Achieving Real-Time Playback with Production Rigs

Andy Lin Gene S. Lee Joe Longson Jay Steele Evan Goldberg Rastko Stefanovic
Walt Disney Animation Studios *†

Fig 1: *Nitro draws entire scene in real-time*

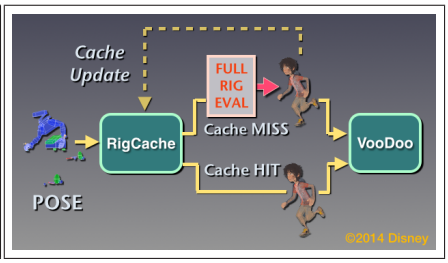

Fig 2: *RigCache queries geometry cache for pose*

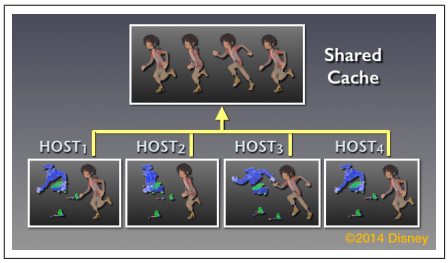

Fig 3: *Parade distributes caching to hosts*

Abstract

Rig speed is of paramount importance to animation pipelines. Real-time performance provides immediate feedback to artists thereby increasing the number of possible iterations and ultimately leading to higher quality animation. This paper presents a novel method for real-time playback of production rigs inside a host application, such as Maya, without sacrificing functionality or ease of use. Real-time performance is achieved by augmenting the host with *Nitro*, a replacement strategy for OpenGL drawing events, *RigCache*, a caching system for minimizing scene graph evaluations, and *Parade*, a distributed system for scheduling cache updates. Only minimal rig changes are required for the three tools to collectively optimize playback. The result is a seamless experience that is natural, unobtrusive, and preserves familiar work flows.

1 Real-Time Playback

Nitro Operating within a host application, Nitro provides an optimized drawing path for geometry to OpenGL. Employing many real-time rendering techniques, including visibility culling and level-of-detail, Nitro's design is functionally similar to many commercial game engines, e.g. Unreal Engine. In addition, Nitro leverages multiple CPUs and provides asynchronous CPU/GPU processing, while employing unique standard optimization techniques, e.g. support for Disney's PTex and SubD. Nitro operates by disabling the host-application's draw path via visibility flags and then commandeering the loading and displaying of geometry. It also provides a set of lightweight hooks and scripts to seamlessly integrate into the host application's drawing and selection events.

To guarantee smooth user interactions, Nitro supports an on-demand, asset loading system that streams film-quality geometry and texture data directly from disk. The loading system is largely data format agnostic and accommodates new formats easily. With its ability to operate on-demand the asset system isolates rig evaluation and other costly computations from the display event. This feature allows Nitro to process extremely large scene graphs within a host application, yielding a 10-20x improvement in playback performance (Fig 1).

*{andy.lin|gene.s.lee|joe.longson}@disney.com
†{evan.goldberg|rastko.stefanovic}@disney.com

RigCache RigCache implements a simple, yet elegant method to minimize rig evaluations. It assumes that a set of rig controls and their values define a pose, and that pose predictably maps to a set of deformed geometry. RigCache utilizes this mapping to cache the results of distinct poses. This approach is very effective and complementary to any rig evaluation method, e.g. threaded graph traversal or GPU optimizations.

For each frame redraw, RigCache computes the current pose and searches for it in the cache (Fig 2). On a cache hit, the mapped geometry is sent directly to Nitro for rendering, bypassing the need to evaluate the deformer chain for the given mesh. On a cache miss, the deformation chain is evaluated to compute the geometry for the given pose. The cache is updated with the new mapping, and then the geometry is sent directly to Nitro for rendering.

The speed of RigCache is dependent on the cost of computing the pose and the frequency of cache hits. Fortunately, most rig controls are driven by simple curves or layers. Therefore, the cost of computing the pose is relatively inexpensive compared to the cost of evaluating the full deformation chain. In practice, it is often 10 times faster. When the rig contains advanced features, such as constraints, the gains are less but still a significant improvement over a full evaluation.

Parade Parade increases the likelihood of cache hits by distributing RigCache across multiple hosts with the poses of past and future frames. Each host operates independently and in parallel to rapidly pre-populate the geometry cache (Fig 3). By the time an animator hits play or scrubs the time line, the cache is full and the resulting playback is seamless.

Parade operates by sampling frames from an animated shot and then distributing the pose of a single frame to each host. Each host uses the pose to update the controls of a base rig which contains no animation. The results are then stored in a common geometry cache. This approach is effective since RigCache assumes that the output of a rig is deterministic and derivable from its pose alone. Relying on pose data only, each host can rapidly process a succession of pose requests for any rig from any number of interactive sessions.

2 Results

Applied together, Nitro, RigCache, and Parade successfully increase the playback of slow production rigs. For both *Frozen* and *Big Hero 6*, rig speed accelerated from 3 fps to 100 fps per rig. Animators quickly embraced real-time scrubbing over time-consuming playblasts to review their motion. Each week, hundreds of artists cached millions of poses thereby increasing iteration and improving animation. They were able to animate more character rigs at once in a single shot than at any time in Disney's past.