Artist-Driven Crowd Authoring Tools

Damien Maupu
Double Negative
dmu@dneg.com

Emanuele Goffredo
Double Negative
eg@dneg.com

Nile Hylton
Double Negative
neh@dneg.com

Mungo Pay
Double Negative
mungo@dneg.com

Martin Pražák
Double Negative
map@dneg.com

Figure 1: Background crowds in *Assassin's Creed* were produced using built-in and custom Houdini crowd nodes: scans on the left and finals on the right. Stills from 'ASSASSIN'S CREED' courtesy of Twentieth Century Fox.

ABSTRACT

While crowd simulation frameworks can be very powerful for virtual crowd generation, in a VFX context they can also be unwieldy due to their chaotic nature. Small changes on the inputs can produce markedly different results, which can be problematic when attempting to adhere to a director's vision. Artist driven tools allow much more flexibility when constructing scenes, speed up turnaround time and can produce extremely dynamic crowd shots. To generate virtual crowds, Double Negative VFX (Dneg) has recently transitioned from an in-house standalone simulation-based solution to an artist-driven framework integrated into SideFX's Houdini.

CCS CONCEPTS

•**Computing methodologies** →**Animation;** *Procedural animation;* Mesh geometry models;

SIGGRAPH '17 Talks, Los Angeles, CA, USA
 978-1-4503-5008-2/17/07…$15.00
DOI: http://dx.doi.org/10.1145/3084363.3085035

KEYWORDS

Crowd Simulation, Animation

ACM Reference format:
Damien Maupu, Emanuele Goffredo, Nile Hylton, Mungo Pay, and Martin Pražák. 2017. Artist-Driven Crowd Authoring Tools. In *Proceedings of SIGGRAPH '17 Talks, Los Angeles, CA, USA, July 30 - August 03, 2017,* 2 pages.
DOI: http://dx.doi.org/10.1145/3084363.3085035

1 SHARE THE BURDEN

Traditionally, the solution used in Dneg for crowds was built as a standalone application, fully developed and supported by a dedicated team in the R&D department. While providing a set of tailor-made solutions targeting specific production needs, this approach imposed a large burden for developers, both implementing new features, and maintaining a large codebase.

By transitioning to an off-the-shelf solution (SideFX's Houdini), the development and support burden was split between teams:

- **SideFx** provides a simulation framework and a set of versatile crowd data structures, with features such as ragdolls, fuzzy logic and other dynamics solver interaction, available out of the box.

- **R&D Crowd Team** is responsible for low-level integration of Houdini Crowds into the pipeline (i.e. interfacing with custom crowd rigs, cache files and animation formats), exposing published data via a tailored interface, building a comprehensive toolset for crowd tasks, and focusing on a novel set of tools.
- **Pipeline Teams** provide pipeline integration for asset management (e.g. environment and cameras via an in-house scene description format) and crowd cache publishing.
- Finally, **Production-side Technical Artists** leverage the modular nature of the software and its many scripting languages to prototype new ideas and develop new workflows (e.g. importing in-house look description for playblasts, custom crowd dynamics behaviours, integration with cloth or water simulation).

We found out that this toolset is friendly to junior artists: it is easy to pick up, and offers them an opportunity to broaden their skills. Senior artists can easily extend the framework's functionality and quickly establish best practices.

As for development, training for crowd tools now involves documentation from several different sources. Houdini comes with a comprehensive set of documentation for simulation nodes, while Dneg's custom crowd nodes are all showcased in a single package of example scenes. These scenes have proven to be easy to update and distribute, offering a natural step-by-step guide for new users.

2 CHANGE THE PARADIGM

The traditional approach to virtual crowds is particle-driven simulation – each agent is given a set of functions to evaluate its environment (e.g. attraction point, obstacle distance) and a set of rules to react to these (e.g. adapt speed/direction, transition between motion clips). The simulation then performs a per-frame computation, where the state of an agent in frame *n* is based on its state on frame *n-1*. For this reason, scene creation can be time-consuming, as an artist must rerun the simulation from the start after each parameter change, before they can inspect the result. Moreover, crowd simulations are notorious for lacking fine control, with small changes of a single agent often having dramatic effects on the final output.

In VFX, it is common to layout vignettes (loopable animation clips) to achieve behaviours that would be difficult to generate with a simulation framework (e.g. complex multi-character interaction). Coupled with geometry variation, terrain adaptation and time warping, vignetting is a powerful, simple and interactive way to enrich virtual environments.

We have extended these existing frameworks by implementing a set of space-time optimisation methods, as described in recent data-driven animation publications (see Section 3). With similar workflows to standard vignetting frameworks, these methods allow users to alter animations, while still offering immediate visual feedback.

Houdini is available in two license types. While our vignetting tools are designed to work well in the standard edition, crowd simulation is only available in the more advanced (and costly) edition. We believe the simulation and the editable vignetting framework are complementary – the decision of which tool to use should be up to the artist, based on the shot type and crowd requirements.

3 EDITABLE VIGNETTES

We have developed a set of custom Houdini nodes to allow intuitive manipulation of the agent trajectories. These are usable as stand-alone tools, acting on single vignettes, or as part of a more comprehensive framework allowing artists to directly alter the output of a crowd simulation.

Inspired by the production needs of current shows, we have implemented the following techniques:

- **Follow path** allows artists to adapt trajectories of a group of agents to a target curve. For each agent, the curve is offset along its normals by the projective distance from the initial position of the agent to the curve. After several smoothing operations, the final position of an agent is computed by mapping the accumulated distance of the agent along the original trajectory to the target trajectory. Additional curve modifications are then applied on top to allow an agent to avoid obstacles. This method has proven to be best suited for complex environments populated by locomoting agents.
- **Edit trajectory** uses a set of agent trajectories and a desired target curve as its inputs, and outputs new agent trajectories similar to the follow path method. The input curve is sampled in space, retaining position and orientation for each sample. Using an inverse distance weighting interpolation scheme, the orientation at any point in space can be computed constructing a warped coordinate-space. By accumulating reoriented incremental displacements of each agent, an agent's trajectory can be mapped to this deformed space. Compared to the Follow Path tool, this method extends to more complex and non-repeating motions (characters turning, stopping, interacting with environment).
- **Group motion editing** tool is an implementation of group motion editing approach [Kwon et al. 2008]. Its core principle is to construct a mesh from the input agent trajectories, and deform it using a laplacian surface-editing technique which minimizes distortion while preserving local features. The mesh is constructed by connecting each agent's position to its previous/next positions and/or surrounding agents' positions; this captures temporal, spatial and spatiotemporal features of the original trajectories. The user can define constraints by pinning down part of the trajectories in either space, time or both.
- **Motion transitions** tool provides the ability for agents to change their motion state (walk, run, idle) as an easy way to build crowd behaviours. A motion transition is generated by blending between two motion states at a point in time. By analyzing an agent trajectory, we can detect at which frame that agent is within the trigger region of a point in space. Good blends are produced where the poses between the input motions are similar. We use a pose metric to find such frames, and build a motion graph [Kovar et al. 2002] to identify parts of an animation that can loop, either to generate new vignettes or to allow reducing/extending the length of existing ones.

REFERENCES

Lucas Kovar, Michael Gleicher, and Frédéric Pighin. 2002. Motion Graphs. In *ACM SIGGRAPH 2002 Papers (SIGGRAPH '02)*. ACM, New York, NY, USA, 473–482.

Taesoo Kwon, Kang Hoon Lee, Jehee Lee, and Shigeo Takahashi. 2008. Group Motion Editing. In *ACM SIGGRAPH 2008 Papers (SIGGRAPH '08)*. ACM, New York, NY, USA, Article 80, 8 pages.