Large Terrain Modeling and Visualization for Planets

Steven Myint Abhinandan Jain Jonathan Cameron
Christopher Lim

Jet Propulsion Laboratory, California Institute of Technology

IROS 2011

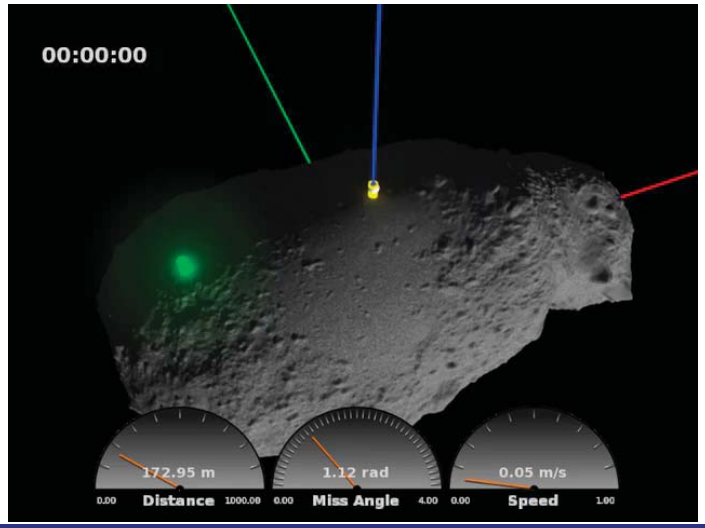
http://dartslab.jpl.nasa.gov

DARTS lab

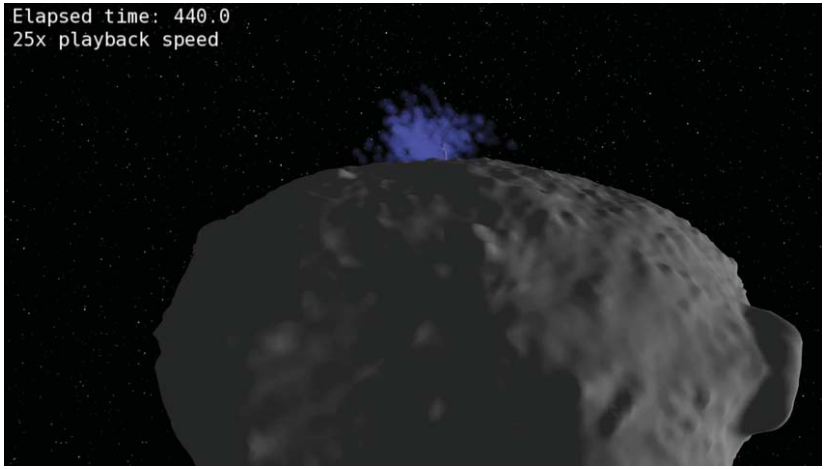
Dynamics And Real-Time Simulation (DARTS)

- EDL simulations (DSENDS)
- Rover simulations (ROAMS)
- Airship simulations
- Robotic arm simulations

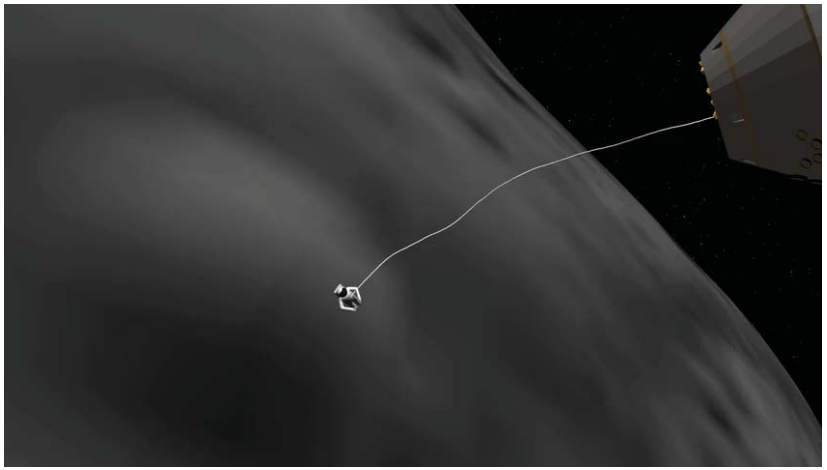
DARTS lab examples (near earth object)



DARTS lab examples (ejecta)

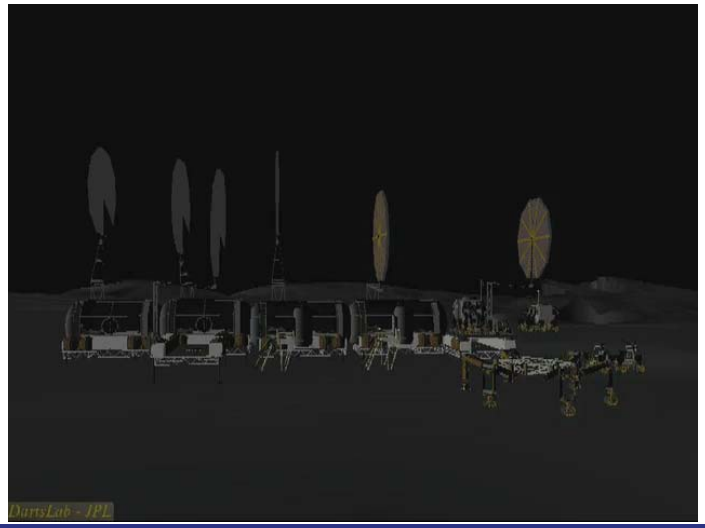


DARTS lab examples (tethering near small body)

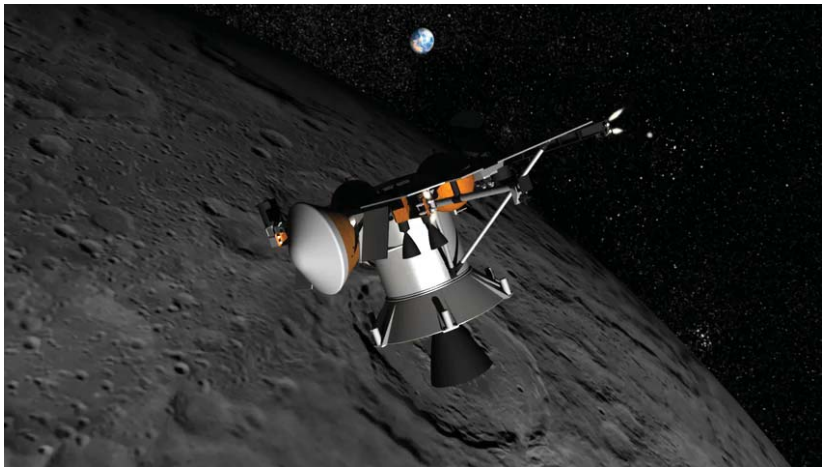


DARTS lab examples (surface operations)

DARTS lab examples (power analysis)



DARTS lab examples (spacecraft)

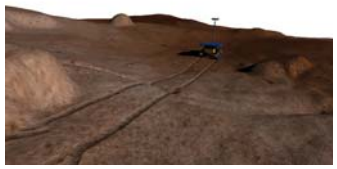


Large terrain modeling and visualization

Large terrain use cases

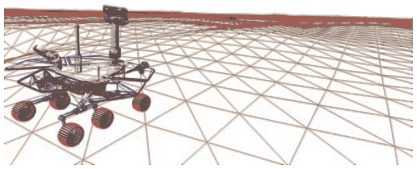
Our real-time simulations often require large terrain modeling/visualization support.

- EDL simulations
- Rover simulations



Large terrain use cases (continued)

- Centimeter-resolution terrain data
- Billions of vertices
- Gigabytes of data
- Too much data to load all at once (due to time and memory constraints)
- Too much data to render in real time (30 fps)

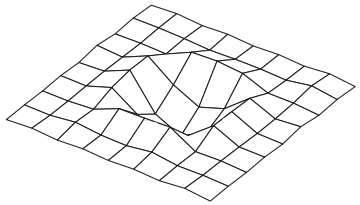


SimScape terrain framework

- Models DEMs, planets, and arbitrary meshes
- C++ and Python APIs
- Store data in HDF5 (Hierarchical Data Format) for fast random access
- Import data from various data formats (PDS, ISIS, GeoTiff, etc.)

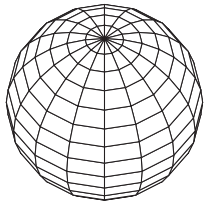
Digital Elevation Maps

- Regular rectangular height data
- Fast access without much arithmetic
- Useful for modeling relatively small areas
- Used extensively in rover simulations



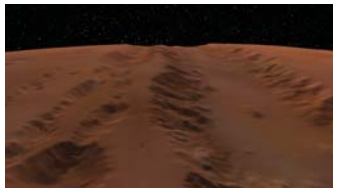
Planets

- Grid of height data in spherical coordinates
- Useful for larger areas when we must consider planet curvature



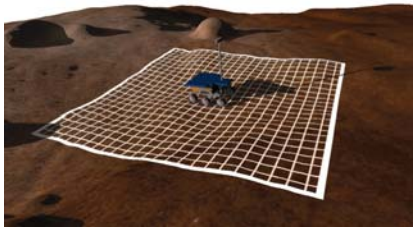
Large terrain modeling

- Support storing and loading of large planetary scale data
- Support data sets that cannot fit into memory
- Support random access
- Be able to access the data in real time



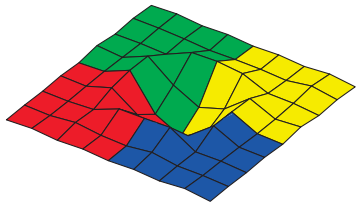
Paging

- Only a subset of terrain data is paged in
- SimScape builds on top of HDF5 to support paging
- In the rover example, only a small patch of data under the rover is kept in memory



Data tiling

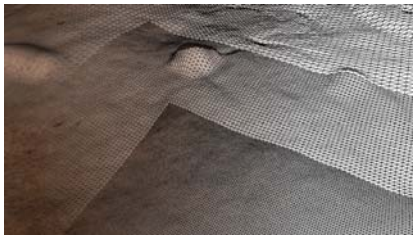
- Large data sets are sometimes broken up into separate tiles
- Mainly useful when modifying and writing data (less memory consumption at any one time)
- Useful for parallel processing of data (running on supercomputer)
- Many data sets (e.g. MOLA) come in tiled format
- Transparent to the data loading API



Continuous level of detail visualization

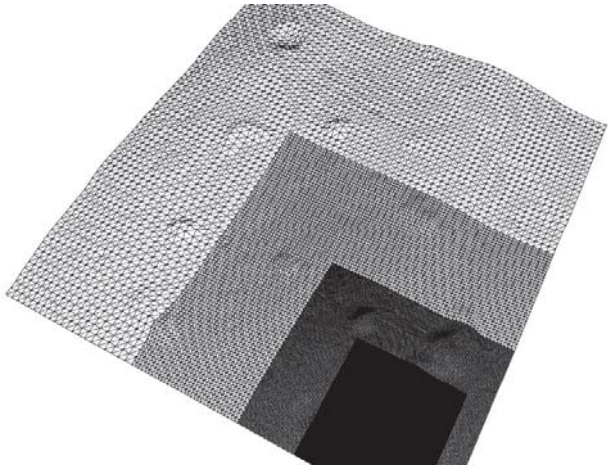
A continuous level of detail (CLOD) technique allows us to render high resolution data only where we want it (e.g, where the camera is pointing).

- Render very large data sets that can't normally be rendered all at once by the graphics card
- Render in real time
- Unlike a discrete LOD technique, transition between levels is smooth (no popping between levels)



Continuous level of detail visualization example

Clipmapping



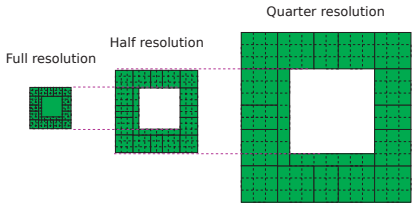
Clipmapping

- Concentric rings of data
- Innermost rings have the highest resolution data
- Each ring is composed of a regular grid

Clipmapping takes advantage of perspective projection

Other advantages of using clipmapping

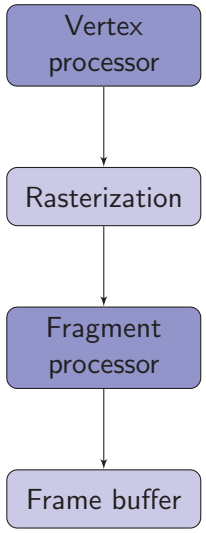
- Each ring has the same basic regular grid geometry
- Nearly the same operations done to each vertex
- Computation easily parallelizable (amenable to GPU implementation)



GPU implementation

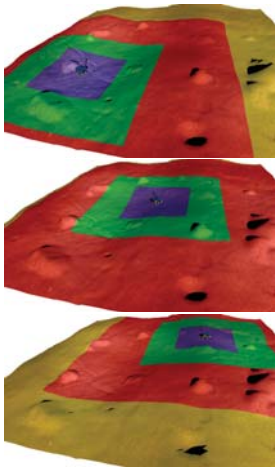
Implementing clipmapping on the GPU allows us to utilize hundreds of GPU cores.

- Upload height map data to the GPU as a texture
- Vertex program computes vertex positions/normals based on:
  - Center position of all clipmaps
  - Clipmap ring level
  - Height map texture
- Fragment program computes final pixel color based on:
  - Albedo sampled from texture
  - Surface normal (passed in from vertex shader)



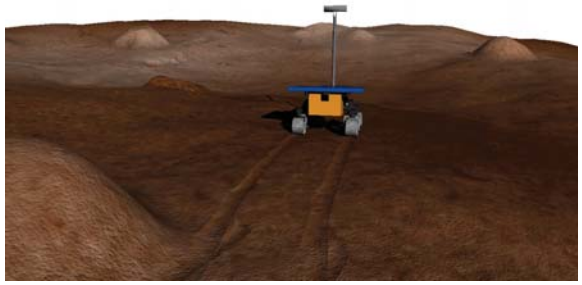
Moving clipmaps

The high resolution area is moved by moving all clipmap rings.



Overlays

Wheel tracks are overlaid on the terrain by perturbing the surface normals on a per-pixel basis.



Overlays (continued)

- Height maps
- Albedo maps
- Slope maps

Summary and future work

- Summary
  - With paging, we can work with arbitrarily large terrain data sets
  - We can visualize these large terrains using continuous level of detail
  - We make use of the GPU's multi-core architecture in our implementation of continuous level of detail
- Future work
  - Support paging of high-resolution (albedo) textures (instead of just geometry)
  - Improve efficiency of GPU programs by dynamically autogenerating them based what features are being used (like height maps)