What is Hypertexture?

Hypertexture is an Oscar award winning method for describing three dimensional shapes. Normally, 3D models are considered as points in a 3D space that are either part of an object, or outside the object. However, this is a simplification that robs 3D models of more complex phenomena such as clouds, fire or fur, which that cannot be categorised in such a binary manner.

In 1989, Ken Perlin introduced the notion of hypertexture, providing a way of including such exciting features. The key part of hypertexture is that shapes are not modelled as a surface, but rather as density values. The phenomena of hypertexture lies in areas on the boundary of the object with a density between 0 and 1, the soft region, where 0 is not part of the object and 1 is solidly the object. Intuitively, fire is hypertexture because it is not entirely solid.

Approach

The language has been created using type class morphisms, where the design process is guided by a mathematical model. The end result is a combination of a Haskell type class and denotational semantics. This method has two main benefits: the desired behaviour of each function in the type class is clear; and any laws associated with the underlying mathematical model come for free with the type class. It will allow the creation of a uniform and consistent interface for different hypertexture implementations that gives guidance to implementers and reliability to users.

Example:

If a language for a key to value map was to be created. The mathematical model could be selected to be partial functions because the designer of the language wants any implementations to fail gracefully. Now the meaning of any operation created for maps has the meaning of that operation on partial functions e.g. the meaning of empty for maps is the meaning of empty on partial functions. This is expressed in two parts. The model:

⟦Map k v⟧ = k -> Maybe v

And semantic function translating a Map k v to its semantic meaning:

```
⟦.⟧  : Map k v -> (k -> Maybe v)
⟦empty⟧          = λk -> Nothing
⟦insert k' v m⟧ = λk -> if (k == k') then (Just v)
                                     else ( ⟦m⟧ v)
```

Contributions

This project provides a formalisation for hypertexture in the form of a domain specific language. While hypertexture has been used widely in the film industry to create engaging CGI shots, it previously had no formalisation. Perlin's work introduced methodology to create hypertextures, this language extends that work, providing a uniform interface for creating hypertextures.

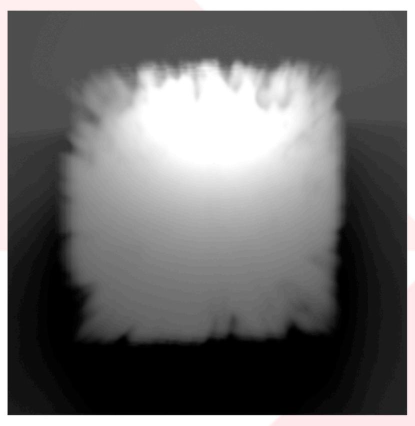
The goal of this language was to make creating hypertexture phenomena easier for the user. Previously to create hypertexture bespoke code was needed. Now the implementation can be abstracted away behind the language, allowing better exploration of what hypertextures can be created.

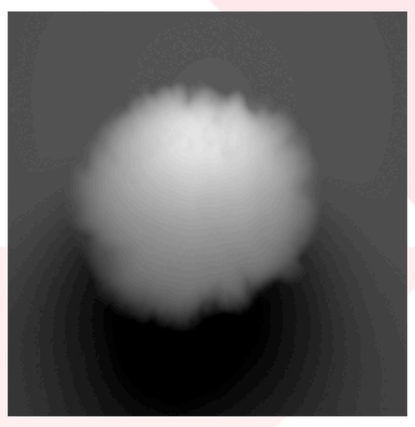
```
union rectangle
      (union rectangle
             rectangle)
```

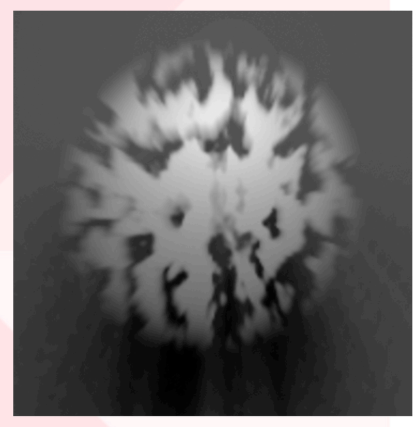
```
implicit2D fancyY
```

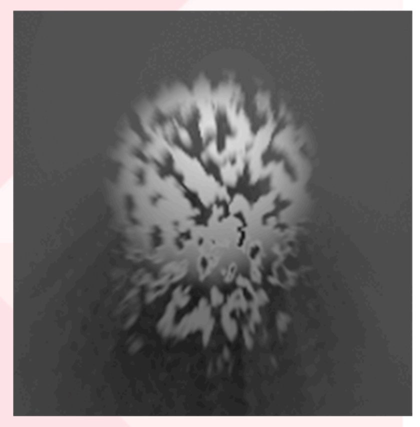
```
modulatePoint addNoise letterP
```

```
fire letterE
```










```
modulatePoint
  dripY
  letterT
```

```
implicit3D
  (λxyz.  if (x² - y² * z ⩽  0) 1
          else 0)
```

```
complement letterR
```

```
modulateGeometry hair letterT
```

University of BRISTOL

Denotational Semantics

The hypertexture language uses three different mathematical models that complement each other. This allows the semantics to be more specialised towards what they are describing. Two and three dimensional shapes are modelled as polynomials using implicit functions. To combine shapes together sets are used. The membership function of these sets takes the implicit function of the shape, evaluates it at a point, and if the value is less than zero then the point is a member. Finally, to represent the exciting phenomena of hypertexture, fuzzy sets are used. Here the density is represented by the point's continuous membership value between 0 and 1.

Polynomials

$$[\![.]\!] :: Prim2D \to f(x,y)$$
$$[\![Circle]\!] = sqrt(x^2+y^2)-1$$
$$[\![Triangle]\!] = x+y-1$$
$$[\![Square]\!] = |x|+|y|-1$$

$$[\![.]\!] :: Prim3D \to f(x,y,z)$$
$$[\![Cube]\!] = max(|x|,|y|,|z|)-1$$
$$[\![Cylinder]\!] = (sqrt(x^2+y^2)-1)+z-1$$
$$[\![Sphere]\!] = x^2+y^2+z^2-1$$
$$[\![Cone]\!] = x^2+y^2-(z-1)^2 \ +z-1$$
$$[\![Torus]\!] = (sqrt(x^2+y^2)-1)^2+z^2-0.5$$

$$[\![.]\!] :: Implicit2D \to f(x,y)$$
$$[\![Implicit2D\ f]\!] = f$$

$$[\![.]\!] :: Implicit3D \to f(x,y,z)$$
$$[\![Implicit3D\ f]\!] = f$$

$$[\![.]\!] :: Prism3D \to f(x,y,z)$$
$$[\![Prism3D\ p]\!] = ([\![p]\!]\ x\ y) + (|z|-1)$$

Normal Sets

$$[\![.]\!] :: Boolean \to Set$$
$$[\![Empty]\!] = \emptyset$$
$$[\![Universal]\!] = U$$
$$[\![Union]\!] = \cap$$
$$[\![Intersection]\!] = \cup$$
$$[\![Complement\ s]\!] = \overline{s}$$
$$[\![Difference\ s_a\ s_b]\!] = s_a \setminus s_b$$

$$[\![.]\!] :: Transformations \to Set$$
$$[\![Translate\ t\ s]\!] = \lambda p \to [\![s]\!]\ (p-t)$$
$$[\![Rotate\ r\ s]\!] = \lambda p \to [\![s]\!]\ (r^{-1}p)$$
$$[\![Scale\ n\ s]\!] = \lambda p \to [\![s]\!]\ (np)$$

Fuzzy Sets

$$[\![.]\!] :: Soften \to Fuzzy\ Set$$
$$[\![ConstSoften\ n\ s]\!] = \lambda p \to n * ([\![s]\!]\ p)$$
$$[\![FromPointSoften\ c\ r\ s]\!] = \lambda p \to (1-d/r) * ([\![s]\!]\ p)\ where\ d = euclideanDist\ p\ c$$

$$[\![.]\!] :: PrimSoft \to Fuzzy\ Set$$
$$[\![SoftSphere]\!] = \text{softSphere 1 (0,0,0)}$$

$$[\![.]\!] :: ModulateDensity \to Fuzzy\ Set$$
$$[\![ModulateDensity\ f\ s]\!] = f \circ [\![s]\!]$$

$$[\![.]\!] :: ModulatePoint \to Fuzzy\ Set$$
$$[\![ModulatePoint\ f\ s]\!] = [\![s]\!] \circ f$$

$$[\![.]\!] :: ModulateGeometry \to Fuzzy\ Set$$
$$[\![ModulateGeometry\ f\ s]\!] = f[\![s]\!]$$