Research report

Mastering the Multitude
A comparative analysis of the USD and Alembic workflows in crowd simulation

FROUIN Carl
ESMA Toulouse
2025-2026
Under the supervision of Xavier ALIOT

TABLE OF CONTENTS

I: “INTRODUCTION” **4**
1.1 - The industry facing the data explosion **4**
1.2 - Alembic, a standard reaching its limits **4**
1.3 - Houdini and USD, a new field for experimentation **4**

II: “ACADEMIC CONTEXT AND QUESTIONS” **6**
2.1 - The challenge of the multitude **6**
2.2 - Alembic vs USD, a technological break **6**
2.3 - Technology serving artistic freedom **6**

III: “STATE OF THE ART” **7**
3.1 - The limits of traditional crowd solutions **7**
3.2 - Houdini, the procedural approach **7**
3.3 - USD Solaris, towards a non-destructive pipeline **8**

IV: “HYPOTHESES AND OBJECTIVES” **9**
4.1 - Technical hypothesis, reducing data volume **9**
4.2 - Artistic hypothesis, preserving the freedom to iterate **9**

V: “METHODOLOGY AND PROTOCOL” **10**
5.1 - The production environment, a small school studio **10**
5.2 - The Tractor farm, 272 PCs and its constraints **10**
5.3 - A protocol adapted to production constraints **10**
5.4 - From a simple case to the real one, a step-by-step approach **11**

VI: “DESCRIPTION OF THE RESEARCH” **12**
6.1 - Starting from a cube, validating each block before adding complexity **12**
6.2 - Extending to the crowd, five variants and four levels of detail **12**
6.3 - The animation cache, where USD shows its full power **13**
6.4 - Variants, LOD and camera culling, full control in LOPs **14**

VII: “RESULTS” **15**
7.1 - Single character export, a reduction factor of 122 ..... **15**
7.2 - Export time, from 2 minutes 43 to 5 seconds **16**
7.3 - Geometry of the five variants, 157 MB against 50 MB **16**
7.4 - 400-agent crowd cache, 138 MB against 900 GB **17**

VIII: “DISCUSSION, LIMITATIONS AND CONCLUSION” **18**
8.1 - From the cube to 400 robots, the gap between theory and reality **18**
8.2 - The practical limits of the USD workflow in production **18**
8.3 - A standard for the future, and a confirmed direction **19**

IX: “BIBLIOGRAPHY” **20**

ACKNOWLEDGEMENTS

First of all, I would like to thank my supervisor, Xavier Aliot, for the guidance provided throughout this project. Combining lightness and rigour, a relaxed outlook and precision, Xavier was a guide whose advice spoke to everyone. That direction allowed me to move forward with confidence and the perspective I needed.

My thanks also go to Arthur Justeau, a friend from day one, with whom I have shared this adventure since the start of our studies. Together we learned, progressed and took on the most demanding technical challenges of this film. Our mutual support was decisive, especially on the USD animation issues that formed the core of this research. Moving forward hand in hand on such an ambitious project is a privilege I fully appreciate.

I: “INTRODUCTION”

1.1 - The industry facing the data explosion

The visual effects and animation industry is going through a period of unprecedented technological change. As audiences demand ever larger, denser and more detailed worlds, studios face a major challenge, managing the explosion of digital data. At the heart of this problem lies crowd simulation, one of the most technically demanding fields. Whether it means populating a city, a stadium or a concert crowd of robots, the need to simulate hundreds or even thousands of autonomous agents pushes pipelines to their limits.

1.2 - Alembic, a standard reaching its limits

For more than a decade, the Alembic format (.abc) reigned supreme for exchanging geometric data. However, its "linear" and destructive nature, where every change often forces a full re-export of the scene, becomes a major bottleneck during intensive production phases. Faced with this rigidity, the emergence of USD (Universal Scene Description), originally designed by Pixar, offers a considerable shift. More than a simple file format, USD is a non-destructive data structure that allows simultaneous collaboration between departments and granular management of artistic iterations.

1.3 - Houdini and USD, a new field for experimentation

In this context, SideFX Houdini has established itself as the ideal testing ground. Thanks to its procedural architecture and its native USD integration through the Solaris environment (LOPs), it offers cutting-edge tools for building complex crowds. The challenge is no longer just to generate a large number of dynamic agents, but to be able to modify, light and integrate them smoothly, without the technology getting in the way of the original artistic vision.

This research report sets out to explore the limits of conventional methods against the new possibilities offered by USD. We will seek to understand how this workflow changes the management of massive data, and whether it truly gives artists more flexibility to iterate. Through a series of technical experiments, from a simple cube to a simulation of dynamic robots, we will analyse whether USD is the definitive answer to today's production challenges.

II: “ACADEMIC CONTEXT AND QUESTIONS”

2.1 - The challenge of the multitude

In today's productions, crowd simulation has become the norm. Moving from background agents to hundreds of "hero" characters radically transforms pipelines. My graduation project aims to animate more than 500 dynamic robots with credible visual and physical autonomy.

2.2 - Alembic vs USD, a technological break

Alembic works as "baked" geometry: the animation is frozen. Between Maya and Houdini, this rigidity forces massive re-exports at the slightest change, weighing down the pipeline. USD, on the other hand, uses a layered structure. By separating animation from rigging, it allows assets to be modified without re-exporting simulations, making the pipeline non-destructive and instantaneous.

2.3 - Technology serving artistic freedom

The central question remains creative freedom: is technology a hindrance? An efficient artistic workflow requires that a requested change can be viewed quickly. If the pipeline imposes hours of exporting, iteration dies. The aim is to check whether the USD ecosystem in Houdini can keep this "flow" despite the technical complexity, so that energy goes into staging and the artistic side rather than into managing disk caches.

With this in mind, this report is built around the following research question: “Does the USD crowd workflow in Houdini outperform traditional methods in terms of iteration flexibility and data management?”

III: “STATE OF THE ART”

3.1 - The limits of traditional crowd solutions

The industry standard (Golaem, Miarmy in Maya) relies on "baking": the simulation is converted into static geometry caches (Alembic). This process freezes the data and breaks the iteration cycle.

- Data explosion: the study by Bhalerao et al. (2024) shows that linear storage (.abc) leads to exponential growth in disk usage. Alembic "bakes" every vertex or transformation per frame, creating massive redundancy and saturating storage.
- Bottleneck: according to Lallier (Pixar, 2018), read and write times become the main obstacle in production. Any artistic change (an agent's speed, a trajectory) forces a full recalculation of the cache, which kills the responsiveness needed for supervision feedback.
- "Black box" effect: the NVIDIA Omniverse documentation stresses that proprietary formats limit interoperability. Once exported, the crowd loses its granularity, and intelligent agents become simple passive meshes, making last-minute adjustments technically impossible without a costly return to the source software.

3.2 - Houdini, the procedural approach

SideFX Houdini offers an alternative through its Agent architecture, where the crowd is no longer geometry but a stream of data.

- Memory optimisation: unlike polygon export, Houdini uses instances that point to shared animation clips. The SideFX documentation (2024) confirms that this method minimises the RAM footprint, making it possible to handle dense crowds in real time in the viewport.
- Attribute control: VOPs and Wrangles make it possible to drive the simulation through scripts or nodes (behaviour triggers). This flexibility is crucial, as it isolates animation data from collision or appearance data, preparing the ground for a modular scene structure.

3.3 - USD Solaris, towards a non-destructive pipeline

Integrating USD in Solaris (LOPs) replaces static files with a dynamic, hierarchical scene structure.

- Instancing and Variants (NVIDIA, 2023): USD loads only one source geometry, and only the transformation matrices change. Variants also make it possible to change a robot's design (colours, accessories) without loading new files, drastically optimising rendering.
- Layering and Composition Arcs: the major advantage is the separation of departments. Animation (SkelAnimation) lives on one layer, while lookdev lives on another. Changing a texture requires no re-export: a simple override on the stage is enough to update it. Every scene referencing the character gets the change instantly.
- Validation with usdview: this Pixar tool makes it possible to inspect the hierarchy and the layers' "opinions" in real time. It guarantees clean data before rendering, avoiding the hierarchy errors that are common when assets move between departments.

Key resources and references

- Academic research: Bhalerao et al. (2024), Scalable Pipelines for Complex Scene Assemblage: https://dl.acm.org/doi/full/10.1145/3721241.3733995
- USD fundamentals: Pixar Graphics Technologies, USD Introduction & Composition: https://openusd.org/release/index.html
- NVIDIA optimisation: NVIDIA Developer, OpenUSD Workflows and Performance: https://developer.nvidia.com/openusd
- Technical documentation: SideFX Houdini, Crowd Agents & Solaris LOPs: https://www.sidefx.com/docs/houdini/crowds/index.html

IV: “HYPOTHESES AND OBJECTIVES”

Based on the state of the art and the production context, two main hypotheses structure this research.

4.1 - Technical hypothesis, reducing data volume

The first hypothesis, a technical one, states that the USD workflow produces significantly less data than Alembic for equivalent animation, thanks to the separation of the geometry, rig and animation layers.

4.2 - Artistic hypothesis, preserving the freedom to iterate

The second hypothesis, artistic and organisational, holds that this non-destructive architecture reduces iteration times and preserves the teams' creative freedom, by allowing changes without a full re-export of the caches.

The goal is to confirm or refute these two hypotheses through reproducible experiments, carried out within the real production of the short film Mortal Koil, on a crowd of more than 500 animated robots.

V: “METHODOLOGY AND PROTOCOL”

This research relies on a comparative experimental methodology, rooted in a real production context. It combines quantitative technical measurements (export time, file size, data granularity) with a qualitative evaluation of the pipeline (iteration fluidity, readability of the hierarchy, independence of departments).

5.1 - The production environment, a small school studio

The experiments were carried out in a school setting that works like a small production studio. The school has a render farm of 272 PCs managed by Tractor 2.4, of which 190 to 200 machines are active and available on average, the others being switched off or unavailable for licensing or maintenance reasons. Each machine has at least 16 GB of DDR4 RAM and a 12-core processor, with the most powerful configurations reaching 32 GB of RAM and 32 cores. RenderMan licences are allocated dynamically according to the number of PCs available on the farm, which ensures optimal use of resources with no bottleneck on that side.

5.2 - The Tractor farm, 272 PCs and its constraints

Tractor 2.4 is an old version of the farm management software, which caused some limitations in the machine communication and control features. This did not directly affect the research results, but it was a real comfort constraint in daily production.

5.3 - A protocol adapted to production constraints

In terms of performance, an average farm PC takes about 10 minutes to load the 400-agent crowd into RAM, then about 15 minutes to render a full frame. The main bottleneck therefore lies not in the rendering itself but in the phase of loading the USD data into RAM, a reality directly linked to the minimum configurations of the available machines.

5.4 - From a simple case to the real one, a step-by-step approach

The initial protocol planned a progressive, scalable comparison between the two approaches: exporting 10, 50 and then 400 animated agents from Maya via Alembic, then repeating the same exercise in Houdini via USD, to establish a clear comparison curve at each level of complexity.

However, the real constraints of production, a tight schedule, render deadlines and the immediate needs of the film Mortal Koil, forced a change of protocol. This adaptation is in itself an important methodological finding: in a real production context, the ideal approach often gives way to a more practical one, built along the way through successive iterations rather than through a fixed protocol.

The experiment therefore unfolded naturally, by testing USD exports organised in separate layers (geometry, rigging, animation), assembling complete crowd agents, and gradually pushing complexity up to real production conditions. A direct comparison with equivalent Alembic exports was then made on several test cases, from a single character to a multi-agent crowd simulation, over frame ranges representative of the film.

The chosen indicators are technical (export time, file size, granularity per layer) and qualitative (ability to iterate without a global re-export, pipeline stability, readability of the hierarchy for the whole team).

VI: “DESCRIPTION OF THE RESEARCH”

6.1 - Starting from a cube, validating each block before adding complexity

The experiment began with a deliberately minimal approach. Before touching any of the film's characters, the first tests were run on basic geometry: a cube, then two cubes, then a dozen simple objects. This step-by-step logic is fundamental, as it validates each building block of the system in isolation, without the complexity of the asset hiding a possible structural error. This principle, going back to the simplest case as soon as something does not work, guided the whole process and consistently proved more effective than trying to debug a complex scene.

Once the behaviour of USD layers was understood and validated on these simple cases, complexity was increased gradually: the robot's hand, then its torso, then the full character with its 2 million polygons. At each step the same protocol was applied: export the three layers separately, assemble them in LOPs, validate visually, then move on to the next step only if the result was clean.

It was during this phase that a key observation emerged: the animation layer, which contains no geometry, stores only joint transformation data. Its size is therefore decoupled from the complexity of the mesh. This property radically changes the logic of iteration: fixing an animation or updating a walk cycle only requires re-exporting that single file, while the geometry and rig remain intact and shared across every shot of the film.

6.2 - Extending to the crowd, five variants and four levels of detail

Building on this validation on a single character, the protocol was extended to real crowd simulation conditions. The Mortal Koil crowd is based on five robot variants (crowdVar01 to crowdVar05), each with four levels of detail (LOD0 to LOD3). This architecture was fully exported to USD from Maya, producing two separate files for each variant: a modeling layer and a rigging layer.

The sizes of the USD modeling layers are as follows: crowdVar01 weighs 13.4 MB, crowdVar02 8.9 MB, crowdVar03 10.5 MB, crowdVar04 8.1 MB and crowdVar05 9.5 MB, a total of about 50.5 MB for the geometry of all five variants. This lightness comes from an important architectural decision: during the USD export, only the raw geometry is written. No custom attribute, no additional primvar and no simulation data weigh these files down. All of this information is handled separately, in other dedicated layers, which keeps the geometry files as light and reusable as possible.

The rigging layers, which contain only the joint and geometry relationships, are particularly light: from 278 KB for crowdVar02 to 893 KB for crowdVar01, for a total under 2.3 MB. By comparison, the source Maya scenes of these same characters exceed 300 MB per .ma file on average, without even including an exportable animation cache.

The same comparison in Alembic reveals significant differences on geometry alone, before animation is even included: crowdVar01 reaches 33.8 MB, crowdVar02 30.5 MB, crowdVar03 36 MB, crowdVar04 26.2 MB and crowdVar05 31.3 MB, a total of 157.8 MB for static geometry only, against 50.5 MB in USD.

6.3 - The animation cache, where USD shows its full power

The next step was to export the animation cache of the full crowd from Houdini SOPs. This cache, generated for 400 agents animated over 70 frames, contains no geometry: it stores only the SkelAnimation data of each agent, that is, the joint transformations frame by frame, along with a set of metadata including the agent's identifier, its birth state and more than forty custom primvar attributes. These per-agent attributes encode information that the shading and lighting departments can use downstream: paint variation, colour, variant index, wear state and other procedural parameters specific to the film's art direction.

The total size of this cache for 400 agents over 70 frames is 138.6 MB. By comparison, extrapolating a standard Alembic export for the same number of agents, using the ratio observed on the single character, leads to an estimated cache of several hundred gigabytes, which would make this workflow unthinkable in a production context with limited shared network storage.

6.4 - Variants, LOD and camera culling, full control in LOPs

Once the cache was imported into Houdini Solaris, several USD-specific features were used to refine the crowd management in LOPs.

The Variant Set system made it possible to assign each agent one of the five geometric variants without duplicating any source file: each agent points to a single prototype and inherits the variant. This approach ensures that changing the mesh of a variant is instantly reflected on every agent that uses it, without any re-export.

Levels of detail were handled through a second LOD Variant Set, switched dynamically according to the distance from the camera. Background agents therefore load a lighter version of their geometry, significantly reducing the memory load at render time without affecting visual quality in the foreground.

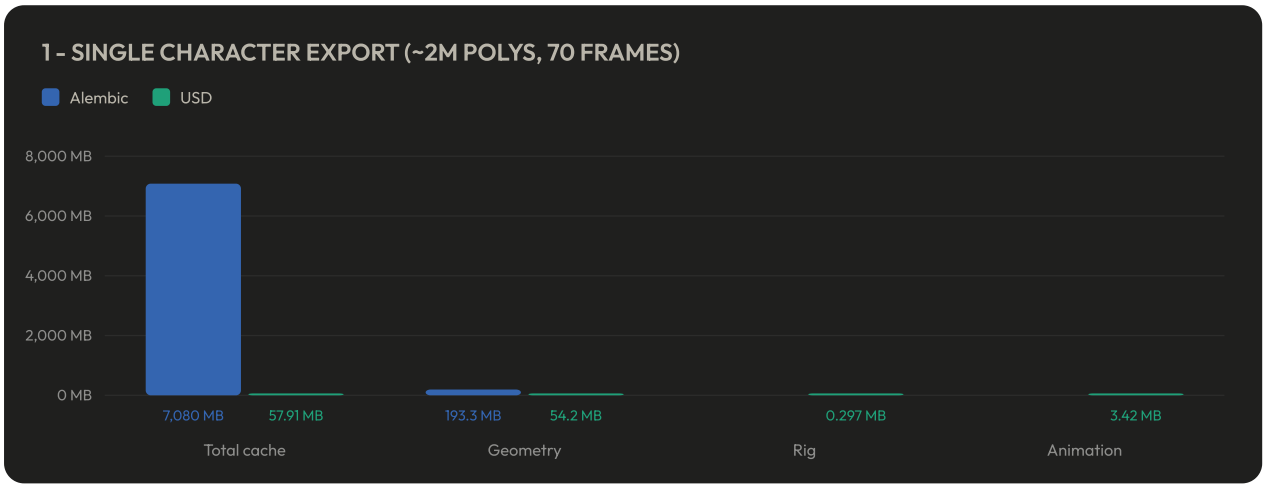
Finally, a camera frustum culling system was set up in a Python LOP: from the shot camera's parameters, a script computes the frustum planes and automatically deactivates the agents outside the frame by pruning them in the stage. This purely non-destructive operation deletes no data and can be undone or recomputed at any time without modifying the source files.

All of these mechanisms, variants, LOD and culling, operate solely at the USD stage level in LOPs, without ever requiring a re-export of the underlying caches. It is precisely this ability to stack artistic and technical decisions as light, reversible layers that fundamentally sets this workflow apart from traditional approaches based on frozen geometry caches.

VII: “RESULTS”

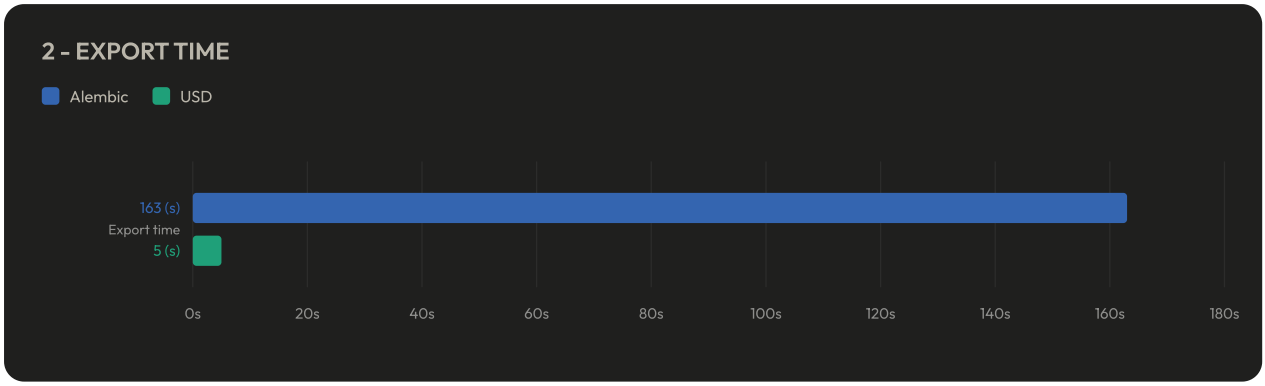
The measurements gathered clearly confirm the two hypotheses set out earlier: USD outperforms Alembic both in data volume and in iteration flexibility.

7.1 - Single character export, a reduction factor of 122



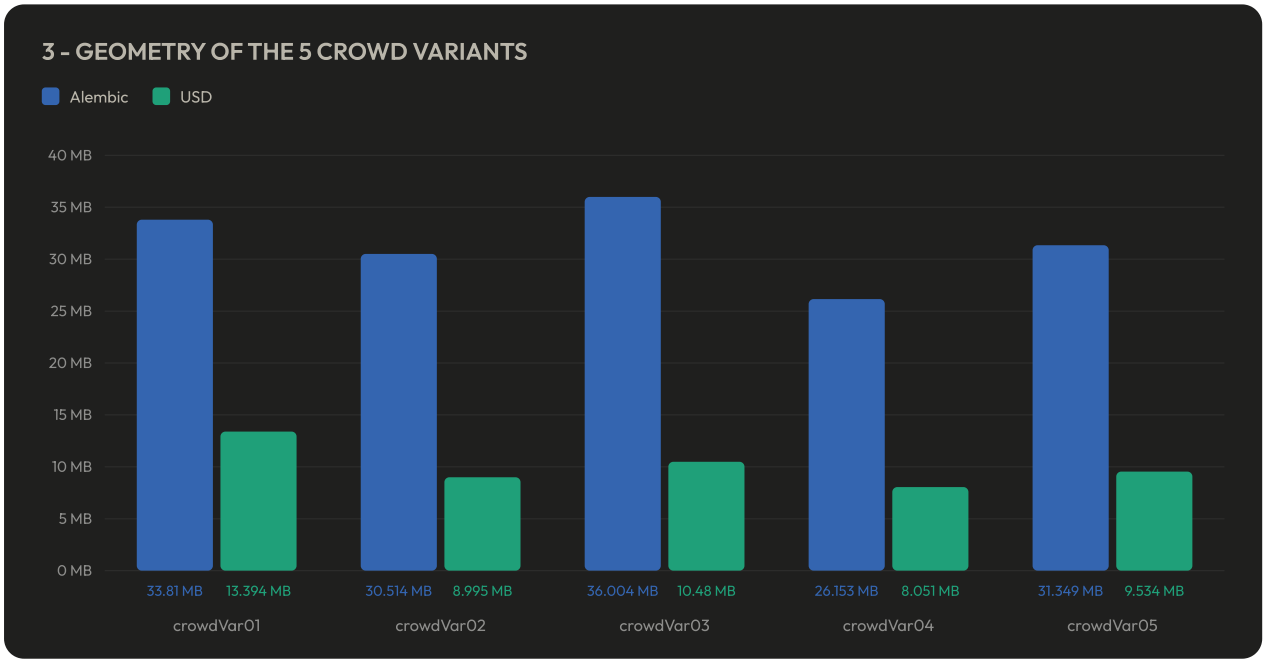
On a single character, the Alembic cache reaches 7.08 GB against 57.91 MB in USD, a reduction by a factor of 122. Alembic writes every vertex position at every frame, while USD stores only the joint transformations in a dedicated layer of 0.297 MB, the geometry and rig being shared across all shots.

7.2 - Export time, from 2 minutes 43 to 5 seconds



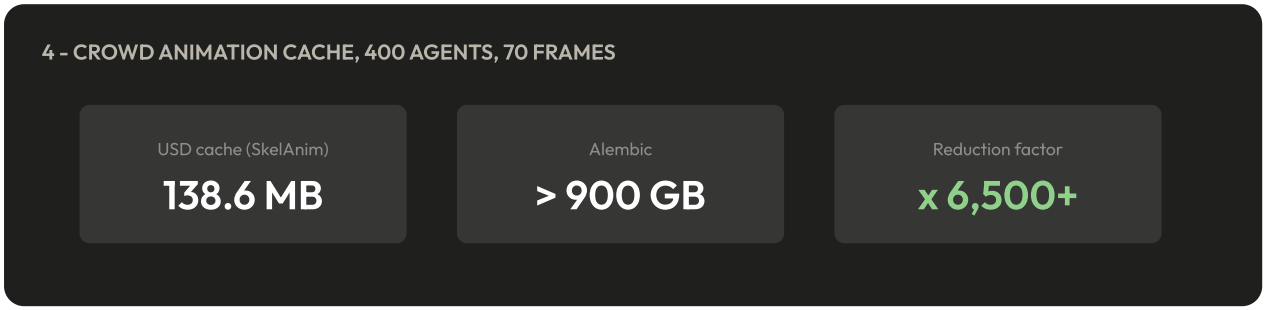
5 seconds in USD against 2 minutes 43 in Alembic on the same test case. In intensive production, this difference directly changes the teams' ability to iterate: an animation fix becomes almost instantaneous, where Alembic forces a systematic wait.

7.3 - Geometry of the five variants, 157 MB against 50 MB



On the static geometry of the five variants, the gap is clear: 157.8 MB in Alembic against 50.5 MB in USD. This is because the USD export keeps only the raw geometry, without any unnecessary attribute or primvar, whereas Alembic embeds all the data of the source mesh.

7.4 - 400-agent crowd cache, 138 MB against 900 GB



The most significant result concerns the full simulation cache. For 400 agents animated over 70 frames, the USD cache weighs 138.6 MB. The Alembic equivalent exceeds 900 GB, a volume completely incompatible with shared network storage in a school setting, and one that would make this workflow unthinkable in production.

VIII: “DISCUSSION, LIMITATIONS AND CONCLUSION”

The results validate the initial hypotheses. The USD workflow outperforms traditional methods on both axes studied, data volume and iteration flexibility. These results must however be qualified by the practical experience of putting them into practice.

8.1 - From the cube to 400 robots, the gap between theory and reality

On paper, the USD logic is elegant. Applied to a cube, it works immediately. Applied to 400 agents with complex rigs coming from Maya, it reveals a much more demanding reality: badly bound skins, incorrect transformation matrices, scale differences between software packages, layer composition errors. The step-by-step method proved essential: add complexity only once each building block is validated, and simplify as soon as a problem resists.

USD is a deep language. For basic uses, it is immediately effective. For advanced uses, it requires a fine understanding of its internal logic that few documentation resources help you acquire quickly. This is both its greatest strength and what sometimes makes it dizzying.

8.2 - The practical limits of the USD workflow in production

The main limitation of the USD workflow appeared when assembling complex shots with many nested references. Unlike Alembic, which flattens all the information into a single file that is therefore fast to read, USD has to resolve all the referenced layers at load time, which can make the viewport very slow to load on dense scenes.

This limitation is directly tied to the speed of the local network server and to the minimum machine configurations, in particular the 16 GB of DDR4 RAM of the least powerful PCs, which struggle to load a full crowd scene interactively. Workarounds were put in place, such as lightweight assembly layers for daily work, loading only 50 agents instead of 400, with the full 400 sent to the farm for final rendering.

The version of Tractor used was also a comfort limitation, with more restricted machine communication and control features than current versions, though without directly affecting the research results.

8.3 - A standard for the future, and a confirmed direction

This research confirmed that USD is a concrete answer to the challenges of crowd simulation, not because it simplifies the problem, but because it provides the tools to master it cleanly. Most of the learning lies in the gap between understanding USD on a simple case and applying it to a crowd of 400 robots. This project opened up prospects towards Pipeline TD or FX TD roles, jobs where technical rigour and the ability to pass knowledge on matter as much as the craft itself.

IX: “BIBLIOGRAPHY”

ACM Digital Library, SIGGRAPH Course ’25, USD in Production.
https://dl.acm.org/doi/full/10.1145/3721241.3733995

NVIDIA Developer (2025). OpenUSD Workflows and Performance. Accessed in 2026.
https://developer.nvidia.com/openusd

NVIDIA Omniverse (2025). Universal Scene Description, Interoperability and Pipeline Integration. Accessed in 2026.
https://docs.omniverse.nvidia.com/usd/latest/index.html

Pixar Graphics Technologies (2025). USD Documentation. Accessed in 2026.
https://openusd.org/release/index.html

SideFX (2025). Houdini Crowd Agents Documentation. Accessed in 2026.
https://www.sidefx.com/docs/houdini/crowds/index.html

SideFX (2025). Houdini Solaris & USD Basics. Accessed in 2026.
https://www.sidefx.com/docs/houdini/solaris/index.html