Mathematical Tricks for scalable and appealing crowds in Walt Disney Animation Studios' "Raya and the Last Dragon"

Nicolas Nghiem
Walt Disney Animation Studios
Burbank, USA

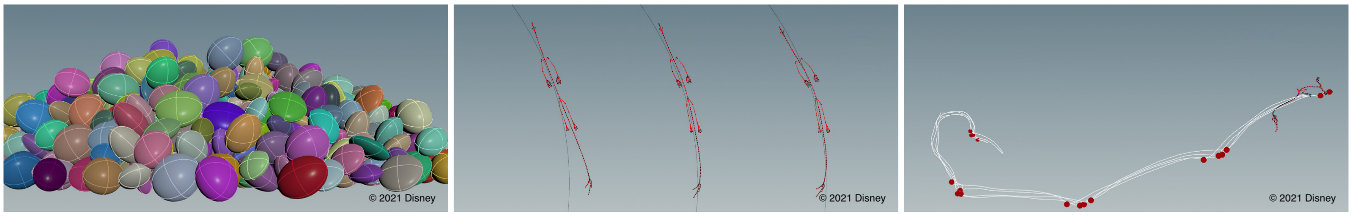

Figure 1: (Left) Anisotropic distances for PBD with ellipsoidal particles, (Center) Procedural curve fitting animation layer, (Right) Footstep detection with distance integral invariant

ABSTRACT

The crowds department had to tackle a variety of challenging shots for Walt Disney Animation Studios' "Raya and the Last Dragon" such as beetles crawling on top of each other, immense fish simulation or dragon choreography.

In order to handle this level of complexity while keeping a good amount of artistic control, we implemented some effective technical solutions such as the use of anisotropic distances in Position based Dynamics (PBD) and boids simulations, procedural animation layers for fish and dragons or distance integral invariant to detect dragon foot contacts.

CCS CONCEPTS

• **Computing methodologies → Procedural animation**; **Collision detection**; **Motion processing**.

KEYWORDS

crowds, simulation, anisotropy, procedural animation, shape analysis

ACM Reference Format:

Nicolas Nghiem. 2021. Mathematical Tricks for scalable and appealing crowds in Walt Disney Animation Studios' "Raya and the Last Dragon". In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Talks (SIGGRAPH '21 Talks), August 09-13, 2021.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3450623.3464650

1 INTRODUCTION

One of the main challenges that crowds artists face when crafting their shots is to balance scalability and appeal. There is an intrinsic tension between keeping each character's performance unique and using a toolset geared towards automation. A solution is to consistently adapt this toolset to fit precisely the sequences' requirements. Those improvements can benefit the crowds artists in every step of their crafts: fleshing the crowd as an organic entity faster, enhancing the individual performances with dedicated tools and automating annotations on their output data to help downstream departments.

We present some of the tools we implemented during production that answer to this broad range of needs: anisotropic distances in crowds simulation for global movement, procedural animation layers for detailed performance and shape analysis features for automated footstep annotation.

2 ANISOTROPY IN CROWDS SIMULATION

In Raya and the Last Dragon, we used simulation-based techniques to handle a school of fish and a swarm of beetles. Using anisotropic distances in these crowds simulation is motivated by two main aspects. In the Boids model [Reynolds 1987], the main behaviors of the agents result from their perceived distance to the rest of the crowd and not necessarily the Euclidean distance. In the algorithm developed in our department to create Ralphzilla with a PBD simulation [Byun et al. 2019], using the Euclidean distance implies a spherical representation of the particles. However, beetles are best approximated by ellipsoids as shown in Figure 2 ; applying directly our code resulted in floating beetles. We decided to use anisotropic distances in those simulations to address those issues.

As a reminder, there is a direct relation between ellipsoids and anisotropic distances. Anisotropy is expressed via a metric tensor $G \in \mathbb{R}^{3x3}$ symmetric definite positive that induces a distance $d_G(x, y) = \sqrt{(x-y)^T G(x-y)}$. The unit sphere with regard to d_G is an ellipsoid and its radius is given by $(\lambda_i^{-1/2})_i$ with λ_i the eigenvalues of G. We use the notation: $S(c, G)$ for the ellipsoid of metric G centered in c.

Computing the metric tensor. We first start by computing a proxy ellipsoid that approximates our agent. A natural way to extract a symmetric definite positive matrix is to compute the covariance matrix. We use the assumption that our asset is well oriented which

SIGGRAPH '21 Talks, August 09-13, 2021, Virtual Event, USA

ACM ISBN 978-1-4503-8373-8/21/08.
https://doi.org/10.1145/3450623.3464650

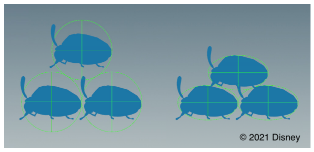

Figure 2: Comparison between spherical and ellipsoidal proxy shapes

means we only have to store the eigenvalues of the covariance matrix in a vector rather than storing the full matrix. We also leave an uniform scale parameter to the artist to allow further control.

Using anisotropy. We adapt existing algorithms by replacing the Euclidean distance with an adapted anisotropic distance $d_{G'}$.

For boids forces and detection radius, each particle follows its own metric, defined by $G'_i = O_i(t)^T G_i O_i(t)$ with $O_i(t)$ the current orientation of the particle i.

For PBD collision detection, let us first note that two ellipsoids $S_0(c_0, G_0)$, $S_1(c_1, G_1)$ collide if and only if $c_0 - c_1$ belongs to the Minkowski sum of the ellipsoids centered around the origin $S'_0(O, G_0)$, $S'_1(O, G_1)$ defined as: $S'_0 \oplus S'_1 = \{x + y | x \in S'_0, y \in S'_1\}$.

The Minkowski sum of two ellipsoids is not generally an ellipsoid. However, [Kurzhanski and Varaiya 2002] gives an interior and an exterior ellipsoidal estimators of the Minkowski sum along the direction $l = (c_0 - c_1)/\|c_0 - c_1\|$:

$$G_- = ((R_0 G_0^{-1/2} + R_1 G_1^{-1/2})^T (R_0 G_0^{-1/2} + R_1 G_1^{-1/2}))^{-1} \tag{1}$$

with R_0, R_1 two orthogonal matrices that align $R_0 l, R_1 l$ and l.

$$G_+ = ((1 + C^{-1}) G_0^{-1} + (1 + C) G_1^{-1})^{-1} \tag{2}$$

with $C = \sqrt{l^T G_0^{-1} l} / \sqrt{l^T G_1^{-1} l}$

Given G^* the chosen estimator, S_0, S_1 collide if $d_{G^*}(c_0, c_1) < 1$ which defines our PBD constraint.

Discussion. This approach introduces limited changes to our existing code, which allowed us to leverage it quickly for production need. Indeed, it only requires to store the particle scale as a vector rather than a float and call a different distance function which involves just a few more $3x3$ matrix multiplications. Using a PBD simulation is advantageous as it is fast -between 14 and 34 fps in our shots with our OpenCL implementation in Houdini- and intuitive to use, allowing the artist to perform more iterations and have more control. While our approach worked satisfyingly in production, we plan in future work to compare it to other methods for ellipsoid collision described in [Jia et al. 2011] or [Müller and Chentanez 2011] in terms of speed and accuracy.

3 PROCEDURAL ANIMATION LAYERS

Once the global movement of the crowd is set, we used procedural animation techniques to enhance individual performances. We work within our Skeleton Library [El-Ali et al. 2016], which lets us add animation layers on top of the existing cycle animation. We store our animation layers as quaternions, and apply them during rig evaluation on top of the base animation. We also use a blend parameter for weighting the animation layer with spherical interpolation (see Figure 1). This blend parameter can be set manually by the artist or driven by attributes such as the speed of the agent for an extra artistic control.

Cycle Amplification. One of those animation layers amplifies the difference between the current pose and a given reference pose. For each desired joint, we compute the quaternion that aligns the current to the reference local transform. We give controls to the artist regarding the reference pose: it can either be the rest pose, a crafted pose, a barycenter of all the cycle poses computed with weighted spherical interpolation or even the cycle itself offsetted in time. This approach was particularly efficient when applied to the fish's spine: amplifying the cycle during high acceleration phases helped getting a more believable motion.

Curve Fitting. In some shots, particularly in the fish and dragon sequences, constraining our particle to a curved trajectory and reading the cycle as is resulted in a stiff motion, breaking the flow of the choreography. To address this issue, we created a curve fitting animation layer: we compute and store the rotations necessary to align the spine of the character with the curve of its trajectory.

4 DETECTING DRAGONS' FOOTSTEPS

For the dragon sequence, the Crowds department had to share data with the Effects department to create magic ripples when the dragons land their feet. We proposed a tool that could, given crowds particles, trails the motion of each foot and automatically labels the given curves with an attribute $footContact$. The labelling process was challenging as the dragons were running in the air and could change their orientation over time. Moreover, they could sometimes blend with a glide cycle and we did not want to detect any foot contact in those periods.

We used the fact that foot contacts correspond to saliencies along the trajectory. We computed two features: the local curvature and the distance integral invariant, used in shape analysis [Manay et al. 2006], and study their local maxima to annotate the curves. Furthermore, we left a threshold parameter so that the artists can filter more or less aggressively the labels detected during glide phases. An example result can be seen in Figure 1.

REFERENCES

Dong Joo Byun, Alberto Luceño Ros, Alexander Moaveni, Marc Bryant, Joyce Le Tong, and Moe El-Ali. 2019. Creating Ralphzilla: Moshpit, Skeleton Library and Automation Framework. In *ACM SIGGRAPH 2019 Talks* (Los Angeles, California) *(SIGGRAPH '19)*. Association for Computing Machinery, New York, NY, USA, Article 66, 2 pages. https://doi.org/10.1145/3306307.3328179

Moe El-Ali, Le Tong, Josh Richards, Tuan Nguyen, Alberto Luceno Ros, and Norman Moses Joseph. 2016. Zootopia Crowd Pipeline. In *ACM SIGGRAPH 2016 Talks* (Anaheim, California) *(SIGGRAPH '16)*. Association for Computing Machinery, New York, NY, USA, Article 59, 2 pages. https://doi.org/10.1145/2897839.2927467

Xiaohong Jia, Yi-King Choi, Bernard Mourrain, and Wenping Wang. 2011. An algebraic approach to continuous collision detection for ellipsoids. *Computer Aided Geometric Design* 28, 3 (2011), 164–176. https://doi.org/10.1016/j.cagd.2011.01.004

A.B. Kurzhanski and P. Varaiya. 2002. On Ellipsoidal Techniques for Reachability Analysis. Part II: Internal Approximations Box-valued Constraints. *Optimization Methods and Software* 17, 2 (2002), 207–237. https://doi.org/10.1080/1055678021000012435

Siddharth Manay, Daniel Cremers, Byung-Woo Hong, Anthony Yezzi, and Stefano Soatto. 2006. *Integral Invariants and Shape Matching.* Birkhäuser Boston, Boston, MA, 137–166. https://doi.org/10.1007/0-8176-4481-4_6

Matthias Müller and Nuttapong Chentanez. 2011. Solid Simulation with Oriented Particles. *ACM Trans. Graph.* 30 (07 2011), 92. https://doi.org/10.1145/2010324.1964987

Craig W. Reynolds. 1987. Flocks, Herds and Schools: A Distributed Behavioral Model. *SIGGRAPH Comput. Graph.* 21, 4 (Aug. 1987), 25–34. https://doi.org/10.1145/37402.37406