DISTORTION IN DISTRIBUTED VIRTUAL ENVIRONMENTS

Matthew D. Ryan Paul M. Sharkey

M.D.Ryan@reading.ac.uk, *P.M.Sharkey@reading.ac.uk*

Interactive Systems Research Group

Department of Cybernetics, The University of Reading, Whiteknights, Reading RG6 6AY, UK

http://www.cyber.reading.ac.uk/research/ISRG/

ABSTRACT

This paper proposes a solution to the problems associated with network latency in multi-user, distributed virtual environments (DVE's). It begins by comparing DVE's with synchronised host clocks against unsynchronised host clocks. A hybrid solution is described, which combines the advantages of both synchronised and unsynchronised models, using the concept of a causal surface. The hybrid system results in distortion of objects, and this is proposed as a solution which facilitates dynamic real-time user collaboration. The final section covers implementation details, with reference to a prototype system available from the Internet.

1. INTRODUCTION

Within a distributed virtual environment (DVE) users interact with objects causing the state of those objects to change. These state change events are normally transmitted to other users within the environment across a network using either of the following mechanisms:

1. ***No clock synchronisation (unsynchronised).*** The virtual environment contains no uniform, global time, although each event is parameterised with respect to a uniform local time. All event messages are sent without reference to when the event occurred, and events are replayed as soon as a user receives them.
2. ***Global time (synchronised).*** Events are time-stamped according to a global time that is synchronised between all users [5]. When events are received by a user, they are replayed at the correct global time. This usually requires the model to be rolled back to the correct time of the event, and then advanced to the current time.

NETWORK LATENCY

The network connecting users within the DVE induces latency of information interchange. This latency causes problems in the following three areas: user representation, object representation and user collaboration.

A. User Representation

User representations will appear discontinuous if a synchronised model is used. Fig. 1 exemplifies the problems associated with the two above approaches. Initially A is moving along the *x*-axis with a velocity of 1 unit per second, and there is a delay of 1 second between the users.

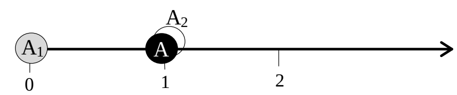
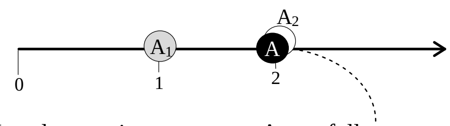
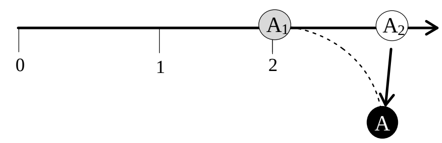
Time	View seen by remote user
t=1	 A is moving along horizontal path. A_1 is seen delayed by 1 second, A_2 is consistent with A.
t=2	 Local user input causes A to follow a new path, message is sent to remote user. A_1 delayed by 1 second, A_2 is still consistent with A.
t=3	 Message about path change is received by remote user. A still follows new path. A_1 still delayed by 1 second, so now the new path can be started without any discontinuity. For the time t=[2,3] A_2 is inconsistent with A, and now A_2 has to discontinuously jump to A.

Figure 1. *User positions as seen by a remote user.* A *is the local user,* A_1 *is the user modeled using no clock synchronisation,* A_2 *is the user modeled using a global time.*

B. Object Representation

If objects are only acted upon by one user (i.e. user collaboration is not permitted), then an unsynchronised model of events is sufficient.

In an unsynchronised model, a local user will always see the past state of remote users, together with how those objects looked at the time of interaction. The model of objects at the local and remote sites will be inconsistent however, so a local user cannot interact with objects that a remote user has interacted with. Objects that are inconsistent cannot be allowed to interact, otherwise divergence of models will occur, as the following example demonstrates.

Consider a distributed billiards game where two users A and B are playing on two tables 1 and 2, respectively. At time *t*, the cue ball is knocked off table 1 and lands on table 2. Two methods of object update exist, as follows:

Objects updated locally by all hosts (distributed update)

To user B, the cue ball lands on table 2 at time $t+\bar{\Delta}$, where $\bar{\Delta}$ is the total network/processor update delay. The collision response of the balls, as seen by user 2, will depend on the state of the balls at time $t+\bar{\Delta}$. To user A, however, the ball will land on table 2 shortly after time t, and interact with those balls in the past state $t-\bar{\Delta}$ (user A's representation of table 2 is delayed by $\bar{\Delta}$). If the balls are moving, they will exist at different positions at times $t+\bar{\Delta}$ and $t-\bar{\Delta}$, so the resulting response will be different in the models of the tables held by users A and B, and both models will have diverged; the balls will appear to be in different places to each user.

Objects updated according to ownership (master update)
In this case the cue ball belongs to A, so only A will generate events for the cue ball. When the ball lands on table 2, A will calculate the collision response relating the position of balls on table 2 at time $t-\bar{\Delta}$, and send the events to B. These events will be received by B at time $t+\bar{\Delta}$, but they will relate to the positions of balls at time $t-\bar{\Delta}$, so the movements of the cue ball will not coincide with the positions of the balls as seen by B.

The only solution to the problem of model divergence is to use a synchronised model, where the positions of objects and the interaction responses will be consistent to all users.

C. Users Interacting with Shared Objects

Current implementations of distributed systems require users to own (either implicitly or explicitly) any objects they wish to interact with – any interaction must be preceded by a request for ownership (if ownership is not already established). When two users wish to interact with the same object, either one user must be subservient to the other in terms of that interaction (i.e. follow the motion of the object) or they must both be able to share ownership. The latter solution can lead to consistency problems.

In addition, where two users are holding either end of an object, standard control theory shows that the limit in bandwidth of interaction is given by a resonant frequency determined by the following condition

$$f_R = 1/(4\bar{\Delta}) \quad (1)$$

where $\bar{\Delta}$ is the network delay (under the reasonable assumption of unity gain in user interaction for frequencies less than f_R) [7]. In real-world interactions this delay (as determined by the stiffness of material and dimensions of the object) is negligible, thus allowing highly dynamic (high frequency) interaction. Where the delay is not negligable, this limiting frequency can be very low.

In a distributed virtual environment where each object's states (e.g. temperature, position, size) are consistent across the whole object, real-time collaborative interaction is not possible. A user cannot change an object until interactions from all users interested in changing that object are collated and the resultant action is determined. This will introduce an interaction delay of the round-trip network delay plus an extra delay to allow for remote user interaction [6].

PROPOSED SOLUTION

The technique described in this paper proposes a hybrid synchronised/unsynchronised model, combining the advantages of each method. A global time must be maintained between all participants to ensure consistency and allow interaction between users (as described in B above). Global time, however, results in discontinuities, (as described in A above), so remote users are displayed at a delayed time to ensure no discontinuities arise. The technique employs the causal surface [3], which facilitates collaborative interaction through the use of object distortion.

2. THE CAUSAL SURFACE

In the 3½-D perception filter [8], objects are delayed by different amounts as they move around space.

Unsynchronised model for remote users
Remote users are shown delayed by $\bar{\Delta}$ to remove discontinuities, as in an unsynchronised model. Objects are delayed by $\bar{\Delta}$ in the local user's view when coincident with the remote user. Therefore, objects that are being interacted with by a remote user are displayed at the same time as that user and the interaction is viewed without discontinuity.

Synchronised model for local user/object interaction
Objects are displayed in real time close to local user to allow interaction in real-time.

The variable delay is achieved by generating a causal surface function, $\Delta=\boldsymbol{S}(x,y,z)$, that relates delay to spatial position [3]. This surface is smooth and continuous to avoid object position/velocity discontinuities. Fig. 2 shows an example surface, generated using radial basis functions [2].

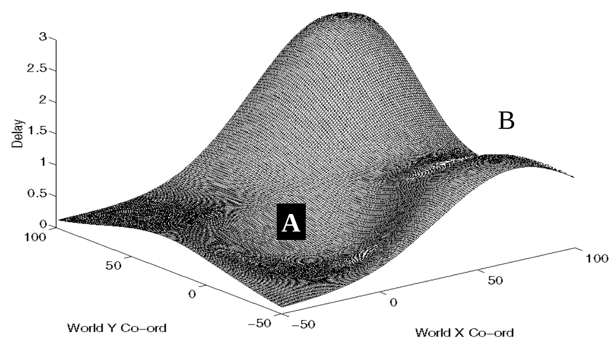


Figure 2. *Example causal surface*

The causal surface divides space-time into various causal regions. Fig. 2 represents a local user, A, at the position (0,0), and three remote users (the peaks on the surface). The vertical heights at positions coincident with each user represents the time it takes interactions from the local user to propagate to that remote user. For example, the surface has a value of Δ=1.5 at user B, showing that it takes 1.5 seconds for information to propagate between A and B.

In space-time it is impossible for information to reach a point under the causal surface, so the surface represents the bound on the causality of events occurring at the local user.

3. DISTORTION OF OBJECTS

DISTORTION IN THE REAL WORLD

In the real world, every event takes time to propagate and affect remote objects. Einstein proposed that all physical phenomena was limited by the speed of light. This critical speed limits the propagation of all causal influences, such as changes in gravitational and magnetic force [1]. Although the speed of light limits all interactions, the nature of some interactions result in an effective critical speed less than the speed of light. Examples of apparent slower critical speeds are the speed of sound, or the speed of wave propagation along a spring. Generally it can be said that all interactions are limited by a critical rate of propagation, although the critical rate may not be constant, and is dependant on the type of signal and on the material being propagated through.

An example of two users collaboratively moving a table will serve to demonstrate the causal restrictions within the real world. As one user (holding one end) moves slightly, the impulse will travel as a wave of distortion through the table, and eventually reach the other end of the table. The other user (holding the other end) will then feel the end move, and can apply their own force. The resultant force is then propagated back along the table to the first user. Under normal circumstances, because the time-delay of force propagation is very short, these causal restrictions are not readily apparent. If the time-delay increases, either as the table gets longer or as the rate of propagation reduces (the table is made of a more flexible material), then the effect of interaction propagation will become more apparent.

Equation 1 determines the frequency of interaction at which instability occurs. As the frequency of interaction approaches this critical frequency (in the absence of any visual feedback) then each user’s motions will be out of phase leading to resonance – making co-operative collaboration impossible. A very good example of resonance occurring can be illustrated if two people try to use a skipping rope *without* viewing the other person (a rope has very low stiffness and hence a discernible propagation delay). By looking only at their own hands it is possible to initiate a low frequency of rotation but it is impossible for higher frequencies. Brief experimentation showed that standing waves occurred at multiples of the resonant frequency (an expected result from the control theory [7]).

DISTORTION IN THE VIRTUAL WORLD

The delay associated with propagation of force along objects in the real world is derived from length and stiffness of the material. In the virtual world, however, the delay is derived from the network. The propagation of force as waves of distortion through objects in the real world is therefore analogous to propagation of event messages through the network. By showing the force propagation through objects in the form of distortion, users are made aware of the delay that is present, and can adapt to the maximum achievable rate of interaction.

As the previous example demonstrated, distortion of objects is a vital effect within the complete package of causal restriction, so if a virtual environment were to be developed that used the effects of causality from the real world, distortion must also be an integral part.

Fig. 3 shows two users holding either end of an object. The apparent stiffness of the object will be determined by the network delay and the object length (i.e. the distance between the users). In contrast to the real world, delays are not negligible – the lower apparent stiffness will lead to distortion. If user A were to move their end to the left as in (a), the force would propagate as a wave along the object, so initially B would feel no force. Some time later (b) the wave will have reached B, and so B will feel the force move their hand. In this way the force felt by B is the force input by A at a time $t-\bar{\Delta}$, where $\bar{\Delta}$ is the time it takes force to propagate through the object.

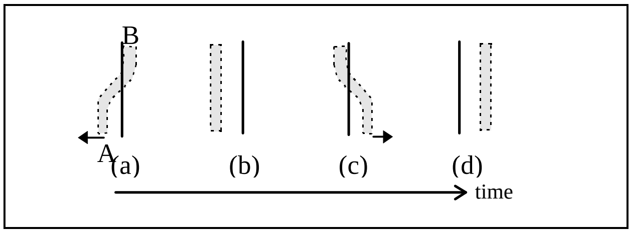


Figure 3. *Distortion of an oscillating object*

COLLABORATIVE INTERACTION

Propagation of state changes allows collaborative interaction. If a user changes the state of objects continually within their vicinity, those state changes will be reflected immediately on parts of the object close to that user. As the state change messages are transmitted across the network, the state changes initiated by the user will be seen to propagate along the object. When the state changes reach the remote user, their interaction is taken into consideration, and resultant actions are determined. These actions are sent continually as state changes back along the object (and also the network) to reach the local user again. Interactions propagate back and forth along objects to facilitate real-time interaction. This interaction occurs as a dynamic closed loop feedback system.

Exaggerated distortion need not be seen as a disadvantage within a virtual environment. Distortion has been used as an effective technique to increase the ‘realism’ and fluidity of cartoon animations for decades. When objects move fast in an animation, a strobing effect occurs when successive images of the object do not overlap. The strobing effect can be eliminated by stretching the object to cause the images to overlap. Also by squashing objects as they bounce against other objects, the collision appears to be more realistic, even though such distortions are rarely seen in the real world [4].

4. IMPLEMENTATION

OBJECT STATES

Each object in the view seen by a local user will be displayed at a time t-Δ, obtained from the causal surface function $\Delta = \boldsymbol{S}(x,y,z)$. The causal surface is continuous and dynamically changes as users move and as network delays vary, such that the value of the causal surface is always greater than the actual network delay at each user. Given that Δ is bounded within the range $[0, \max\{\boldsymbol{S}\}]$, all the states of each virtual object must be defined over the range $[t-\max\{\boldsymbol{S}\}, t]$, where t is current time. The trajectory of each object state can be implemented as a function or set of n functions:

$$X = \begin{cases} f_i(t), & t_i \le t < t_{i+1} \\ f_n(t), & t_n \ge t \end{cases} \qquad (2)$$

for i = 1, 2, ... n, where t_i is the start time of the i^{th} function, and $t_1 \le (t-\max\{\mathbf{S}\})$. The functions $f_i(t)$ are stored in a temporally ordered list of functions, the most recent function being held at the head of the list. To find the current function at time *t*, the list is traversed starting at the head, until the function start time is less than *t*.

A function $f_i(t)$ can be removed from the list if a function $f_j(t)$ exists in the list such that $t_j < (t\text{-max}\{\mathbf{S}\})$. The value of max{**S**} depends on the network latencies present in the system, and can therefore be very variable. To ensure maximum robustness, previous object states back to time *t=0* may be stored, but the constraint of host memory limits the amount of functions that can be preserved. As a compromise, a margin of error may be introduced by modifying the causal surface, **S**, such that the value of ***S*** at each user is equal to the current delay plus a suitable margin, e.g. $\boldsymbol{S}_{\mathrm{R}} = 150\% \bar{\Delta}_{\mathrm{R}}$, where R denotes a remote user.

Every defined parameter of each object (the states of the objects) must be specified over the interval $[t\text{–max}\{\boldsymbol{S}\}, t]$. The states of object may include, for example, position, orientation, colour, temperature, but all types of interaction available to the users must be defined parametrically. A separate list of functions must exist for each object state.

New state trajectories will be added to the list of trajectories in two ways:

1. ***Interaction*** – when an object interacts with other objects or users within the environment. For example, the position state trajectory will change when an object bounces on the floor.
2. ***Behaviour*** **–** autonomous objects will change state trajectories without interaction with other objects or users. For example, an autonomous virtual robot exploring the environment.

OBJECT REPRESENTATION

For more realism, each particle that makes up the virtual object must be displayed in the correct state for that particle's position in space. However, objects in most VR systems are described as points, lines and polygons, with graphics hardware developed for displaying these primitives efficiently. For graphics and processing efficiency, virtual objects are defined in terms of polygons. Fig. 3 showed an example of an object distorting in the virtual environment. The object appears smoothly distorted (curved) because the causal surface it spans is also smoothly curved. If that object were defined in the virtual environment in terms of a single rectangular polygon, then the object would appear linearly sheared and not smoothly curved as would be expected. To implement the solution proposed in this paper the object should be divided into a number of polygons depending on the amount of curvature of the surface. A maximum error bound, δ, can be defined, and the object divided into an optimal number of polygons such that for each distorted point on the surface of the object, the distance to the bounding polygon is less than δ.

VIRTUAL STATES

The previous sections have concentrated on distortion of *shape*. Object states define *all* dynamic parameters of the object, including all the states that can be influenced by interaction. Therefore, as each point on the object is delayed by a different amount, *all* dynamic states of the object will change across its surface. For example, if the temperature of an object is increasing over time, a delayed part of the object will be at a lower temperature than the part closer to a local user.

DEMONSTRATION PROGRAM

A movie of a demonstration program, showing the use of the causal surface is available from the ISRG website. The program allows two users to interact with objects within the same world. A simulated communication channel between the users allows variable network latencies.

5. CONCLUSION

This paper proposes an alternative solution to prediction for many of the problems associated with network latency within distributed virtual environments. Remote users, and objects close to them are viewed by local users as they were some time in the past, thereby removing state discontinuities, and allowing visualisation of remote interaction. Objects close to local users are displayed in real-time, such that interaction is possible. Objects are delayed a smoothly varying amount between the local and remote users. State changes are propagated through objects at a rate derived from the network delay to facilitate visualisation of information transfer, and allow real-time dynamic interaction.

Acknowledgement. This work is supported by a Postgraduate Studentship Award from the UK's Engineering and Physical Sciences Research Council.

6. REFERENCES

[1] F. R. Ellis, R. M. Williams, FLAT AND CURVED SPACE-TIMES, Clarendon Press, Oxford, 1988.

[2] P. M. Sharkey, D. J. Roberts, P. D. Sandoz, R. Cruse, "A Single-user Perception Model for Multi-user Distributed Virtual Environments", *Proc. Intl Workshop Collaborative Virtual Environments*, Nottingham, UK, Jun. 1996.

[3] M. D. Ryan, P. M. Sharkey, "Causal Volumes in Distributed Virtual Reality", *Proc. IEEE Intl Conf. Syst Man Cybernetics*, Orlando, Oct. 1997.

[4] J. Lasseter, "Principles of Traditional Animation Applied to 3D Computer Animation", *Computer Graphics*, **21**, *4*, pp. 35-44, July 1987.

[5] D. J. Roberts, P.M. Sharkey, "Maximising Concurrency and Scalability in a Consistent, Causal, Distributed Virtual Reality System, Whilst Minimising the Effect of Network Delays", *Proc IEEE 6th Workshops Enabling Tech: Infrastrcut. Collab. Enterprises (WETICE '97)*, Boston, Jun. 1997.

[6] W.Broll, "Interacting in Distributed Collaborative Virtual Environments", *Proc. IEEE Virtual Reality Annual Intl. Symp. (VRAIS '95)*, North Carolina, Mar. 1995.

[7] G.F. Franklin, J.D. Powell, A. Emami-Naeini, FEEDBACK CONTROL OF DYNAMIC SYSTEMS, Addison-Wesley, 1986.

[8] P.M. Sharkey, M.D. Ryan, D.J. Roberts, "A Local Perception Filter For Distributed Virtual Environments", *Proc. IEEE Virtual Reality Annual Intl. Symp. (VRAIS '98)*, Atlanta, Georgia, Mar. 1998.