Spore API: Accessing a Unique Database of Player Creativity

Dan Moskowitz
Maxis, Electronic Arts
dmoskowitz@maxis.com

Shalin Shodhan
Maxis, Electronic Arts
sshodhan@maxis.com

Michael Twardos
Maxis, Electronic Arts
mtwardos@ea.com

1 Introduction

In the video game Spore, players create their own creatures, buildings, and vehicles using intuitive yet powerful editing tools. During the course of game play, these creations are automatically published to our servers and made available in our online catalog for browsing, downloading, and sharing. Players can add tags and write descriptions about their creations, as well as assign ratings, write comments, organize creations into feeds called Sporecasts, or download other players' creations into their own game. Players can also choose to organize the flow of creations into their game by subscribing to Sporecasts and adding other players as buddies. Since its release in September 2008, Spore players have created and published millions of creations resulting in a unique database of player imagination and creativity.

2 The Spore API

In addition to a player's creations being shared on the Spore website, the data is also accessible with a public web service called the Spore API. For each creation, the web service provides information about its construction, morphology, color, and other statistics. This information is sufficient to reconstruct the asset on any other machine running Spore. The API also provides rich data about relationships between users and creations, such as lists of creations per user, lists of buddies per user, or the assets rated highest by the community. This web service enables developers to build applications that use Spore data in new and unique ways. In this talk, we provide examples of applications and visualizations that use the Spore API to explore this data.

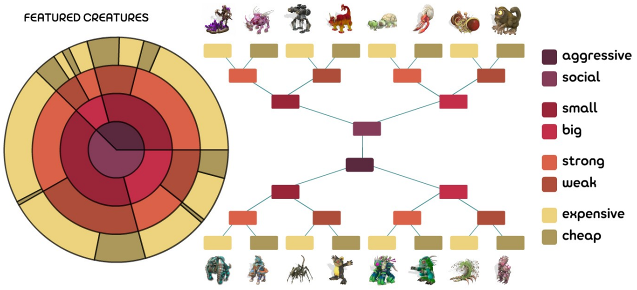

Figure 1: *Two examples of creature taxonomy visualizations*

3 Creature Taxonomy

Inside the creature editor, each rigblock assigns the creature a corresponding game play attribute and level, such as "carnivorous", "bite +3", or "speed +2". Using the web service, which provides these abilities and other statistics about a creature, we can classify creatures based on behavior, diet, cost, complexity, and size. We use these attributes as nodes in a binary tree to sort a vast range of creatures into appropriate leaf nodes. This reveals the amount of creations with specific attribute combinations. We experiment with different formats for visualizing this data. (Figure 1) These visualizations have been fed back to the community as a challenge for them to make creatures with rarer combinations of traits.

4 Asset Reconstruction and Breeding

Using the Spore API, we show that it is possible to reconstruct Spore creations in a web browser without the core data included on the Spore game disc. Our application renders Spore creatures in a stylized 2D form using Flash. (Figure 2) An artist created simplified 2D versions of each rigblock, which we position and color appropriately based off the data in the XML. The ubiquity of the web means that we are able to create a low barrier of entry for developers wishing to showcase Spore creatures inside their own applications. One such application is the Creature Breeder which combines traits of two creatures to form unique mixed-breed offspring. The results of this application are visualized using the stylized 2D look.

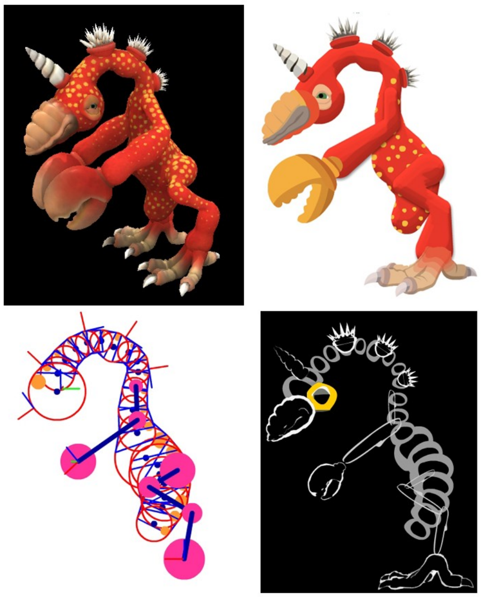

Figure 2: *Using the XML to reconstruct creatures in 2D*

5 Creation Lineage and Aesthetic Matching

When a player uploads a modification to a Spore creation, the upload transaction contains the ID of its predecessor, which is accessible in the API. This allows us to look at the evolutionary paths of creatures that progress through the game, as well as observe patterns of modification of the most popular creations on the website. We experiment with interesting ways of visualizing this data and tracking lineage trends.

Another area of work is aesthetic matching. Given one creation as a target, we experiment with ways of searching our database for other creations that visually match the look and/or theme of the target creation. This is a complex problem given the staggering number of creations stored in the database. To aid in the search, certain rigblocks have been pre-tagged with aesthetic markers allowing us to more easily match, for example, vehicles made with the same style of chassis.