A Scalable Server for 3D Metaverses

Ewen Cheslack-Postava[1], Tahir Azim[1], Behram F.T. Mistree[1],
Daniel Reiter Horn[1], Jeff Terrace[2], Philip Levis[1], and Michael J. Freedman[2]

[1]Stanford University [2]Princeton University

Metaverses are three-dimensional virtual worlds where anyone can add and script new objects. Metaverses today, such as Second Life, are dull, lifeless, and stagnant because users can see and interact with only a tiny region around them. Current metaverses impose this distance restriction in order to scale to large worlds, as the restriction avoids appreciable shared state in underlying distributed systems.

We present the design and implementation of the Sirikata metaverse server. Sirikata scales to support large, complex worlds, even as it allows users to see and interact with the entire world. It achieves both goals by leveraging properties of the real world in its core systems, such as a novel distributed data structure for virtual object queries based on visible size. Applications developed by Sirikata users support our claim that removing the distance restriction enables new, compelling applications that are infeasible in today's metaverses.

1. MOTIVATION

Users of today's metaverses miss out on a truly immersive experience because they can see and interact with only the tiny, local region around them. For example, Second Life rendered completely is a fantastic and eye-popping megalopolis. But users are limited to their local surroundings because Second Life only displays nearby objects and imposes a maximum object interaction distance of ~100 meters. This restriction allows a metaverse to scale easily because a server only shares state and communicates with the servers that manage neighboring regions. But this scalability comes at the cost of user experience. A distant building should be able to attract a user's attention, and she should be able to select it and send it messages to learn more about it. Such a simple interaction is impossible in today's metaverses.

In contrast to metaverses, games, such as World of Warcraft, avoid this problem because the game provider is the centralized author of all world content and the system is application-specific. The provider can design content around technical limitations, for example precomputing scene optimizations [3] and imposters for efficient rendering [2]. As narrow, tailored applications, games can leverage game mechanics for improved performance: Donnybrook, for instance, showed how focusing updates around the flag bearer in capture the flag leads to lower bandwidth demand [1]. Application-specific optimizations commonly permeate the designs of these systems.

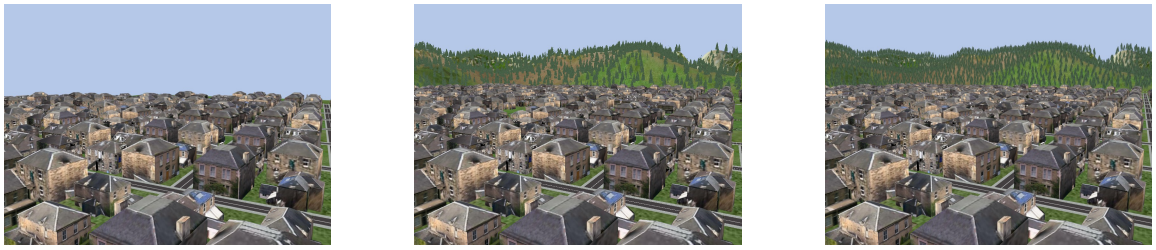
We believe that metaverse users should be able to view and interact with the entire world. However, metaverses differ greatly from games and encounter correspondingly different technical challenges. Instead of static, centrally authored content, a metaverse is populated by many user-generated applications whose dynamic behavior, placement, and appearance are controlled by their owners. To support such worlds, metaverses must somehow limit interaction. Without limits the system would quickly exceed capacity: clients cannot render the entire world in full detail, position updates for every object would saturate a server's capacity, and even sending a list of all objects to a client would overload a server.

2. DESIGN

Sirikata establishes a novel principle for how to build scalable systems for dynamic, user-extensible virtual environments. The key insight for achieving this scalability goal is that a virtual world, being a 3D, geometric environment, has many similarities to the real world. By leveraging real-world-like constraints, the server can simultaneously scale to a large world and provide a natural as well as intuitive experience. Sirikata allows clients to see and interact with a large and complex world, even as the system grows to many servers.

This principle of leveraging real-world constraints underlies all of Sirikata's core systems and is critical to scalability. Sirikata "space servers," or simply servers, are authoritative for the presence and positions of objects, simulate physics, inform observers of other relevant objects and route application level messages. The space server comprises a set of services designed around real-world constraints:

- A server partitioning service that divides a world into regions and maps regions to servers. Assum-



(a) Distance (3,000 objects) (b) Sirikata (3,000 objects) (c) Ideal (10,000 objects)

Figure 1: Comparison of current systems (distance queries), Sirikata (solid angles with aggregates), and the ideal scene. Sirikata allows the user to see and interact with the entire world.

ing an object density distribution similar to the real world leads to a design using a distributed, split-axis kd-tree, with a highly replicated, stable upper tree and distributed lower trees to spread load.

- An object discovery service that prioritizes objects by visual importance, choosing objects with larger solid angles, with a novel distributed data structure and supporting query processor. Collections of objects that are too small to be seen individually are reported and displayed as aggregate objects.
- A message routing table that maps object identifiers to servers. Every server can forward to any object in the world, but the routing table scales well because, like the real world, most interactions are local and most objects are stationary.
- A message forwarder that guarantees a minimum throughput even as the number of communicating objects grows very large, decaying smoothly over distance.

3. EVALUATION

Figure 1 demonstrates the visual improvement of Sirikata compared to existing appraoches. The full scene in Figure 1(c) is a small town with 10,000 objects — houses, streets, terrain, and trees. Clients can only render ~3,000 objects interactively, so queries return a subset of the full scene. Figure 1(a) shows the experience when display and interaction are limited to short distance, missing important objectects like the terrain. As Figure 1(b) shows, solid angles with aggregates complete the picture, including detail — such as the distant forest — that is lost otherwise, allowing a client to see a compelling approximation of the entire world. Further, an avatar can interact with large and distance objects, messaging them to interact with them.

Performance and scalability are only useful if the ability to see and interact with the entire world leads to engaging applications that are not possible in today's systems. We hired 11 undergraduate computer science students for the Summer of 2011 to develop applications in Sirikata. In a few weeks, they created a variety of applications and systems, including:

Minimap - a top-down display of objects and events,

Wiki-world - embedded Wikipedia articles from object metadata,

Escrow - secure object and virtual currency exchange,

Games, including a Pokemon-based RPG, a 3D Pacman clone, 3D tower defense, and a Minecraft clone,

Procedural generators for both city road networks and building layouts, and

Phys-lib, a library for customizable physics with an example billiards application.

These applications demonstrate that Sirikata supports a variety of metaverse applications running concurrently, including some that would be very difficult or impossible in current systems.

4. CONCLUSION

The Sirikata metaverse server allows users to see and interact with a large virtual world filled with dynamic, user-generated objects. Properties of the real world inform its design and provide constraints that enable scalability with intuitive implications: observers have limited resolution, objects are mostly static, and object density is in a small, fixed range. Sirikata enables applications that are very difficult to build in existing systems without sacrificing performance.

5. REFERENCES

[1] A. Bharambe, J. R. Douceur, J. R. Lorch, T. Moscibroda, J. Pang, S. Seshan, and X. Zhuang. Donnybrook: Enabling large-scale, high-speed, peer-to-peer games. In *SIGCOMM Comput. Commun. Rev.*, Aug. 2008.

[2] T. A. Funkhouser and C. H. Séquin. Adaptive display algorithm for interactive frame rates during visualization of complex virtual environments. ACM SIGGRAPH, 1993.

[3] R. Shumacker, R. Brand, M. Gilliand, and W. Sharp. Study for applying computer-generated images to visual simulation. Technical Report AFHRL-TR-69-14, U.S. Air Force Human Resources Lab, 1969.