Bond: USD-Integrated Hybrid CPU, GPU Deformation System

Aloys Baillet
Animal Logic
aloysb@al.com.au

Josh Murtack
Animal Logic
joshm@al.com.au

Hongbin Hu
Animal Logic
hongbinh@al.com.au

Haoliang Jiang
Animal Logic
haoliangj@al.com.au

Michael Quandt*
Animal Logic
michael@mquandt.com

Figure 1: a) *Bond* rig in Maya viewport; b) *Bond* corresponding geometry deformation graph;

ABSTRACT

We introduce *Bond*, a proprietary deformation system able to load geometry data directly from Pixar™ Universal Scene Description (USD) and to compute complex deformation chains on the GPU using NVIDIA® CUDA®. *Bond* also integrates tightly with Autodesk Maya®. This system follows on from our work to integrate USD into our animation pipeline [Baillet et al. 2018].

Bond has been used to deform all characters and props on the *Peter Rabbit 2* movie's 1300 shots to achieve high frame rate during playback and rig interaction.

CCS CONCEPTS

• **Computing methodologies → Computer graphics**.

KEYWORDS

animation deformation USD maya

ACM Reference Format:
Aloys Baillet, Josh Murtack, Hongbin Hu, Haoliang Jiang, and Michael Quandt. 2020. Bond: USD-Integrated Hybrid CPU, GPU Deformation System. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Talks (SIGGRAPH '20 Talks), August 17, 2020*. ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3388767.3407324

1 INTRODUCTION

Bond was designed to meet two main goals: to improve rig load times and to increase the execution speed of complex deformations.

*Also with Frontier Developments.

SIGGRAPH '20 Talks, August 17, 2020, Virtual Event, USA

ACM ISBN 978-1-4503-7971-7/20/08.
https://doi.org/10.1145/3388767.3407324

Firstly, to decrease rig load times we leveraged our new animation pipeline built on top of USD and the USD binary *Crate* file format.

Secondly, in 2016 we had established via a set of prototypes that the fastest way of computing the deformation of production-sized meshes was to use GPUs and the CUDA programming language.

We started developing *Bond* by simply using USD as our "Rig Bindings" serialization system on the first *Peter Rabbit* production: we created a set of tools to import and export Maya skin weights and blend shapes to and from USD files. This gave us highly valuable production data that we could use to further test *Bond*.

The full *Bond* deformation system was used in production on the *Peter Rabbit 2: The Runaway* sequel in 2018.

2 RELATED WORK

We evaluated Pixar's *usdSkel* schemas very early on, but *usdSkel*'s requirements about being optimized for on-device deformation or crowd characters were too restrictive for our creative needs. *Bond* instead uses custom USD schemas to store arbitrarily complex bindings data in an efficient and lossless way. We would be excited to share our findings and our USD rig bindings schemas.

3 ARCHITECTURE

The *Bond* core engine is written in *C++14* and is exposed via a simple *C* API. We typically use it embedded in Maya, but it can also run standalone. It has a plugin system to provide node implementations for both CPU and GPU modes. The memory is fully managed by the Bond Chain on both CPU and GPU.

We chose to create linear chains of Bond nodes, where each node is evaluated in order. Bond processes buffers, which have a type and a string ID, and tracks which buffers are requested by the client and which nodes can provide them (e.g. Maya viewport requires point positions, triangles, normals, and possibly more buffers for complex shading).

This forms a dependency graph of buffers as represented in Figure2. A validation step is run before evaluation and verifies if all requested buffers can be computed correctly.

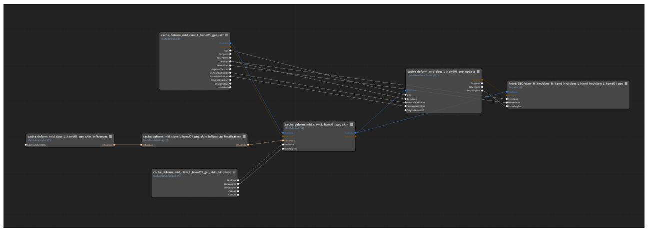

Figure 2: A simple *Bond* chain graph representation.

For maximum performance, a single *Bond* chain can use multiple geometries to deform multiple targets, with all the work scheduled on the GPU. An example of this kind of complex configuration would be a low resolution mesh used as a cage to deform a high resolution mesh, with curves being "rivetted" to the surface of that second mesh.

4 INTEGRATION WITH USD IN MAYA®

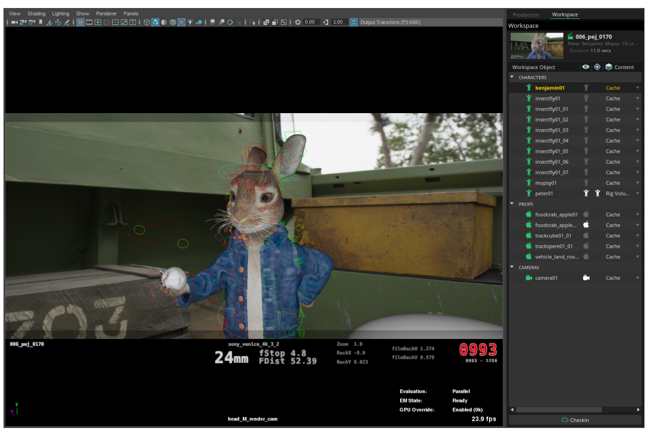

Figure 3: A shot in *Peter Rabbit 2* showing the Maya control rig and the *Bond* deformed geometry, Forge UI docked.

Our rigs have a very strong separation of the "control rig" and the "deformation rig". We load the control rig as a Maya reference, and the deformation rig by generating *Bond* maya nodes that fetch the USD geometry and bindings data from the *UsdStage* managed by the open source *AL_USDMaya* [Bateman et al. 2019] *ProxyShape* plugin node.

The high-level USD management is done by the *Forge* animation tool: the activation of a "bindings" USD Prim will cause the *AL_USDMaya* translator plugins to create the *Bond* shape hierarchy and deformation chains for the whole rig.

Riggers have access to the CPU mode by default to enable most features such as component selection, float weightmap painting, skin weight painting and snapping.

Animators get GPU as the default mode via a single Maya *SubSceneOverride* per rig to enable the fastest playback speed. This scene configuration allows us to schedule the deformation computation as early as possible on the GPU.

5 RESULTS

In order to fairly compare the performance differences between Bond and Maya 2018, we created a simplified production rig, removing the rabbit's whiskers and more advanced face deformation nodes. With only skinning and blend shapes, Bond CPU mode was about 33% faster than Maya CPU deformers, and Bond GPU mode was around 11% faster than Maya's OpenCL GPU evaluator.

As additional deformation nodes were added to the Peter Rabbit rig, *Bond* managed to keep the rig playing at around 34 frames per second with full high resolution face and body. This allowed animators to work on one character with full resolution rig and multiple geometry caches (streamed from USD by Pixar's HdStorm) at 24fps in many cases.

Since the end of the *Peter Rabbit 2* production we integrated *Bond* with Maya 2020's new Background Evaluation Caching by using our CPU deformation chain in the background context and keeping GPU for interaction and non-cached playback. One unexpected bonus we discovered is that our Bond caches use less than half the memory compared to Maya meshes (from many gigabytes down to around 1GB for 100 frames of cached deformation).

We also have implemented a Shot Sculpting toolset to allow animators to layer *Bond* deformers, such as tangent space blend shapes, in a shot context.

6 LIMITATIONS AND FUTURE WORK

The rig bindings authoring workflow is still a great challenge as we rely on a whole new toolset to create deformers, paint weight maps and skin weights, all on geometry that only exists in USD.

In order to help riggers feel comfortable we have built tools to maintain a Maya copy of the USD geometry, this enables them to use off-the-shelf tools to paint skin weights. They can then convert the bindings data back to USD when done. We also rely on Maya geometry when editing blend shapes so that artists can use all the Maya modelling tools they already know.

Finally, the deformation chain authoring is currently done within a large tree view, but as our chains are growing in complexity we would like to extend our currently read-only graph user interface to allow full editing capatilities and help manage complex inter-geometry dependencies.

ACKNOWLEDGMENTS

Thanks to Miguel Gao, Stefan Feess, Valerie Bernard, Raffaele Fragapane and to all the riggers for their patience as we were developing Bond. Thanks also to Sebastian Grassia, George ElKoura and the USD team at Pixar™ for their support, and to Krystian Ligenza for his advice on the Maya integration.

REFERENCES

Aloys Baillet, Eoin Murphy, Oliver Dunn, and Miguel Gao. 2018. Forging a New Animation Pipeline with USD. In *ACM SIGGRAPH 2018 Talks (SIGGRAPH '18)*. Association for Computing Machinery, New York, NY, USA, Article Article 54, 2 pages. https://doi.org/10.1145/3214745.3214779

Rob Bateman, Eoin Murphy, Fabrice Macagno, Paul Molodowitch, and Aloys Baillet. 2019. AL_USDMaya. https://github.com/AnimalLogic/AL_USDMaya.