Rhino-Palooza: Procedural Animation and Mesh Smoothing

Evan Goldberg Dmitriy Pinskiy
Walt Disney Animation Studios

Figure 1: (a) The rig. (b) Rolling diagram. (c) Rhino rendered in hamster ball. (d) Mesh with dents. (e) Dents removed.

In Disney's *Bolt*, the character of Rhino poses many technical challenges. He spends a majority of the movie inside a plastic ball, frequently contacts the ground surface, and presents a complicated skinning problem. Innovative tools and technology were developed to solve these issues for the production.

1 Rigging: Rolling the Hamster Ball

Rhino's hamster ball is his acting stage, a prop, and an extension of his personality. Hand animating the ball proved too tedious a task, so we developed an in-scene procedural animation solver that uses a single manipulator to interactively compute physically accurate rolling motion. We base our system on explicit integration methods to track position from initialization to the current time step. On each time delta, a velocity vector Vp is defined between the current and previous positions p_i and p_{i-1}, either from keyframe data or interactive manipulation. The cross product of Vp with the ground surface normal Vn yields a quaternion axis for rotation Vq (see Figure 1b). Angular rotation around this axis is computed in radians by dividing the velocity vector's Euclidean norm by the radius of the ball. The system can be enabled or disabled on any frame, which allows animators to provide custom rolling motion on a downstream node without disrupting the procedural animation solver.

A constraint system interconnects the Rhino and hamster ball rigs, allowing them to translate in unison, while the ball rolls as a result of the aforementioned algorithm. Surface-constrained guides let animators easily position Rhino's hands and feet along the inner contour of the ball. The enable/disable functionality of the rolling animation system permits the guides to roll with the ball or slide independently.

2 Animation: Aiming and Walk Cycles

The data collected to produce rolling animation can also be used to auto-orient Rhino in the direction of motion. On each time step, we align the character's forward axis with Vp. The remaining rotational degree of freedom around this vector is disambiguated via the ground surface normal (if none exists, the cross product of the current and previous velocity vectors is used). Additionally, if a walk cycle with known stride length is available, we correlate $\|Vp\|$ with a distance-to-time conversion to link the animation cycle speed to the character's velocity. The converse holds true as well, where blocking animation can be re-timed to match cycle speed. The result is a simple-to-use path animation tool which allows for easier animator manipulation than those that constrain motion to a spline.

evan.goldberg@disney.com, dmitriy.pinskiy@disney.com

3 Layout: Subdivision Visualization

During layout and animation, it is common to use a simplified polygonal stand-in to quickly position the character. However, when a polygonal ground plane is subdivided to the limit surface in renders, the character appears to "float" above it. Since Rhino is frequently in contact with the ground, we developed a heavily-optimized subdivision visualization tool to accurately establish ground contact and maintain interactive frame rates. An adaptive subdivision scheme determines surface resolution as a function of camera distance while back-face and view frustum culling is performed based on camera orientation. Vertex shader optimization is achieved via parallelized limit surface evaluation on the GPU and a reduction of the data sent per glVertex call.

4 Character Finaling: Dent Pulling

Animating Rhino's overweight physique often produces undesirable dents. These problematic regions produce visual artifacts that reduce the character's appeal. Traditional mesh relaxation based on iterative Laplacian smoothing (weighted averaging) is insufficient as it does not remove the dent, but rather, widens its width. Even smoothing methods that are highly sensitive to the direction of maximum curvature do not achieve the desired results because they work locally and do not account for the dent's global orientation. For regions that bend, such as Rhino's neck, a vertex may initially exhibit maximum curvature parallel to the dent, but after a few iterations, these smoothing schemes simply flatten the bent slope.

Therefore, we introduce a novel fairing approach that dynamically adapts its relaxation scheme to the dent's orientation and is based on global curvature of geodesic curves in the dent. For each dent vertex p_i, we maximize global curvature of a curve over all geodesic curves that pass through p_i:

$$\max_{p_i \in \gamma} \mathrm{K}(\gamma) = \max_{p_i \in \gamma} \int_a^b \kappa(t)\,dt$$

where γ is a geodesic curve that lies on the dented region and runs across p_i, $[a,b]$ is the parameter range of γ, K is total curvature of γ, and κ is a curvature at a particular point of γ. By maximizing the total curvature of γ, we ensure that the curve spans the dent's valley across the dent. The direction of γ serves as a reference for the dent's orientation. To pull a vertex, we still average its position with the positions of its neighboring vertices. However, unlike traditional Laplacian smoothing, our weights are based on how closely aligned the immediate neighbors are with the corresponding geodesic curve of the dent. Thus, the weights reflect the dent's overall shape, rather than local geometric particularities.