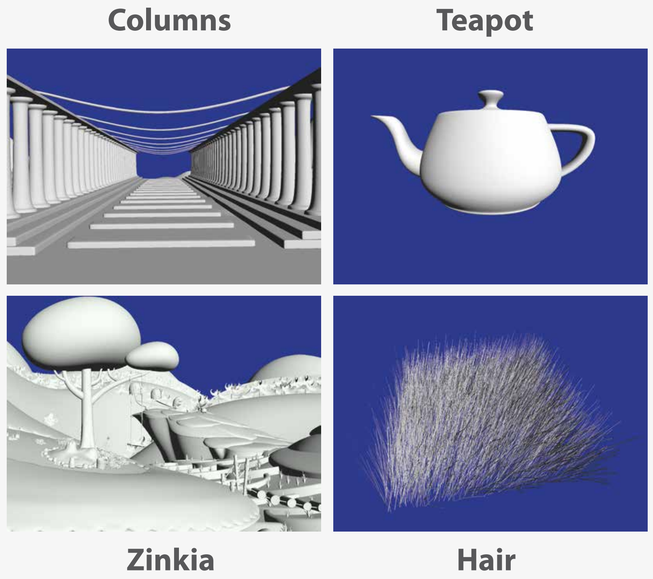
Once the surfaces have been successfully subdivided, they are evaluated at regular positions to create grids of polygons. After this, the polygons are shaded and rasterized.
These operations are implemented as a succession of OpenCL kernels. Dicing happens in a separate kernel (**1**). The resulting grids are split into smaller chunks (**2**), and polygon shading is performed (**3**). The shading kernel also checks if all polygons are back-facing and the chunk can be culled.
Chunks are then rasterized in a separate kernel that checks for polygon/sample intersections for all chunk polygons in parallel (**4**).



Results, Conclusions, Outlook

The figure to the right plots the surface subdivision rate for the amount of assigned memory. The performance converges toward that of breadth-first. This, combined with our shading and rasterization kernels, allows us to render very detailed scenes at real-time or interactive framerates.
One of the largest performance costs when subdividing is host-device communication. Device-side kernel enqueue could potentially improve performance significantly.





Kontakt: weber.t@gmx.at, thomas.weber@cg.tuwien.ac.at