From Procedural Panda-monium to Fast Vectorized Execution using PCF Crowd Primitives

Aaron Lo
alo@pixar.com
Pixar Animation Studios
Emeryville, USA

Venkateswaran Krishna
venkat@pixar.com
Pixar Animation Studios
Emeryville, USA

J.D. Northrup
jdn@pixar.com
Pixar Animation Studios
Emeryville, USA

Mark Hessler
mhessler@pixar.com
Pixar Animation Studios
Emeryville, USA

Arnold Moon
amoon@pixar.com
Pixar Animation Studios
Emeryville, USA

Michael Lorenzen
lorenzen@pixar.com
Pixar Animation Studios
Emeryville, USA

Jonah Laird
bluestr@pixar.com
Pixar Animation Studios
Emeryville, USA

Paul Kanyuk
pkanyuk@pixar.com
Pixar Animation Studios
Emeryville, USA

Figure 1: A Crowd of Fantastical Red Pandas Created with a Pcf Crowds in *Turning Red* ©Pixar.

ABSTRACT

In animation and VFX, crowds are too often considered an "edge case", to be handled by specialized pipeline outside the main workflows. Requirements of scale and traditional reliance on history based simulation have been obstacles to properly building crowd systems into the core functionality of digital content creation software. Pixar's crowds team has worked to reverse this trend, developing a fast vectorized crowd system directly within the execution engine of our proprietary animation software, Presto. Dubbed Pcf, for Presto Crowds Framework, this system uses aggregate models, called crowd primitives, to provide artists directly manipulable crowds while maintaining proceduralism for mass edits. Like traditional models, they contain rigs (a graph of operators) which run parallelized through Presto's execution engine [Watt et al. 2014], but rather than posing points, they set joint angles and blendshape weights to pose entire crowds. The core operations of crowd artists: placement, casting, clip sequencing, transitions, look-ats, and curve following, are all well expressed as rigging operators (known as "actions" in Presto parlance) in Pcf. They provide interactive control of entire crowds in context using the same animation tool as our layout artists, animators, simulation TDs, etc. The first film to use Pcf, *Turning Red*, reaped massive benefits by building a stadium's worth of characters in a fraction of the time of previous films' efforts. However, because Pcf is tightly integrated into Presto, the benefits extended beyond efficiency for the crowds team. By providing our layout department Pcf rigging controls, they were able to shoot inside the crowd and use procedural operators to clear room for the camera and maintain crowd density only where needed. Similarly, the principal animation team could animate main characters in context of the crowd they were acting in, providing the proper context which all too often is absent in crowd shots. Taken together Pcf, is a huge step forward in bringing crowds out of the margins and into the core of animation workflows at Pixar, demonstrating that fast vectorized crowds can be an integral part of digital content creation software.

SIGGRAPH '22 Talks, August 07-11, 2022, Vancouver, BC, Canada

ACM ISBN 978-1-4503-9371-3/22/08.
https://doi.org/10.1145/3532836.3536256

ACM Reference Format:
Aaron Lo, Venkateswaran Krishna, J.D. Northrup, Mark Hessler, Arnold Moon, Michael Lorenzen, Jonah Laird, and Paul Kanyuk. 2022. From Procedural Panda-monium to Fast Vectorized Execution using PCF Crowd Primitives. In *Special Interest Group on Computer Graphics and Interactive*

Techniques Conference Talks (SIGGRAPH '22 Talks), August 07-11, 2022. ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3532836.3536256

1 BACKGROUND

To support fast iteration and computation of frames in parallel, the Presto animation system executes rigs in a history-free way. In other words, we can't guarantee the order in which frames are computed, so simulation is not supported. For a crowds system, this results in a design with interesting tradeoffs.

The Pcf framework computes crowds in three phases. First, we compute the "rest state" or the crowd including the location, casting, orientation, and other fixed distributions that don't need to vary over time. Next, we compute the clip sequence and resulting per-frame root motion deltas for each agent. These first two phases are done in a batch computation for the entire frame range of the shot, and cached in a way that allows subsequent frames to re-use this information. Finally, we compute data like the actual joint orientations, ground contact, orientation constraints, path trajectory, etc. for single frames as needed.

Some benefits of this are the user can view any frame in the shot, update parameters of the crowd setup, and see results immediately without requiring an expensive re-simulation. Similarly, because motion is defined in terms of kinematic path-following, changes to the input don't result in potentially large look differences in the outcome. The benefits of "no-sim" crowds seem to be well-known to studios in practice [Arumugam et al. 2013][Hood 2014][El-Ali et al. 2016][Maupu et al. 2017], but we mention this aspect of our system to distinguish it from the more common simulation-based crowds frameworks [Massive Software [n.d.]][Golaem Crowd [n.d.]].

For setups requiring complex agent interaction or dynamic motion, we do rely on Houdini's native Crowd system [SideFX [n.d.]], and import the results back into Presto through UsdSkel caches.

2 IMPLEMENTATION

In Pcf, our new crowd prim is a single composite model that can have a procedural network of associated actions and represent an entire crowd of agents.

Agent properties in Pcf are aggregated into an attribute pack inspired from Houdini's geometry detail structure. A pack can be thought of as a dictionary of vectorized arrays (each array having all the agent values for a single property). The widespread usage of spans removes a lot of the complexity of operating on vectorized arrays and makes it feel akin to an array of arrays. Pcf has vectorized actions which are multithreaded wherever possible for high performance. There is a reader/writer pattern for attributes allowing for thread safe read-only or write access after uniquification as needed. Pcf also has a well established schema pattern which reduces a lot of boilerplate. The order of a span of attribute values is configurable in Pcf. This is invaluable in many situations, such as where the joint order changes. Pcf also has aliasing allowing for convenient small tokens to be used as keys instead of unwieldy hierarchy paths.

In Pcf adding agents only involves modification of values. No prims, relationships, or attributes, need to be created and the execution network is unchanged, so agents can be added within execution engine (cached) computations. This opens up possibilities like a rig where artists manipulate a curve, and agents are procedurally spawned and animated walking down the length of the curve, allowing for fast population of cityscapes. Sparse overrides of attribute values are tricky but can be performed by late firing actions in the rig. Interactive agent transform nudges, agent promotion/deactivation, etc, are some examples of such actions that have been successfully deployed.

3 USE CASES

For crowds artists on *Turning Red*, Pcf was successfully utilized to create stadium sized crowds without leaving Presto. Workflows that could only run interactively for hundreds of agents could now operate on thousands. Beyond speed though, utilizing Presto's composition framework (shared by the opensource USD project), which allowed Pcf crowdPrims to be shared between hundreds of shots with sparse edits only added in specific shots where necessary. This applied as well to crowdPrims backed by imported Houdini simulations, which could be dressed modularly by the layout department as vignettes or retimed and even recast by crowd TDs per shot. When the existing Pcf procedural operators proved insufficient to hit a particular note, artists could use a python action to script ad hoc behaviors. Technical artists found this ability freeing and were able to script themselves one-off effects that otherwise would never be available in a common codebase. The success of these workflows at scale on *Turning Red* have now lead crowd artists to use Pcf for smaller and smaller groups of character, proving that the value of a well integrated crowd system goes beyond just the largest shots.

4 FUTURE WORK

Despite the success of Pcf on *Turning Red*, much work still remains to fully realize the technology's potential. While CPU execution speed is excellent, bottlenecks in dispatching the data to the GPU remain the limiting factor in performance and remains a work in progress. Also, despite the "no-sim" underpinnings of Pcf, collision resolution, navigation, motion synthesis have important use cases that we hope to approximate while staying true to the workflow requirements.

REFERENCES

Hemagiri Arumugam, Michael Frederickson, and J. D. Northrup. 2013. Crowds at Monsters University. In *ACM SIGGRAPH 2013 Talks* (Anaheim, California) *(SIGGRAPH '13)*. Association for Computing Machinery, New York, NY, USA, Article 51, 1 pages. https://doi.org/10.1145/2504459.2504523

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

Golaem Crowd. [n.d.]. Company Website: http://golaem.com.

John Hood. 2014. CrAM: Artist-Friendly Crowds on "Edge of Tomorrow". In *ACM SIGGRAPH 2014 Talks* (Vancouver, Canada) *(SIGGRAPH '14)*. Association for Computing Machinery, New York, NY, USA, Article 5, 1 pages. https://doi.org/10.1145/2614106.2614204

Massive Software. [n.d.]. Company Website: http://www.massivesoftware.com.

Damien Maupu, Emanuele Goffredo, Nile Hylton, Mungo Pay, and Martin Pražák. 2017. Artist-Driven Crowd Authoring Tools. In *ACM SIGGRAPH 2017 Talks* (Los Angeles, California) *(SIGGRAPH '17)*. Association for Computing Machinery, New York, NY, USA, Article 69, 2 pages. https://doi.org/10.1145/3084363.3085035

SideFX. [n.d.].

Martin Watt, Erwin Coumans, George ElKoura, Ronald Henderson, Manuel Kraemer, Jeff Lait, and James Reinders. 2014. *Multithreading for Visual Effects* (1st ed.). A. K. Peters, Ltd., Natick, MA, USA.