SOLIPSIS: A Massively Multi-Participant Virtual World

Joaquín KELLER, Gwendal SIMON

Abstract—**This paper presents a massively shared virtual reality system based on a network of peers. It does not rely on any server nor on IP multicast, and intends to be scalable to an unlimited number of participants.**

Following a peer-to-peer scheme, entities collaborate to build up a common virtual world. The behavior of entities, running algorithms in order to maintain local properties, ensures the consistency of the virtual world and the connexity of the network.

The paper also describes how entities join the network and enter the virtual world at a particular position.

Keywords— **Peer-to-peer System, Shared Virtual Reality, Massively Distributed System, Distributed Algorithms, Self-organizing Systems, Computational Geometry.**

I. INTRODUCTION

A virtual world or shared virtual environment is a computer-generated space used as a metaphor for interaction. Entities, driven by users (avatars) or by computer (virtual objects), enter and leave the world, move from one virtual place to another, and interact in real-time. Shared virtual reality applications provide a similar perception of the same scene for any two entities.

Recent years have seen a growing interest among the scientific community in large-scale shared virtual reality applications inhabited by thousands of entities (eg. [1], [2], [3]). And commercial applications, like MMORPGs (Massively Multiplayer Online RolePlaying Game, eg. [4], [5], [6]) offer now real time interactions in a common virtual world to thousands of simultaneous gamers. But, as actual implementations rely on servers and/or IP multicast, only a limited number of participants can interact simultaneously. Increasing the number of participants can only be achieved by increasing the number of servers in a well connected cluster.

The system we are designing and building, SOLIPSIS [7], intends to be scalable to an unlimited number of users and accessible by any computer connected to the Internet. It does not make use of any server and is solely based on a network of peers.

Joaquín Keller is a researcher at France Telecom R&D and Gwendal Simon a PhD student at INRIA-Rennes and France Telecom R&D ; Email: {joaquin.keller, gwendal.simon}@rd.francetelecom.com ; Address: 38, rue du Général Leclerc - 92794 Issy-Moulineaux - France ; Phone: +33 1 45 29 52 86 ; Fax: +33 1 45 29 52 94

In our system, each participating computer runs a specific software that holds and controls one or several peers. These peers implement the entities of the virtual world and "perceive" their surroundings. Peers can be lite pieces of software and SOLIPSIS aims to be accessible to low end computers connected at 56Kbs and to mobile wireless devices and not only to full featured broadband connected engines. Connected peers may exchange data like video, audio, avatars movements or any kind of events affecting the representation of the virtual world.

After an introduction to our notations (§II), we present the local properties of the SOLIPSIS network in §III and the algorithms to maintain these properties in §IV. Due to physical limitations, a peer cannot be connected to an unlimited number of peers. The §V describe policies to drop out connections. Then, virtual teleportation and world login require specific algorithms presented in §VI.

II. NOTATIONS AND DEFINITIONS

Each entity of the SOLIPSIS virtual world is implemented by a node or peer of the SOLIPSIS network. Peer, node and entity will be considered as synonyms in this document. The only elements of the SOLIPSIS world are the entities. The difference between a virtual object and an avatar is that an avatar is associated to an user. Each entity is identified by an unique id and the set of entities is noted E.

Entities determine and are responsible for their own position in the SOLIPSIS world, a two-dimensional torus $T = \{(x, y) \in \mathbb{N}_{size_x} \times \mathbb{N}_{size_y}\}$ where $size_x$ and $size_y$ are the size of SOLIPSIS world and $\mathbb{N}_k$ denotes the positive integers modulo k. We have chosen $size_x$ and $size_y$ very large: 128 bits each, so approximately 10^{75} different positions.

The torus is one of the simplest finite unbounded surface. By "unrolling" a torus, it can be represented by a *flat torus* and considered as a rectangular tile [8]. The figure 1 represents the tiling defined by horizontal and vertical translations of this tile. An entity e is determined by its position $(x_e, y_e) \in T$. Its infinite copies in T are $\{(x_e + m, y_e + n); m, n \in Z\}$. In the following, $[e, e']$ refers to the shortest geodesic joining the entities e and e' and $d(e, e')$ is its length.

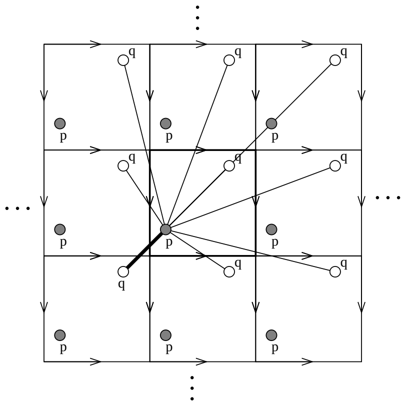

Fig. 1. The tiling defined by infinite copies of the flat torus, the infinites copies of points p and q and the geodesics joining p and q.

A connection between two entities e and e' in SOLIPSIS network is bidirectional and means mutual knowledge. Connected entities are able to communicate their respective positions on a regular basis, modifications on their virtual representations or informations about some other entities. The set of connections is noted C.

Entities and connections define a graph G where the set of vertices is E and the set of edges is C.

Let $k(e)$ be the set of adjacents of e in G. The cardinal of $k(e)$ is the degree of e.

The directed angle $\angle(e'\ e\ e'')$ is defined by the counterclockwise angle formed at e by $[e, e']$ and $[e, e'']$. An entity e_i lies in the directed sector $\nabla(e'\ e\ e'')$ when $\angle(e'\ e\ e'') = \angle(e'\ e\ e_i) + \angle(e_i\ e\ e'')$. Also, we define the successor of an entity e' for an entity e_0 by:

$$s_{e_0}e' = e'' \quad \Leftrightarrow \quad \forall e \in k(e_0) : e \notin \nabla(e'\ e_0\ e'')$$

The p^{th} successor of e' for e is noted $s_e^p e'$, the predecessor of e' is $s_e^{-1}e'$ and, if e knows n entities, $s_e^n e' = e'$.

Note that, anytime, users get in and off, entities move and connections are modified. So, the SOLIPSIS network is highly dynamic and almost all variables at time t may be different at time $t + \Delta t$. So G can be considered as a dynamic graph [9].

III. PROPERTIES

The global properties of SOLIPSIS network must match with the virtual reality application features. In particular, SOLIPSIS must provide consistency and ability to move all over the virtual world. These global challenges require that each entity respects two local properties detailed in this section.

A. Local Awareness

Each entity perceives only a part of the virtual world inhabited by some entities. It should be aware of all modifications on the virtual representations of entities lying in this area. When an entity modifies its virtual representation, it informs only its adjacents on this event. So, each entity should be adjacent of all entities belonging in its perception space. We call **Awareness Area** of an entity e the virtual space perceived by e and **Local Awareness** the property ensuring that e knows all the entities in its Awareness Area.

In the reality, the Awareness Area corresponds to the immediate surroundings. So, we define the Awareness Area of e as the disk of radius $r(e)$, centered on (x_e, y_e) and noted $A(e) = \{(x, y) : d(e, (x, y)) \leq r(e)\}$

The radius $r(e)$ is variable and depends on density of entities in this area and physical capacity of e. We note $a(e')$ the set of entities lying in $A(e')$:

$$a(e') = \{e \in E : pos_e \in A(e')\}$$

Property 1 *The Local Awareness property of an entity e is ensured when:* $a(e) \subseteq k(e)$

Consider two entities e' and e'' so that $A(e') \cap A(e'') \neq \emptyset$ and an entity $e \in a(e') \cap a(e'')$. If e' and e'' respect the property 1, they will be informed simultaneously on a modification on e. So, the respect of the property 1 leads to the local consistency.

B. Global Connectivity

In order to ensure the Local Awareness property, an entity can only rely on its adjacents. If it does not know any entity in some large sector, it will hardly know about an entity arriving from this sector. Conversely, if it moves forward a sector with no known entity, it will hardly get aware of entities it should met on its path.

Based on Computational Geometry notions [10], the Global Connectivity property aims that an entity will not "turns its back" to a portion of the world.

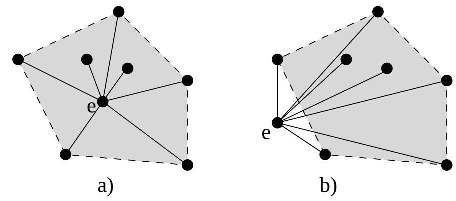

Fig. 2. The convex hull of the adjacents of e. In situation a, e respects the Global connectivity property while it does not in b.

The convex hull of a set of points (here entities) is known as the smallest convex polygon containing this set. We note $CH(E')$ the set of positions enclosed in the convex hull of the subset $E' \subseteq E$

Property 2 *The Global Connectivity property of an entity e is ensured when:* $pos_e \in CH(k(e))$

It is important to point out that we do not force an entity to change its position in order to ensure its global connectivity property. On the contrary, we impose to each entity to be connected with entities that allow it to ensure the property 2. Thus, we propose another formal definition of the Global Connectivity property.

Let $H(e)$ be the set of subset of entities that could permit to e to ensure its property:

$$H(e) = \{h \subseteq E : pos_e \in CH(h)\}$$

The Global Connectivity property is ensured for e when: $\exists h \in H(e) : h \subseteq k(e)$

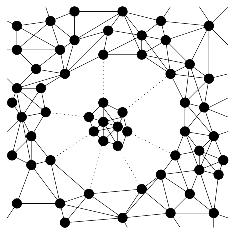

Fig. 3. The dotted connections are due to property 2.

This property reduces the risk of graph disconnection by avoiding isolation of a set of entities (see Figure 3). Moreover, it forces entities to have, at least, three adjacents, so at least two failures or departures are tolerated. Finally, it impacts on topology of the SOLIPSIS network that looks like a mesh.

IV. MAINTAINING PROPERTIES

This section describes the collaboration scheme that maintains Local Awareness and the recursive mechanism that recovers from a loss of Global Connectivity.

A. Spontaneous Collaboration for Local Awareness

In order to respect the Local Awareness property, an entity must know all entities in its surroundings. Due to mobility, anytime, some entities may enter in its Awareness Area. A query-response mechanism should require that each entity asks on a regular basis to its adjacents if they have detected any new entity in its Awareness Area since last query. It would generate many irrelevant messages and temporal lacks of consistency.

We propose a spontaneous collaboration scheme in which each entity sends a message when it detects that an entity enters in the Awareness Area of another. It could generate some redundant messages, but that reduces the nuisance capacity of malicious entities.

Rule 1 $\forall e', e'' \in k(e)$, *if* e' *enters in* $A(e'')$, e *must send to* e'' *a specific message containing* $id_{e'}$ *and* $pos_{e'}$.

Five events may force e to send a message to e'':

- e' moves closer to e'';
- e'' moves closer to e';
- e moves away from e'';
- $A(e'')$ grows up;
- e' was not in $k(e)$ at time $t - \Delta t$ and is at time t.

The respect of this rule ensures that (i) e'' will get inform of the arrival of e' and (ii) e'' will acquire information about its new environment as it goes forward.

B. Recursive Query-Response for Global Connectivity

The SOLIPSIS dynamic characteristic may sometimes lead to situations where an entity e respects the Global Connectivity property at instant $t - \Delta t$ and does not at instant t. An entity e is able to determine easily if it respects the property 2 by verifying:

$$\forall e' \in k(e) : \; s_e e' = e'' \; \Rightarrow \; \angle(e' \; e \; e'') < \pi$$

When an entity e detects two consecutive adjacent entities e_1 and e_f with $\angle(e_1 \; e \; e_f) \geq \pi$, immediately, it sends a message, querying entities in the sector $\nabla(e_1 \; e \; e_f)$ (see Figure 4). If e_1 receives this message and if it respects the property 2, it is connected with an entity e_2 that lies in the half-plane delimited by Δ_1. The entity e_2 verifies $\angle(e_2 \; e \; e_f) < \angle(e_1 \; e \; e_f)$.

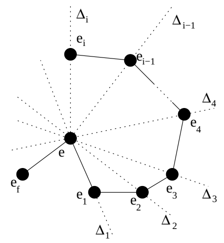

Fig. 4. Restoring the Global Connectivity of an entity e.

If $\angle(e_2 \; e \; e_f) < \pi$, the entity respects back its Global Connectivity property. But, if e_2 is not sufficient, e can send recursively a message to e_2, querying an entity in the sector $\nabla(e_2 \; e \; e_f)$. In the same manner, if e_2 respects the property 2, it knows an entity e_3 in the half-plane delimited by Δ_2.

Recursively, e receives informations about an entity e_i so that $\angle(e_i \; e \; e_f) < \angle(e_{i-1} \; e \; e_f)$. The algorithm ends when $\angle(e_i \; e \; e_f) < \pi$.

We can implement an optimized scheme where the entity e sends only two messages, one for e_f and one for e_1. Each entity receiving the message forwards it to an adjacent that narrows the wide sector until it reaches an entity e_α that lies in $\nabla(e'_f \; e \; e'_1)$, where e'_1 and e'_f are the images of e_1 and e_f by central symmetry with center e. The entity e_α verifies $\angle(e_1 \; e \; e_\alpha) < \pi$ and $\angle(e_\alpha \; e \; e_f) < \pi$ so e recovers its Global Connectivity property.

V. OPTIMIZATION

Previous sections have dealt with how to inquire more entities and make more connections. But as entities has limited resources, they may also need sometimes to drop out connections.

When an entity e' wants to abort some connections, it can only choose those with entities e verifying:

$$e \notin a(e') \ \wedge \ e' \notin a(e) \tag{1}$$

The entity e' cannot drop out connections too with entities e so that:

$$pos_{e'} \notin CH(k(e')\backslash\{e\}) \wedge pos_e \notin CH(k(e)\backslash\{e'\}) \tag{2}$$

One first solution is obvious: e' can reduce its awareness radius $r(e')$. Thus, it can drop out connections with entities that do not belong any more to $a(e')$ and verify condition 2.

Another solution comes from the condition 2. Let $h(e')$ be the set of entities sets that allow e' to respect the Global Connectivity: $h(e') = \{h \subseteq k(e') : e' \in CH(h)\}$

If the cardinal of $h(e')$ is greater than one, there is at least an element of $h(e')$ (a set of entities) that is not necessary for e'. Therefore, there are entities which may not be essential to the Global Connectivity respect. A potential optimization consists in ordering the element of $h(e')$ in order to determine an element $min(h(e'))$ as the "better" convex polygon of adjacents. Then, e' could drop out connections with entities e verifying the condition 1 and $e \notin min(h(e'))$.

A challenge is to determine the order relation. We identify some possibilities:

- based on statistics on behavior of adjacents, it is possible to order the sets of entities using their verbosity and/or their expected aliveness. An entity that sends a lot of messages may use a large part of the bandwidth, while an entity that has a big potential aliveness has less chance to disconnect. By adding these statistics, each set of entities can be characterized by a value;
- an entity that owns a large Awareness Area is aware on a vast world region. So, sets of entities can be ordered by their awareness radius length in order to be connected with entities that cover a larger part of the world. As there is a correlation between Awareness Area size and physical capacity, this order relation allows to be connected with entities that have the largest physical capacities;
- an entity whose Awareness Area covers a large part of the boundary of the Awareness Area of an entity e has more chance to indicate to e that another entity enters in its Awareness Area. So, entities may be ordered by their Awareness Area boundaries coverage;
- the simplest solution consists in ordering the sets of entities by the perimeter of the convex polygon generated.

Finally, an entity e cares more on entities that are near to enter in its Awareness Area or move in its direction. We introduce another rule that aims to know the entities that go closer to e.

We note $C(e', e)$ the disk of radius $d(e', e)$ centered on e' so that: $C(e', e) = \{(x, y) : d(e', (x, y)) \leq d(e', e)\}$

When an entity e' lies in $C(e, e'')$, it means that e' is nearest to e than e''. This natural property is often used in neighborhood graphs [11].

Rule 2 *$\forall e', e'' \in k(e)$, if e'' enters in $C(e, e')$, e must send to e' a specific message containing $id_{e''}$ and $pos_{e''}$.*

The respect of this rule allows each entity to be connected with its nearest neighbors, whether they belong to its Awareness Area or not. Moreover, an entity that moves in a direction know, in advance, entities on its path. At last, we expect that the policy chosen to drop out connections is more efficient with a better knowledge of surroundings.

VI. LOGIN AND TELEPORTATION

When an entity e makes an abrupt move to a completely new location, $k(e)$ becomes inaccurate and neither local awareness nor global connectivity will be restored using the previously described algorithms. Same problem arises when e has been logged off SOLIPSIS for a long time.

We need analgorithm that allows an entity to restore its local awareness and global connectivity "from scratch", knowing only its expected or desired location. The algorithm is called *Reverse Localization* in the sense that it takes a location as argument and returns the nearest entities to this location. It is similar on many points with some position-based routing algorithms in ad-hoc networks [12] and precisely the greedy-perimeter algorithm [13], [14].

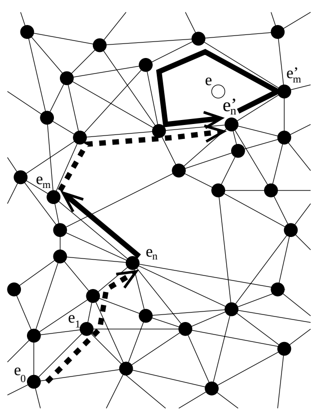

Fig. 5. Reverse Localization Algorithm: a first stage to reach e_n, an aborted turn around e_m, another stage to reach e'_n and turn around e.

When an entity e wants to go (at log in or whenever) to a specific target position (x_e, y_e), it first needs to know an entity e_0 (it can be e itself before moving) connected to the SOLIPSIS network (see Figure 5). To start the algorithm, a message specifying id_e and target position is sent to e_0.

Following a greedy routing scheme, this message is routed from e_0 to SOLIPSIS nodes to the nearest-to-the-target known entity. Thus, e_0 will forward it to $e_1 \in k(e_0)$, which is the closest to the target among entities in $k(e_0) \cup \{e_0\}$. And so on, recursively until the message reaches an entity e_n that has no closer neighbor to the target than itself.

As e_n assumes to be the nearest neighbor of e (in its new position), it opens a connection with e. Actually, this could be false but next stage will reveal it.

The algorithm enters now a new stage aiming to collect entities all around the target. Entity e_n starts the recursive process by sending a message to e_m, the nearest known entity (besides itself) to the target. Note that the way of the rotation around e depends on the position of e_m. If e_m is on the right of $[e_n, e]$, the message will turn around e counterclockwise and *vice versa*.

At first, e_m contacts e. All entities met on this turn will generate the convex polygon required for the global connectivity property. Then, it determines e_{m+1} the nearest entity to target in $k(e_m) \setminus \{e_n\}$ and it compares $d(e, e_{m+1})$ and $d(e, e_n)$. If e_{m+1} is nearer to the target than e_n, the assumptions that e_n was the nearest entity to target does not hold any more and the algorithm starts back at the early stage, with a new e_0 set to e_{m+1}.

If e_n is indeed the nearest entity, e_m forwards the message to the nearest to target entities in the half-plane delimited by the straight line (e, e_m) and turning in the direction initiated by e_n. And so on recursively until the half-line (e, e_n) is crossed. The end of the Reverse Localization Algorithm is notified to e by a specific message.

VII. CONCLUSION

This paper has presented the algorithms that make likely a peer-to-peer virtual reality. The SOLIPSIS virtual world will be able to handle million and even billions of entities: participants, bots and static objects.

The reasons that make this possible are not only the facts that SOLIPSIS does not rely on servers (that might constitute potential bottlenecks) and that any computer joining SOLIPSIS add computer power to the system. But, also because the underlying distributed algorithms are conceived in such a way that each node generates messages that remain local so the global amount of generated messages are proportional to the number of entities in the virtual world.

Algorithms have been first implemented in a simulator. This simulator have helped in improving the algorithms and in specifying the protocols. Simulations have shown that local awareness and global connectivity are well maintained and that traffic remains low and local despite variations in number of participants and huge mobility [15]. In particular, teleportation algorithms allow entering the world without disturbing the system.

We are now implementing for real a SOLIPSIS node. Also, we are specifying a light-weight protocol for communication that will allow anyone to build up its own SOLIPSIS node.

So expect in a near future, as people will start to run SOLIPSIS nodes, to enable a *Metaverse*-like [16] cyberspace.

REFERENCES

[1] "ANSI/IEEE 1516-2000 High Level Architecture (HLA)," 2000.

[2] J. Purbrick and C. Greenhalgh, "Extending Locales : Awareness Management in MASSIVE-3," *IEEE Virtual Reality 2000*.

[3] F. Dang Tran, M. Deslaugiers, A. Gerodolle, L. Hazard, and N. Rivierre, "An Open Middleware for Large-scale Networked Virtual Environments," in *IEEE Virtual Reality Conference 2002*.

[4] "The Anarchy Online game," http://www.anarchy-online.com/.

[5] "The Everquest game," http://everquest.station.sony.com/.

[6] "The Mankind game," http://www.mankind.net/.

[7] J. Keller and G. Simon, "Toward a Peer-to-Peer Shared Virtual Reality," in *IEEE Workshop on Resource Sharing in Massively Distributed Systems*, July 2002.

[8] C. I. Grima and A. Màrquez, *Computational Geometry on Surfaces*, Kluwer Academic Publishers, 2001.

[9] D. Eppstein, E. Galil, and G. F. Italiano, "Dynamic Graph Algorithms," Tech. Rep., University Ca'Foscari di Venezia, 1996.

[10] F. P. Preparata and M. I. Shamos, *Computational Geometry: An Introduction*, Springer-Verlag, 1985.

[11] J. W. Jaromczyk and G. T. Toussaint, "Relative Neighborhood Graphs and Their Relatives," in *Proc. IEEE*, 1992, vol. 80.

[12] I. Stojmenovic, "Position-Based Routing in Ad Hoc Networks," *IEEE Communications Magazine*, vol. 40, no. 7, 2002.

[13] B. Karp and H. T. Kung, "GPSR: Greedy Perimeter Stateless Routing for Wireless Networks," in *6th ACM Conference on Mobile Computing and Networking (Mobicom '00)*, 2000.

[14] L. Barriere, P. Fraigniaud, L. Narayanan, and J. Opatrny, "Robust Position-Based Routing in Wireless Ad Hoc Networks with Unstable Transmission Ranges," in *Proceedings of DialM*, 2001.

[15] T. Camp, J. Boleng, and V. Davies, "A survey of mobility models for ad-hoc network research," *Wireless Communications and Mobile Computing*, 2002.

[16] Neal Stephenson, *Snow Crash*, 1992.