The LEGO Movie: Bricks, Bricks and More Bricks

Bryan Smith Daniel Heckenberg Jean-Pascal leBlanc
Animal Logic*

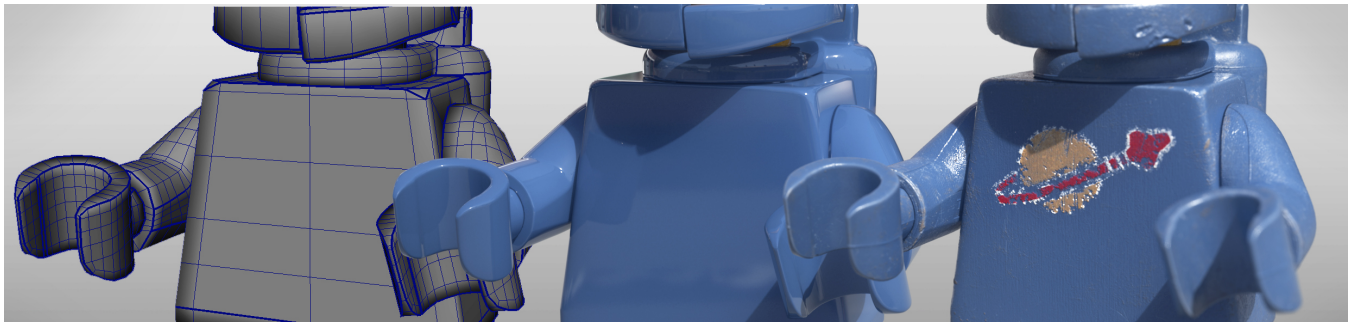

Figure 1: *Geometry/Plastic/Final Shading*

Abstract

In *The LEGO Movie* everything on screen from characters and sets to oceans and explosions is realised with bricks. By building a brick library we were able to automatically assemble optimized models from XML files output by LEGO Digital Designer. A sophisticated shader and numerous texture variations for each brick were also created, allowing us to produce models with high quality default shading. For realistic lighting a Physically Based approach was used, as well as aggressive geometry instancing. To simulate imperfectly assembled models, a small procedural transform jitter was applied.

1 Modeling

To achieve photo-realistic rendering of bricks in Pixar's *Renderman* we developed a set of tools and modeling techniques that incorporated software developed by The LEGO Group. Modellers used a customized version of the publicly available *LEGO Digital Designer* (LDD) for constructing the models, brick by brick. This software not only provided access to the brick library, but also ensured physically correct brick placement and produced an XML output. We developed an importer/exporter of this format for Maya to allow finer control when required. An automated process called *Shell Baking* was developed to construct our own geometry, utilizing optimisation algorithms developed by LEGO to produce several levels of detail for each model.

A procedural brick placement system called LEGOscape was developed to allow users to rapidly create large environments such as terrains, canyons, or mesas. Using an iterative algorithm, a simple mesh volume is filled with bricks of in user configurable patterns.

Render Brick geometry was carefully constructed to allow for procedural optimization when connected to other bricks. The geometry provided by LEGO was unsuitable for final rendering, so a library of over 2000 Subdivision Surface bricks was commissioned. A standard set of studs and tubes were designed early on and a tool was created to help place these on the geometry.

*{bryans, danielh, jpascall}@al.com.au

2 Surfacing

Efficient surfacing of brick-based models required a specialised, multi-level pipeline. Semi-procedural scratches and stud details, combined with procedural edge wear meant that models would have high quality default shading even without any individual asset work. Multiple variants of textures could be randomly or manually assigned to channels such as plastic warp, roughness and decals. A single LEGO shader was used, with texture assignments, plastic type and surfacing properties managed by references to external files.

To embody surfacing effects and details relating to the bricks in their context in an assembled model, PTEX was used for baking of occlusion for procedural grunge and wear shading, and for optional assembly-level texture painting. A customized texture painting pipeline allowed us to author PTEX files efficiently with merged and subdivided meshes for texture painting and automatic geometry remapping in instances of model changes.

Character faces were rendered with a custom vector-based RSL shadeop. Animators worked with rigged curves and used a GLSL shader for real-time review. These non-resolution limited face shapes were then procedurally shaded to give the effect of the layered printing used in minifig decals, and could be efficiently rendered at any scale. For use in crowd scenarios, a range of face poses were baked to textures.

3 Lighting

A Physically Based approach was used for Lighting. Renderable models aggressively exploited PRMan 17s new instancing mechanisms to improve performance and memory usage. To simulate imperfectly assembled models, a small procedural transform jitter was applied to achieve more interesting lighting highlights and shadows. A workflow for automatic instancing of light rigs in assets (e.g. car headlights) was developed that extended to procedural animation FX (e.g. lasers).