LambdaCube 3D - A purely functional API for GPU graphics

Csaba Hruska[1]

[1]Freelancer

At its heart, the graphics pipeline is simply a configurable data-flow network. The main input of the network arrives as a stream of vertex descriptions, and additional data can be provided in various slots, which are constant during a rendering pass: uniforms of basic types and samplers (textures with some attached logic). After some processing steps, the final output is one or more raster images.

We can look at this data-flow network as a mathematical function that maps scene descriptions to bitmaps. The internal structure of this function can be defined in terms of smaller building blocks that correspond to various stages of the pipeline. Even the programmable pipeline has a more or less fixed global structure, but the transformations within the main stages can be freely defined through shaders, which suggests that the pipeline can be naturally modelled as a higher-order function.

I'll present LambdaCube 3D, a Haskell-like purely functional domain specific language for GPU programming. The purpose of t is tool is to provide a platform and host language independent graphics API. It allows the programmer to define the rendering pipeline with a single language, which is compiled into shaders and CPU-side setup code.

The basic motivation for creating LambdaCube is really simple: it represents the perfect intersection of different fields that have excited us for a long time – pure functional programming, computer graphics, and compiler research. We intend to explore a totally new branch of the design space for authoring and developer tools.