

DS7: Investigate the feasibility of implementing a terrain modification

The current implementation of Digital Spaces does not support dynamic modification of terrain meshes. This document details the requirements of a practical terrain modification system, and the feasibility of implementing these requirements.

Current Implementation

The current implementation uses a simple pre-modeled mesh for both the visual representation and physics simulation of the terrain. This is the quickest to implement, and treats the terrain as simply model in the scene graph.

Practical Implementation

In a practical terrain system, the visual representation should be constructed at run time from another data source. Commonly this is a bitmap height-map. The physics representation of the terrain should not be a mesh (or polygon soup), but rather generated from the same height-map data. Also the terrain should be "painted", meaning that the visual representation allows the blending of different types of material. Commonly this may be grass, dirt and concrete, in a lunar setting it would be different types of regolith and rock densities.

Additionally, it should be capable of paging, which allows distant terrain to not be rendered or simulated. As the viewpoint moves, additional areas of terrain are "paged" in, while now distant areas are "paged" out. This allows for larger terrains with a lower system load.

For simulation of excavation, it will be required that the resolution of the terrain data be quite fine grained. For example, common terrain implementations allow for a resolution of a meter, which is usually sufficient for interaction at a human scale. However when modeling excavation, for highest visual and simulation fidelity, the terrain data resolution should be finer than the size of the bucket being simulated. In order to support this, a terrain system should allow differing terrain resolution, possibly on a per page basis.

Implementing in Digital Spaces

Graphical Implementation

The current implementation of Digital Spaces does allow for dynamic construction and alteration of meshes (through the use of OGRE). Additionally, there are freely available projects which provide a paged terrain implementation for use with OGRE, and there are suggestions that a future version of OGRE will include paged terrain as a core feature.

In the worst case, the available implementations can be used as reference to construct our own paging and terrain system, however we should be able to use a goodly portion of the provided functionality as-is (such as mesh construction from height data), although due to our use of a common scene-graph (versus OGRE's

private scene-graph), it is unlikely we will be able to use the available implementation without some Digital Spaces specific customization.

Physics Implementation

The physics engine used in Digital Spaces (ODE) provides a mechanism commonly used for modeling terrain. Digital Spaces simply needs to be able to supply a surface height at any coordinate on the terrain. Anything below this height is considered to have penetrated the solid terrain, and will be pushed away.

This would be an improvement over the current implementation in that the current implementation uses a polygon mesh (implemented simply as a soup in ODE). This means that the terrain is actually infinitely thin, and once an object passes through the polygon, there are no further collisions. This can cause rapidly falling objects to pass through the terrain, as the simulation time-step resolution never resolves the object as intersecting the polygons of the terrain. With the terrain height-map method, this cannot happen.

Common Functionality

Due to the modularized nature of Digital Spaces, it should be able to make certain portions of the terrain simulation a common module. This separation not only reduces duplication of calculation (eg both the visual and the physics interpreting the height-data separately), but with the clean abstraction design used in Digital Spaces between modules, this should allow expansion of the terrain generation without affecting the simulation aspects. Examples of this that may be possible are using remotely stored data to generate height values, or using mathematical models such as Perlin Noise (a form of fractal) to generate realistic appearing terrain from simple seed values.