An Efficient Spatio-Temporal Architecture for Animation Rendering

Vlastimil Havran, Cyrille Damez, Karol Myszkowski, and Hans-Peter Seidel
MPI Informatik, Saarbrücken, Germany

Producing high quality animations featuring rich object appearance and compelling lighting effects is very time consuming using traditional frame-by-frame rendering systems. In this work we present a rendering framework for computing multiple frames at once by exploiting the coherence between image samples in the temporal domain. For each sample representing a given point in the scene we update its view-dependent components for each frame and we add its contribution to pixels identified through the compensation of camera and object motion.

In this text we describe in more details two major challenges that we face in our framework: The visibility and global illumination computation in dynamic environments. Also, we discuss some standard rendering tasks such as shading and motion blur.

The **visibility** computation in our rendering framework is based on a **multi-frame visibility data structure** (MFVDS). MFVDS for a given ray provides the visibility information for all considered frames at once. A *kd*-tree data structure is used to implement MFVDS. Static objects are stored in a global *kd*-tree. Dynamic objects are instantiated for every frame. The instantiated objects are processed using a hierarchical clustering algorithm to separate them in space. For each cluster of instantiated objects a separate *kd*-tree is constructed that is inserted to a global *kd*-tree. Finally, global *kd*-tree is refined in spatial regions where *kd*-trees have been inserted. During processing of visibility queries the performance of MFVDS is improved by intensive caching of intermediate results.

A single visibility query using our data structures provides a ray intersection information for a given frame and marks those frames for which this information becomes invalid due to dynamic changes in the scene. This allows us to avoid redundant computation for each frame if the ray does not hit any instantiated object. In regions not populated by animated objects the same ray traversal cost is achieved as for completely static scenes.

The **global illumination** computation in our framework is based on a **bidirectional path tracing** (BPT) algorithm [Lafortune 1996; Veach 1997] which uses MFVDS to query visibility for all considered frames at once. Each bi-directional estimation of a given pixel color is reused for several frames before and after the one it was originally computed for. To reuse these estimates, the BRDF values at the first hit point of the eye path needs to be recomputed to take into account the new viewpoint. The corresponding estimates are then added to the pixel through which this hit point can be seen for the considered frame. Since it involves only the evaluation of direct visibility from the viewpoint and a few BRDF recomputations, reusing a sample is much faster than recomputing a new one from scratch. Reusing samples for several frames also makes the noise inherent to stochastic methods fixed in object space, which enhances the quality of the resulting animations.

Shaders add rich appearance of objects and can be efficiently computed in our animation framework. We split our shading functions into view-independent and view-dependent components, where the former is computed only once for a given sample point and the latter is recomputed for each frame. It is worth noting that in our BPT technique we need to recompute the view-dependent component only for sample points that are hit by primary rays, while for the remaining path segments shading results are just reused.

Motion blur is an important visual effect in high quality animation rendering, which is extremely easy to obtain in our framework. A fragment of the motion-compensation trajectory traversed by a given BPT sample within the camera shutter opening time is computed and projected to each frame. The sample radiance contributes to each pixel traversed by the projected trajectory roughly in a proportion to the length of traversed path. In practice, we use 2DDA algorithm for piecewise-linear approximation of this trajectory and linear interpolation of sample radiance between a pair of frames, which leads to very good visual results (refer to Figure 2).

We use **disc caching**, instead of storing in memory all frames, which makes the resolution of our animations limited only by the disc capacity. Since the computation for the subsequent frames is localized in coherent regions in the image space located along the motion-compensation trajectory, we achieve a pretty high cache-hit ratio. As a consequence, disk accesses only increase the total computation time of our method by about 10% on average.

Conclusions: The main advantage of our framework is significant speedup of animation rendering, which is usually over one order of magnitude in respect to traditional frame-by-frame rendering, while the obtained quality is always much higher due to significant reduction of flickering.

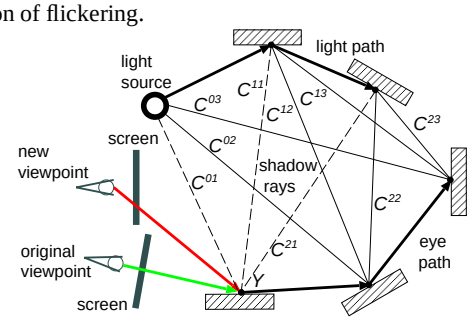

Figure 1: Bidirectional estimations for a given pixel can be used for several camera positions. Only direct visibility and the BRDF corresponding to connections C_{01}, C_{11}, and C_{21} need to be recomputed.

Figure 2: A sample animation frame featuring global illumination and motion blur.

References

LAFORTUNE, E. 1996. *Mathematical Models and Monte Carlo Algorithms*. PhD thesis, Katholieke Universiteit Leuven.

VEACH, E. 1997. *Robust Monte Carlo Methods for Light Transport Simulation*. PhD thesis, Stanford University.