The LEGO Movie: Construction, Animation And Demolition

Aloys Baillet Daniel Heckenberg Eoin Murphy Aidan Sarsfield Bryan Smith
Animal Logic[*]

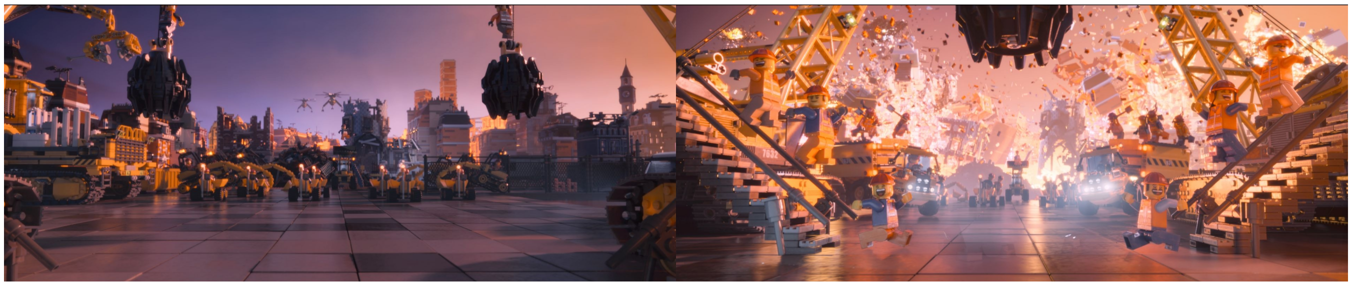
Figure 1: *Set destruction with volumetric simulation of explosions and dust clouds converted to bricks.©Warner Bros Inc., Village Roadshow, The LEGO Corporation. All rights reserved.*

Abstract

The creative requirements for *The LEGO Movie* demanded that the entire world be made of individual LEGO bricks, with no cheating. Whole buildings needed to be demolished into their component bricks, vehicles pulled apart and re-assembled differently and some parts of the set were to be ripped up and formed into other objects. Even simulated FX like oceans, dust and clouds were to be made of bricks. To achieve this challenging brief and support rendering of massive amounts of geometry, we added a brick-based layer under our existing asset pipeline, which we kept 'live' all the way through to final rendering. This approach allowed us to leverage brick specific render optimisations, and to automate various tasks such as model building and surfacing.

1 Construction

LEGO supplied us with a database of their bricks that contained useful metadata. Each brick had an ID which was used to look up information about its size, weight, center of gravity and connectivity points. For flexible pieces such as chains and tubes, a simple joint chain was also available for deformation. A library of over 2000 different bricks needed for the movie were re-modelled using SubDivision Surfaces, which were ideal for representing the "studs" and bevelled edges with minimal topology.

We used *LEGO Digital Designer*(LDD), a freely downloadable tool, to build our models. LDD provided access to the library of available bricks and the palette of LEGO materials, and also handled brick connectivity, snapping, grouping and other LEGO-specific functionality. The files produced by LDD were fed into our "ShellBake" process which generated multiple Levels of Detail for the LEGO model (e.g with inner studs and tubes stripped out) and in a number of forms (render geometry, GL-friendly geometry, point clouds for FX), all tagged with indices into the brick database. Accompanying "sidecar" data contained heavier, often LEGO-specific data which was useful for the shader at render time.

There were 3 layers to the surfacing of models - an automatic assignment of colors, transparency and other visual properties from the LDD models; a semi-procedural level of per-brick surfacing to provide detail such as scratches, wear and tear, decals etc.; and a 'model' level to add grunge and oxidization effects. The per-brick surfacing could contain texture and other variations to reduce visual repetition.

2 Animation

The Environment and Layout teams dressed the sets traditionally, but could tweak position and visibility of any bricks. New bricks could also be created by choosing from the brick library, or duplicating an existing brick in the scene.

Traditional rigging techniques could not handle large props made of thousands of bricks, so our node-based procedural system *ALF* was used with a combination of dynamic reparenting and geometry merging to enable efficient rig building, and fast playback for animators. We also developed a novel "Brick Blur" technique that allowed animators to artistically represent motion blur as a smear of colored bricks replacing the original character.

3 Demolition

Destroying LEGO models was made easier thanks to their brick-based nature. Houdini was used to set up glue constraint networks based on connectivity for Rigid Body Simulation. Voxel-based effects such as smoke, explosions and water were done in the traditional way, then replaced with LEGO bricks of a matching size and shape. Crowds were also brick-based, with possibilities to 'mix and match' minifigure parts for extra variation.

For rendering, a custom PRMan procedural was used to instance and optimise bricks on the fly. Tools were developed to allow lighters to instance light rigs or add incandescence to glowing bricks based on brick properties. The sheer quantity of geometry to be rendered was the biggest challenge, so aggressive geometric instancing was required as well as simplifying the calculation of transparency and shadows in the shader. Lighters could also perform custom optimisations based on brick properties such as curvature, concavity and so on.

[*]{aloysb, danielh, eoinm, aidan, bryans}@al.com.au

SIGGRAPH 2014, August 10 – 14, 2014, Vancouver, British Columbia, Canada.
2014 Copyright held by the Owner/Author.
ACM 978-1-4503-2960-6/14/08