

The Effects of Local Lag on Tightly-Coupled Interaction in Distributed Groupware

Dane Stuckel and Carl Gutwin

Department of Computer Science, University of Saskatchewan
110 Science Place, Saskatoon, Saskatchewan, Canada, S7N 5C9
dane.stuckel@usask.ca, gutwin@cs.usask.ca

ABSTRACT

Tightly-coupled interaction is shared work in which each person's actions immediately and continuously influence the actions of others. Tightly-coupled collaboration is a hallmark of expert behavior in face-to-face activity, but becomes extremely difficult to accomplish over distributed groupware. The main cause of this difficulty is network delay that disrupts people's ability to synchronize their actions with another person. In this paper we report on two studies that explore *local lag* as a way of reducing this problem. When applied to visual feedback, local lag synchronizes the visual environments of the local and remote clients, preventing one person from getting ahead of the other. We tested the effects of local lag in several delay conditions: we found that the technique significantly improved performance, and that users did not rate local lag as more difficult or frustrating to use. Our studies improve our understanding of local lag and of how it improves tightly-coupled interaction in distributed groupware.

ACM Classification Keywords

H.5.3 [Information Interfaces and Presentation]: Group and Organization Interfaces—*CSCW*

Author Keywords

Real-time groupware, network delay, coordination.

INTRODUCTION

Tightly-coupled interaction is collaborative activity in which individual actions are closely coordinated, and in which visual feedback from each person's actions immediately and continuously influence others' actions. Tightly-coupled interaction is a hallmark of expert activity in the real world: e.g., one person handing a tool to another must watch the other person's approaching hand and adjust their own movement to match it [21]; similarly, two people playing a fighting game must time their moves to a fraction of a second based on what the other person is doing.

A fundamental characteristic of tightly-coupled interaction is that the coordination is achieved without slowing down

the activity or serializing the task (i.e., adopting a 'you move, then I'll move' strategy). These temporal constraints imply three main scenarios in which tight coupling occurs:

- *Competition*. In real-time competitive activities (e.g., fighting games), any slowdown on one player's part will be exploited by the other; therefore, people must move and respond to others' moves as quickly as possible.
- *External events*. In other situations, the speed of the activity is driven by external events that the group does not control – e.g., new requests arriving at a distribution centre, or new enemies arriving on screen. In these cases, the group must coordinate their actions at the time scale of the external events, which forces tighter coupling.
- *Expert collaboration*. Tight coupling is common even in tasks where people could slow down the interaction. As people become expert in the task and in the behaviour of their partner, they naturally shorten the time between dependent actions, to the point where interaction is tightly coupled. For example, expert interactions in surgical teams or tabletop work exhibit the time scale and responsiveness of tightly-coupled interaction.

Although tightly-coupled interaction is common in the real world, the precision timing and exact movements required for this degree of coordination are extremely difficult to achieve in distributed groupware. This is largely because of the presence of network delay: delay makes it difficult for people to coordinate their actions, because the visual information that shows what other people are doing arrives late, and the shared action quickly become disorganized. The interdependency and the fine granularity of tightly-coupled interaction makes this type of coordination sensitive to latencies as low as 100-200ms [2], which are still frequently seen in Internet applications.

Researchers have suggested several techniques for dealing with network delay in distributed systems (e.g., [4,6,8]). However, most of these techniques are concerned with data consistency – that is, ensuring that the models at different sites contain the same information at the same time. Very little work, in contrast, has been done to determine the effects of these techniques on human interaction, and particularly on highly visual tightly-coupled interaction.

In this paper, we investigate the effects of a latency-compensation technique called *local lag* [15]. Although local lag was proposed as a method for maintaining the consistency of underlying data, it also has potential for

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

CSCW'08, November 8–12, 2008, San Diego, California, USA.

Copyright 2008 ACM 978-1-60558-007-4/08/11...\$5.00.

assisting visual coordination in tightly-coupled human interaction. Latency-based problems in coordinated actions arise because of differences between the visual state of the world for a local and a remote user. Local lag works by re-synchronizing the local visual feedback and remote visual feedthrough, by delaying local feedback so that it is shown at the same time for both local and remote sites.

With both participants seeing the same view of the world, there is less chance of one person getting ahead of the other, and so the activity remains coordinated. However, the cost of the local-lag technique is that local controls are delayed by the amount of network delay, and so can feel sluggish and less responsive to the local user as latency rises.

An earlier study of local lag showed that the technique can improve group performance in the presence of network delay [4]. However, this prior work did not study a true tightly-coupled task, and left several questions unanswered:

- How does local lag affect performance in true tightly-coupled interactions?
- Do users perceive local lag (with its less responsive controls) as causing higher effort or higher frustration?
- How does local lag change people's adaptation to delay, and does experience with games make a difference?
- Is a secondary immediate representation of objects needed for local lag (as suggested by Chen et al. [4])?

To investigate these questions, we carried out two studies. The first experiment tested the effects of local lag on group performance in a tightly-coupled game, and measured participants' subjective perceptions of effort and frustration. The second study compared 'regular' local lag with the 'echo' version proposed by Chen et al. [4], in which a second representation shows the immediate position of the object, in addition to the lagged true position. From these two experiments, we found that:

- Local lag provided substantial protection from the negative effects of network latency in the game task – particularly at latencies up to 200ms.
- The benefits of local lag did not appear to cause greater perception of effort; there was no difference in effort ratings between local lag and immediate feedback.
- Local lag worked equally well for experienced online gamers and for less-experienced participants.
- The 'echo' representation had no effect on performance.

Although these results need to be explored in other tasks and with more realistic groupware systems, they suggest that applying local lag to visual feedback can improve the usability of real-time distributed groupware. The amount of delay that is regularly seen in real-world networks often makes tightly-coupled work impossible; local lag shows promise for making these activities feasible. The ability to support tightly-coupled interaction will help distributed groupware towards its goal of making distributed work feel as natural and efficient as face-to-face interaction. The contribution of our studies is that they add to our understanding of when and how local lag works. Our studies are the first to look at perception of effort with local

lag, the first to report additional observations and analyses of the technique, and the first to provide evidence in opposition to claims about echo representations.

In the following sections, we review related research and describe how local lag works when applied to visual feedback. We then report on the two studies that we carried out to test the technique in a distributed groupware system.

RELATED WORK

Our interest in exploring local lag arises from previous work on the effects of delay on collaboration, and from related research on techniques for dealing with delay.

Network delay and its effects on collaboration

Delay is a fact of life in real-world distributed applications because information must be transmitted across a network and processed at the other end before it can be displayed [9]. There are two main types of delay: latency and jitter. *Latency* is the elapsed time between an event's occurrence and its display on a remote system; latency causes people's actions to be seen after they actually occur. *Jitter* is variation in latency, caused by network traffic or processing delays; jitter causes halts and jerks in the display of a remote user's movement.

Delays can have severe effects on collaboration – on coordination, communication, and understanding of the shared situation [2,9,18,22]. Delay can make turn taking difficult to negotiate, can hinder social locking protocols, and can cause inconsistencies that lead to confusing roll-back actions. Delay may also cause users to disagree over the timing or simultaneity of key events, and may lead to missed causal links or wrongly-inferred dependencies.

When interaction occurs more quickly (as in the case of tightly-coupled tasks), smaller amounts of latency are more likely to cause problems. In a virtual ping-pong game that has large interaction granularity, users did not seem to perceive less than 150 ms latency, and were able to keep playing up to latencies of 500 ms [22]. In contrast, a study of a racing game, where players interact more quickly, showed that latency above 100ms made the game difficult [18]. Similarly, studies of first-person shooter games – where people must hit moving targets or act in concert with another person – show that delay above 100ms can significantly reduce user performance, and latency greater than 150ms feels extremely sluggish [2].

Jitter has also been shown to have significant effects on people's ability to predict others' movement [9], but does not always affect users' perceptions in game environments [19]. In our explorations, we focus primarily on latency, but consider the implications of jitter later in the discussion.

Techniques for Dealing with Network Delay

A number of strategies have been proposed to deal with the problems caused by network delay. These techniques come from research into distributed simulations, networking, and groupware, and can be organized by their general approach: hiding delay, revealing delay, or adding delay.

Hiding Delay

Several techniques attempt to maintain the illusion that there is no delay in the environment. Prediction is one main strategy – if the local client can accurately predict the positions of remote users to compensate for the network delay, then it appears there is no delay at all. Accurate prediction models are difficult to achieve, however, and work best only when actions are regular and things move slowly. Dead reckoning is a common prediction model that calculates future positions based on past position, velocity, and trajectory [1,11]. Dead reckoning works well for objects that have high inertia (such as vehicles), but works poorly for quicker objects such as telepointers [11].

Local Perception Filters [20] take a different approach: they warp the shared environment to hide the effects of delay. In this scheme, visual effects provide extra time for network messages to reach remote clients. For example, bullets might appear to slow down as they approach the avatar of a lagged user, but speed up near ships with less delay.

Time Warp [14] is a third strategy – in this approach, clients allow local actions to occur optimistically (as if there is no delay), and then when conflicts occur, the system rolls back to a previous consistent state. Although Time Warp has been recommended for fast-paced games [5], the use of rollback can be distracting and confusing; locations of shared objects change suddenly, and users experience gaps in the ordering of events [10].

Revealing Delay

Users have the ability to adapt to a changing environment, and in some cases can adjust to compensate for network delay. If users are to adapt their behaviour appropriately, however, they need information about current network conditions, and several methods have been suggested for revealing delays to the user.

Simple displays such as latency indicators can show the current amount of network delay. These are now common in many games, where information such as ping times show the delay for individual clients. Studies have shown that users are able to perform better when they are more informed in this way [7,10]. *Decorators* are an extension of this idea, and place indications in the game environment itself [22]. A variety of decorators have been designed to show information such as current latency or jitter, past and predicted future states, or reliability. These techniques have also been shown to improve user performance [10,22].

Mechanisms for revealing delay to the user can also be more subtle. Metaphors such as the weather inside a shared world can be used to reveal the state of the network – e.g., rainy weather could indicate poor network conditions [17].

Adding Delay

Some techniques deal with network delay by adding latency to various points in the distributed system. Adding delay to the receiver in the form of a buffer is a common method for dealing with jitter; but there are also ways of using extra delay to address problems of latency.

Bucket Synchronization [8], Delta-causality Control [13], and Local Lag [15] are techniques that ensure everyone in the group experiences the same amount of delay, which achieves fairness and consistency between the clients. These techniques include time information in messages, either by separating messages into turns or by including timestamps; the system can then determine when receivers should process those messages to achieve consistency. A client processes any messages that it has received at the appropriate time (or turn), keeping clients in synchrony.

As mentioned above, these techniques are concerned with data consistency, not user coordination. In addition, they are not immediately appropriate to the problem of tightly-coupled interaction – for example, the techniques add delay to make everyone as slow as the slowest client, even though tightly-coupled interaction may only be occurring between two participants. In the remainder of the paper, we assume that the delay of local lag is applied only to the visual feedback between two people who are engaged in tightly-coupled interaction.

Previous Studies of Local Lag

Although local lag mechanisms have been implemented in numerous systems including first-person shooter games [16], we know of only one study that has looked at the way that local lag affects human interaction [3,4]. This study tested an augmented local-lag technique that adds a secondary representation of local objects to the local view. This ‘echo’ representation is not lagged, and therefore is intended to provide immediate feedback to the user as they carry out actions in the environment.

Although this work showed that the augmented technique improved performance over a regular delayed network, there are limitations to this study that motivated our work:

- The task used in the study was not a true tightly-coupled task. Participants could adopt a turn-taking strategy, and there was therefore no need for the continuous and immediate adjustment to the other person’s motion that is a fundamental part of tightly-coupled interaction.
- The study only compared the local-lag+echo technique to the regular delayed network, but did not compare to the regular local-lag technique; therefore, the value of the ‘echo’ representation is as yet untested.
- The study did not investigate the issue of lagged local controls; there is still no understanding of whether the benefits of local lag cause undue frustration or effort.

The experiments that we report below are designed to address these limitations, and help to improve our understanding of when and how local lag works for tightly-coupled interaction. Before describing our experiments, we summarize the way in which local lag, when applied to visual feedback, can assist tightly-coupled coordination.

LOCAL LAG FOR VISUAL FEEDBACK

When applied to the visual feedback of a user’s actions, local lag aids coordination by re-synchronizing the multiple copies of feedback that are displayed at different sites. We

distinguish between *feedback*, the visual information displayed to the local user as a result of their actions (e.g., the movement of their telepointer), and *feedthrough*, the same information displayed to a remote user.

Feedback and feedthrough, and the times at which they are displayed, are important when people engage in tightly-coupled interaction. Both the local and remote users need the visual information produced by each person's actions as input to their next actions – the idea is similar to that of closed-loop motor control, except that in tightly-coupled interaction, both the local feedback and the remote feedthrough are used as input. Figure 1 shows a diagram of the action-feedback cycle for a single user (left), shows how the cycle becomes a continuous process over time (middle), and shows how both users in a tightly-coupled interaction use feedback (right). Note that there is no network delay shown in this example.

With this representation, we can show how network delay affects tightly-coupled interaction (Figure 2, left). The diagram shows several iterations of the action-feedback loop for the local user, and shows that the feedthrough from the first action arrives late at the remote site. At time T_1 in the diagram (at the dashed line), it can be seen that the local user has carried out a second action by the time that the remote user sees the results of their first local action – therefore, they now have different views of the state of the world. This situation results from what we call an 'immediate feedback' display policy.

Local lag does not immediately display feedback to the local user. Instead, it displays local feedback at approximately the same time that the feedthrough is displayed at the remote site (Figure 2, right). (The timing is accomplished by monitoring the current network latency, and so the synchronization will not be exact). Delaying the local user's feedback makes an important difference: since the local user needs the feedback to determine their next action, they do not move ahead until the remote user also sees the visual result of their last action. This means that both the local and remote users start their next actions at the same time, and based on the same view of the state of the world. It is the timing of the two people's actions that is important in successful coordination, and local lag prevents either of them from getting ahead of the other. However, the delayed feedback means that the local user's controls become less responsive and more sluggish.

Figure 3 shows an example of the difference between immediate feedback and local lag, using a simple game scenario. In the game, the local user moves a fire truck (the rectangle at the bottom of each screen) left and right; the remote user moves the direction of the water spray towards a fire (the circle at the top of each screen). The left side of the figure shows what happens with immediate feedback. The local user starts moving to the right in frame 1, and immediately sees their own feedback (frame 2). However, this visual information has not yet reached the remote user, and so they do not adjust the water angle. Assuming that

person 1's machine determines the score, the players are now losing points – but the remote user has no idea that anything is wrong.

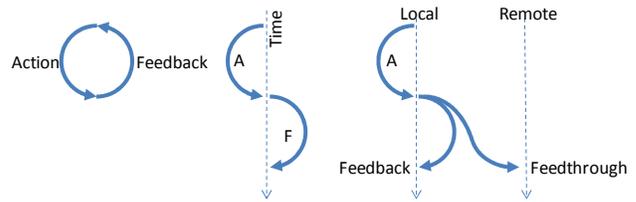


Figure 1. Action-feedback cycle (left); the cycle over time (middle); feedback and feedthrough (right).

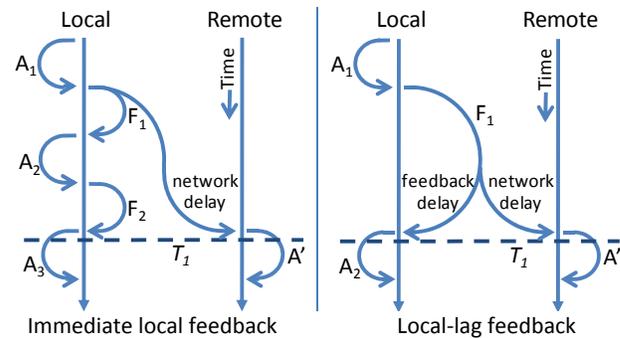


Figure 2. The effects of delay when using immediate feedback (left) and when using local lag (right).

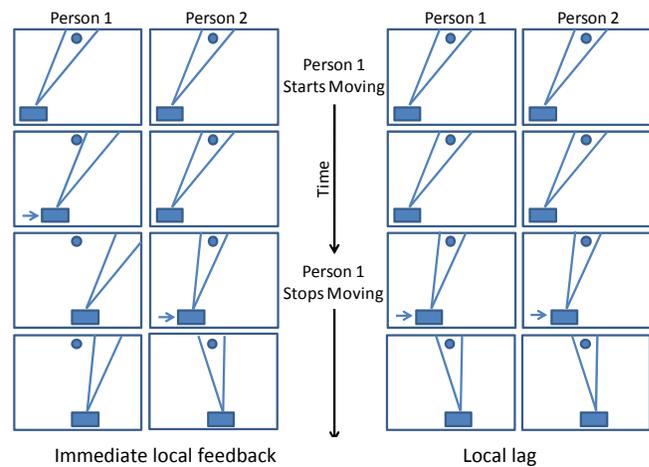


Figure 3. Comparison of immediate feedback and local lag.

This discrepancy does not happen when local lag is applied (right side of Figure 3). When the local user starts moving, the local machine holds the feedback until it also arrives at the remote site. This means that both people see the truck start moving at the same time (in frame 3), and means that the remote user's actions are correct on both machines.

Local lag works by changing the location, in the process of coordinated action, where delay must be dealt with by the users. Whereas immediate feedback requires that users deal with delay in the coordination of actions, local lag moves the issue to the local input controller – that is, coping with delay becomes a task of dealing with sluggish controls, rather than attempting to simulate what is happening on the other person's screen. Users therefore adapt to the network

delay by dealing with the feedback delay – something that they can interact with and experience.

In addition, local lag ensures that both users' screens show (at least approximately) the same view of the world, which aids in communication about the task. One of the main problems with divergent views is that people have difficulty discussing the world, and local lag allows people to be much more confident that the other person will understand what they say and their references to objects in the world.

EVALUATION

We carried out two studies to answer six questions:

1. How is tightly-coupled interaction affected by latency?
2. How does local lag affect performance in a tightly-coupled interactive task?
3. Do the delayed controls caused by local lag affect perception of effort or frustration in the task?
4. Does local lag affect different task roles differently?
5. Does experience with games and network delay change the effect of latency or local lag?
6. Is the secondary 'echo' representation required for local lag to work, as suggested by Chen and colleagues [4]?

STUDY 1: PERFORMANCE AND PERCEIVED EFFORT

The first experiment measured the effects of lagged and immediate feedback, at several levels of latency, on group performance and perceptions of effort and frustration.

Participants

Twenty-four participants (7 female, 17 male) were recruited for the study from a local university, and ranged in age from 18 to 33 years (mean 22.4). All were regular computer users (mean 30 hrs/wk); 13 people described themselves as highly experienced players of multiplayer online games, and 12 people reported that they had experienced substantial network delay in online gaming situations.

Task

The study task was a simple multiplayer game in which participants controlled either the driver or the turret of a 2D fire truck. The shared task was to keep a stream of water on a fire for as long as possible (Figure 4). There were specific jobs for the driver and the turret operator. The driver's job was to collect water tanks by driving in front of them; when a tank was collected, its water was added to the fire truck's supply, and a new tank appeared somewhere else on the screen. The driver used the arrow keys on the keyboard to move the truck left and right. The turret operator's job was to keep the stream of water pointed at the fire, and they aimed the stream by moving the mouse.

Although the task was simple, it was still a true tightly-coupled activity – the turret operator had to continuously adjust the angle of the water stream to respond to the movement of the truck, and the driver could not stop, since the truck would quickly run out of water.

Procedure and Apparatus

Participants carried out the task in groups of two. Participants were seated at different computers in the same room, separated by a divider so that they could not see each

others' screens or keyboards. Participants were allowed to talk freely and were encouraged to discuss the task and strategies for collaboration. Verbal communication, however, was not intended as the primary means for coordination in the task – the task was designed so that the visual information on the screen would be the main source of information for coordinating the activity.

Each session was divided into two blocks, one using local lag and one using immediate feedback. Each block consisted of six two-minute trials in which participants carried out the fire-truck task; each trial used an increasing level of delay. The first thirty seconds of every trial was considered training time, to allow the users to familiarize themselves with the task at each level of delay. There were short breaks between trials, during which participants completed a NASA task load index (TLX) questionnaire [12], for the trial they had just completed. Delay levels always increased from 0ms to 500ms, but the order of the two blocks (local lag and immediate feedback) was balanced. In the second block, participants also switched roles (driver and turret operator).

The study used custom-built experimental software, and ran on two Windows XP computers. Each computer had a 15-inch monitor set to 1024x728 resolution, an optical mouse, and a standard keyboard. The machines were connected via a dedicated router. The study application was built in C# using the GT toolkit and DirectX; the system maintained a frame rate of approximately 60 fps.

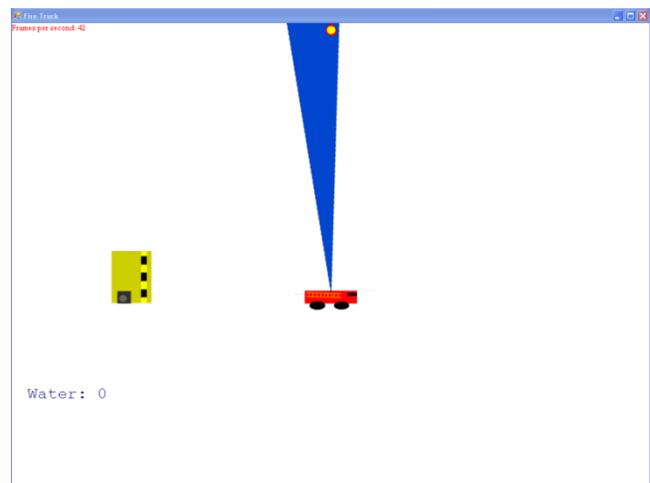


Figure 4. Fire-fighting task used in the study. One user moved the truck left and right to collect water tanks (shown at left); the other user controlled the water stream with the mouse.

Simulated Latency

There were varying amounts of delay between the two computers during the task. Participants were introduced to the idea of network latency, and were informed that their screen might not look exactly like their partner's because it took time for information to travel between their computers.

The first trial in each run had zero added network delay, so that the users could learn the task quickly (there was approximately 20ms of real latency between the clients).

Each successive trial then had a progressively larger amount of delay. Latency was simulated by holding messages at the server for a specified amount of time.

Study Design

The study used a within-participants full factorial design, with two factors: *feedback scheme* (local lag or immediate feedback) and *latency* (0, 100, 200, 300, 400, or 500 ms). Dependent variables were the groups' accuracy during the task (i.e., the fraction of the trial that they were able to keep the water stream on the fire), and individuals' subjective perception of effort from the TLX questionnaires.

STUDY 1 RESULTS

We first report the effects of latency and feedback scheme on performance and perception of effort, and then report on additional analyses of task role and game experience.

Effects of Delay on Performance

Performance in the task was measured in terms of accuracy – the fraction of the trial that groups were able to maintain the goal state (i.e., keep the stream of water on the fire).

Overall, accuracy decreased as network delay increased. As can be seen in Figure 5, average accuracy over all conditions dropped from nearly 95% at 0ms delay, to less than 50% with 500ms delay. A 2x6 ANOVA showed a significant main effect of delay amount ($F_{5,55}=101.7$, $p<0.001$). This supports earlier findings that performance in coordinated activities is strongly affected by delay.

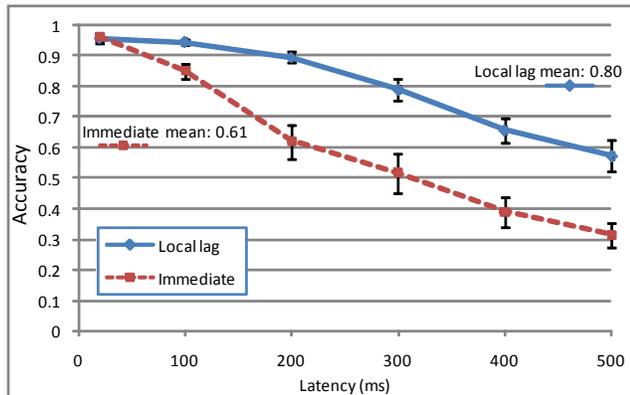


Figure 5. Mean accuracy (±SE) by latency and scheme.

Effects of Feedback Scheme on Performance

A 2x6 ANOVA also showed a significant main effect of feedback scheme ($F_{1,11}=89.3$, $p<0.001$). The mean accuracy of local lag over all levels of delay was 0.80, and of immediate feedback was 0.61 (see Figure 5).

In addition, there was a significant interaction between latency amount and feedback scheme ($F_{5,45}=7.904$, $p<0.001$). The interaction arises because (as expected) there was no difference between immediate feedback and local lag at 0ms latency, and the difference was small at 100ms. The advantage of local lag is greater at higher levels of delay (see Figure 5). The largest difference between feedback schemes was at 200ms delay, and corresponds with the largest drop in performance for the immediate-feedback condition. The average performance of immediate

feedback dropped 34% between 0 and 200 ms, whereas performance dropped only 7% when using local lag.

Effects on Subjective Perception of Effort

We measured participants' subjective perception of effort using the NASA TLX, which asks participants to rate mental and physical demand, overall effort, frustration, and perceived performance on a seven-point scale [12]. The questionnaire results showed that participants felt that the tasks became more difficult as latency increased, and this effect was significant for each of the TLX questions (using one-way ANOVA, all $p<0.05$).

However, there was very little difference in responses between the two feedback schemes: as can be seen from Figure 6, participants felt that the task was no more difficult or frustrating with one feedback scheme or the other and felt that their performance was approximately equal in both conditions. Separate one-way ANOVAs showed no significant effect of feedback scheme for any of the TLX questions (all $p>0.05$).

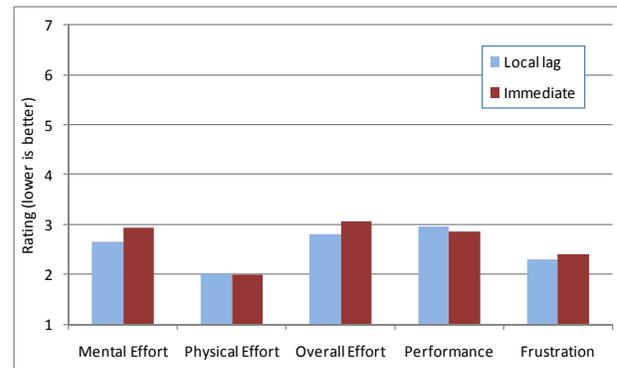


Figure 6. Mean TLX ratings by feedback scheme.

Post-Hoc: Differences Between Game Roles

The driver and the turret operator had considerably different roles, with different control tasks and different input devices. We wanted to see whether local lag had any differential effect on these two roles, so we carried out a post-hoc exploratory analysis using the TLX ratings. There were an equal number of each role and each feedback scheme in the two groups. The driver appeared to have the easier of the two jobs in the game – drivers gave significantly lower (i.e., more positive) ratings for physical effort ($p=0.023$), overall effort ($p=0.033$), and frustration ($p=0.023$). However, there was no interaction with feedback scheme for any of the TLX measures (all $p>0.05$), indicating that the effect of local lag on perception of effort was not dependent on the person's task role.

Post-Hoc: Effects of Prior Experience with Games

Half of the 12 groups were made up of participants who described themselves as highly-experienced players of online games. We investigated whether gaming experience affected people's performance or perceptions.

Performance. Although participants with game experience were slightly more accurate overall (74% to 68%), one-way ANOVA did not show any significant effect of experience

on accuracy ($F_{1,10}=1.57$, $p=0.24$). There was also no interaction with feedback scheme ($F_{1,10}=0.34$, $p=0.57$).

Perceived Effort. A one-way ANOVA did not show any significant differences, on any of the TLX ratings, between experienced and less-experienced participants (all $p>0.05$).

Observations and Participant Comments

Our observations suggest that people responded to delay differently when local lag was used. High levels of delay with immediate feedback led to users commenting that the display did not make sense, or that they did not know how to carry out the task. These comments did not arise when people used local lag; instead, local lag made it increasingly difficult to control the truck or the water jet.

Immediate feedback at latency greater than 200ms led to considerable confusion: participants regularly commented that they did not know what was going on with their partner, or that they did not understand what was happening on screen. In several cases the turret operator would have the stream aimed at the fire, but the fire continued to burn, which was confusing and annoying for several participants. One person stated that this made the task “quite difficult - the water on my screen didn’t put out the fire. I relied on the driver’s instructions to put out the flames.”

Several groups found strategies for working with the delay; for example, moving more predictably: as one participant said, “I would pause before I turned so my partner would be able to remain on the fire.” However, communicating these strategies was often difficult. One person stated “I told the person on the hose to compensate for the lag by pointing ahead of the fire, but this took the most effort to explain.”

With local lag, there were far fewer apparent problems with confusion or miscommunication. As latency increased, however, the reduced responsiveness of the local controls became obvious: for example, one participant commented “the truck lagged in response to the change of direction, so you had to change direction slightly before;” another stated “the mouse and the water had a gap in reaction.” Some participants (particularly in the driver role) found it relatively easy to adjust to the delayed controls (e.g., “it took a couple of rips across the screen to get the timing down for it to reverse directions”), but at higher latencies, the controls lag became severe, especially for the turret operator: “the last one was very difficult—the mouse was touchy and did not seem to coordinate with the stream.”

STUDY 2: LOCAL-LAG VS. LAG+ECHO

We carried out an additional study to test claims that a secondary immediate representation of local objects is necessary for the local-lag technique to be effective. We built a second version of the study system described above that included an ‘echo’ representation of the local object (either the fire truck or the water stream) that was not lagged. This means that there was both an immediate representation (the echo) and a lagged representation (the ‘real’ object) for the local user. The echo representation was coloured red and made semi-transparent.

We used the same apparatus and procedures as in study 1, except that we added a third condition – local lag + echo. This meant that there were now three levels of the factor *feedback scheme* (immediate, local lag, lag+echo). Additional training was done before the echo condition, to explain to participants what the dual representations meant.

We recruited twelve participants for this study, none of whom had participated in the first experiment. Since the task was done in pairs, this meant six data points per condition for the performance measure, and twelve data points per condition for the TLX ratings.

Study 2 Results

We tested the effects of feedback scheme on performance and on subjective TLX ratings.

Effects of feedback scheme on performance. There was a significant main effect of feedback scheme on accuracy ($F_{2,10}=24.3$, $p<0.001$). Post-hoc t-tests showed that there were significant differences between the ordinary local-lag scheme and both the immediate and the lag+echo schemes (both $p<0.05$). The performance of ordinary local lag was superior to both of the other techniques at all levels of delay above 0ms (Figure 7). There was no significant difference between immediate feedback and the lag+echo scheme.

Effects of feedback scheme on perceived effort. Figure 8 shows mean ratings for each of the TLX questions. One-way ANOVAs did not find effects of feedback scheme for any question (all $p>0.05$). As with experiment one, ratings of all of the feedback schemes was roughly similar.

Observations and Participant Comments

The main observational result is that several of the participants were distracted or confused by the double representation of objects on the screen, despite earlier training to explain what each representation meant. Two main problems were observed. First, several participants simply used the immediate echo representation as their primary feedback, and ignored the ‘real’ lagged representation. This meant that their task behaviour was identical to the immediate-feedback scheme.

Second, some participants thought that both representations were active, rather than being different temporal versions of the same object. This led to some confusions about the task – for example, one participant said that they liked this condition “because the water was wider;” another said that they had to be careful keeping track of “which truck filled up with water;” a third said that “keeping both sprays on the fire was difficult.” Since there was only one ‘real’ representation (the lagged version), these confusions led to increased errors and reduced performance.

Third, several participants found the double representation distracting. As one participant stated, “[this condition] was very ineffective...only one stream did anything and the other was simply a distraction.” Another said that “I had to ignore the red [immediate] truck and just concentrate on the green [lagged] truck,” which essentially reduced the task to local lag, but with extra information that had to be ignored.

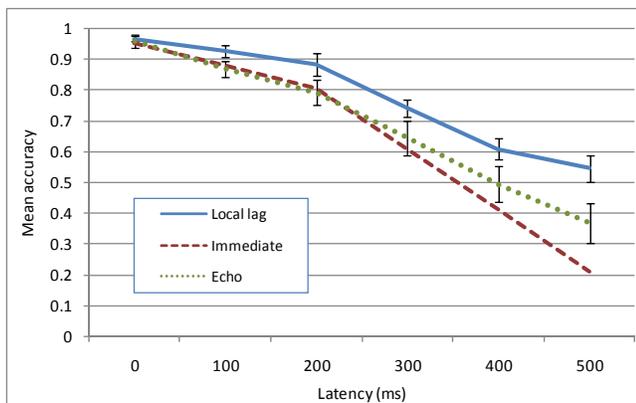


Figure 7. Accuracy of local-lag, lag+echo, and immediate feedback at different levels of latency.

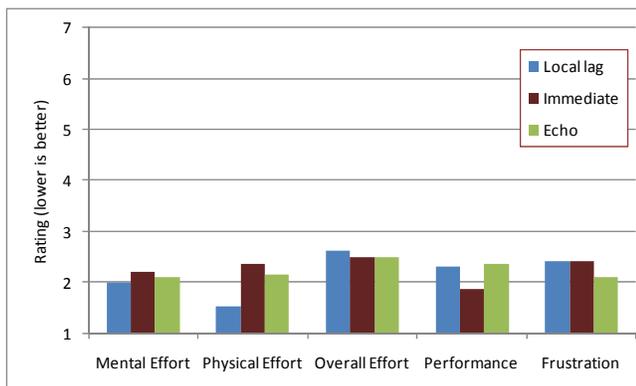


Figure 8. Mean ratings from TLX questions (lower is better)

DISCUSSION

There are five main findings from the studies:

- Latency is a major problem for tightly-coupled interaction: 200ms reduced accuracy in our task from 95% to 61%; 500ms reduced accuracy to 34%.
- In the fire-truck game, local lag protected tightly-coupled interaction from the negative effects of latency.
- Local lag did not appear to incur subjective costs: participants' perception of effort and frustration was no different between local lag and immediate feedback.
- Local lag benefited both task roles equally, and also worked equally well for gamers and non-gamers.
- The addition of an 'echo' representation did not improve the performance of local lag; accuracy with the echo technique was lower than with ordinary local lag.

Below, we provide explanations for our results, discuss several issues in the deployment of local lag for groupware, and outline several directions for future work in the area.

Explanations of Results

Why did local lag work well for tightly-coupled tasks?

People were able to cope with network delay more effectively with local lag. There are three possible reasons for this improvement. First, the re-synchronization of feedback and feedthrough information appeared to provide substantial benefit for coordination. Having the same view of the world was extremely useful for communication, and

having a view of the world that was accurate enabled turret operators to maintain a consistent causal understanding of what was going on in the environment.

Second, it appears that users find it easier to adapt to lagged controls than to adapt to delayed visual information. Although people did notice the reduced responsiveness of the mouse and keyboard, the problems were purely local, and did not cause the interaction to become uncoordinated.

Third, the reduced responsiveness of local controls may have also led participants to behave in a more careful, and therefore more predictable, manner. Several participants commented on the 'touchiness' of the controls, and that they had to work them more gingerly than they normally would have done.

Why were there no differences in subjective effort?

Although participants clearly felt that the task became more difficult with increasing latency, there were no significant differences in perception of effort between local lag and immediate feedback. We believe that this is because local lag does not make the interaction more difficult, it just moves the difficulty to a new location – that is, to the local input controller. Although the sluggish controls clearly became a problem as latency increased, the difficulty was balanced by easier interpretation of the visual environment.

This result shows that performance can be improved without adding substantially to the user's perception of effort or frustration. The idea behind the technique (of shifting the location where users must deal with delay) may also be applicable elsewhere – there may be other situations in which human perception of network phenomena can be used to reduce problems and improve performance.

Why did the 'echo' version perform poorly?

There are two reasons why the dual-representation version of local lag did not perform well in our study. First, the extra visual object confused some participants, even after training on the representation. It was clear that for some people, it was difficult to understand that the two objects represented the same thing, but at different points in time.

Second, it is difficult for users to understand which of the representations they should be using for feedback – the echo object or the lagged object. The intention of the echo representation was to provide users with immediate feedback to assist them in executing their actions; however, if participants in our task used only the immediate representation, the technique degenerated to become the same as the original immediate-feedback scheme.

We believe that in tasks where both users move at the same time, the echo representation will never be useful. If the local user must continuously adapt their motion to the movement of a remote user, then the immediate echo will always be out of synchrony with the other person's actions – and will therefore lead to errors in coordination.

As an example, the diagram in Figure 9 shows the problem in the context of the fire-truck task. As long as the driver

continues to move the truck to the right, the turret operator will never be able to use the echo version of the water stream to target the fire – the ‘real’ stream will always lag behind, and never reach the target.

We believe that an echo is only useful when it is used in relation to a non-moving (or non-delayed) object; if the other object is not moving, then the interaction is not tightly coupled.

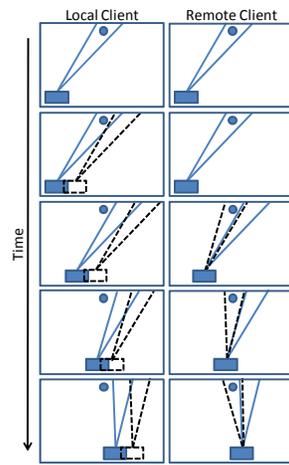


Figure 9. Study task with echo representation (dotted lines).

Deployment Issues for Local Lag in Groupware

Our experiences have identified several issues that may affect actual deployment in real-world groupware systems:

When to apply local lag. Tightly-coupled interaction may not occur throughout a collaborative session; therefore, systems could turn the technique on and off as needed. This approach provides better overall responsiveness for the local user, since the lagged local controls would only be required at certain times. In this approach, the system would need an indicator for the presence of tight-coupled interaction – one possibility could be to use the proximity between avatars. A question for further study is that of how users will adapt to low lag during normal operation, and higher lag during tightly-coupled interactions.

Hiding local delay. In some applications, local lag on discrete actions could be hidden by using visual effects. For example, local lag in a fighting game means that a punch command will not result in immediate visible action. The system could hide this delay by using effects such as an animated ‘wind-up’ for the punch: the command would be sent to the other player immediately, but the local execution lag would be made more explainable by the animation. The animation would not be played at the receiver’s end, since the receiver does not have the extra time.

Variable delay and jitter. A related issue is how local lag will perform in situations where latencies change. When people adapt to delay, they are best at dealing with a fixed amount; it is possible that people will have difficulty adapting to variable delay (we note that this is a problem for all approaches, however). Variance can be smoothed by buffering, but comes at the cost of adding extra latency.

Scaling to more than two people. In general, there are rarely more than two people in a tightly-coupled interaction, and we believe there are very limited requirements for scaling the technique beyond two people. In these situations, however, a reasonable strategy is to use the mean latency of remote clients as the delay for local feedback.

System complexity. Local lag will increase the complexity and development time of a groupware system. The technique delays only local feedback while remote actions are displayed immediately, so the logistics of what to draw on the screen can be problematic for the software designer. The simple solution of directly buffering the input cannot always be used in applications that also send the results of the user’s input (e.g., kills or movement resulting from in-game physics), because the results of user input are not known until the input is applied to the shared environment. Implementing local lag in such a system is not impossible, but designing a systematic way to implement the technique in complex applications is an area of future work.

Generalizability of Results to Other Systems and Tasks

Our studies involved an artificial task, and one main issue that must be considered is how the results will generalize to other collaboration situations and other tasks. To begin, we believe that any potential problems of generalization will involve the slow local controls, not the network delay itself – since local lag synchronizes the two systems, it is unlikely that problems with the technique will be based on delay; rather, they will involve local difficulties in dealing with sluggish controls. There are several factors that should be considered when considering local lag for other situations:

Expected responsiveness. People’s expectations about how quickly objects should respond can alter the experience of working with local lag. For example, the driver of the fire truck may make assumptions about the inertia of the truck, helping to explain the delay when changing directions; in contrast, the turret operator likely expects greater responsiveness from the system, and could be more frustrated by delayed controls. In general, the higher the expectation of quick responsiveness, the more problems will be caused by local lag. In future work, we will further explore the technique with highly-responsive objects such as telepointers, to investigate the limits of this issue. We note, however, that the difficulties of working with local lag may still be better than the alternative: in our studies, even though the water jet was a fast-response object, turret operators still performed better with local lag than they did with immediate feedback.

Situations where local feedback is critical. Some tasks have strict requirements for the accuracy or precision of the local user’s actions – for example, executing a handoff action at a very specific point in space. Local lag makes it more difficult for local users to control their actions – causing, for example, overshooting errors in our tasks. If local accuracy is critical, local lag will cause considerable problems. However, if the tasks are also tightly coupled, designers will have to determine whether the individual or the group needs to have better information.

Long- vs. short-duration actions. Our task involved continuous tightly-coupled action over an extended period (a minute at a time), but other tightly-coupled tasks such as handoff are interactions that occur over a much shorter duration. Short-duration actions provide less time for

people to get used to the lagged controls, and if local lag is only applied during infrequent tightly-coupled episodes, it may be difficult for users to adjust to the lag.

Perceived unfairness. The use of local lag can make it seem to the local user that only their actions are slowed down by local lag. For example, if the ‘windup’ animation discussed earlier is used, it will seem that local commands take extra time to execute, but remote actions happen much more quickly. Users are able to learn about and adapt to delay phenomena, but it is not clear whether they will accept these types of perceived differences in the system.

CONCLUSIONS AND FUTURE WORK

Tightly-coupled interaction is common in many kinds of face-to-face activity, but is difficult to accomplish in distributed groupware. The main cause of this difficulty is network latency that disrupts people’s ability to coordinate shared actions. Local lag has potential to reduce this problem by re-synchronizing local feedback and remote feedthrough. We evaluated local lag’s effects on user performance and subjective perception of effort, and found that the technique leads to significantly higher performance without an increase in perceived effort. We also showed that an immediate ‘echo’ representation is not useful for tightly-coupled tasks where both users move at once.

Further study of this technique is needed – but with an increased understanding of how local lag works, groupware designers can raise the threshold of usability for distributed groupware. Local lag will be particularly valuable for applications that see problems with latencies of 100-200ms – delays that are common on the real-world Internet.

In future work, we plan to explore several of the research directions raised here. We will carry out additional studies of the technique with other tasks and other input devices, and explore the effects of variable latency on people’s adaptation abilities. We also plan to simplify the application programmer’s job in implementing local lag, by adding the technique to a groupware toolkit. Finally, we will deploy local lag in realistic groupware systems and carry out longer-term studies of the technique in actual use.

REFERENCES

1. Aggarwal, S., Banavar, H., Khandelwal, A., Mukherjee, S., and Rangarajan, S. Accuracy in dead-reckoning based distributed multi-player games. *Proc. NetGames 2004*, 161-165.
2. Beigbeder, T., Coughlan, R., Lusher, C., Plunkett, J., Agu, E., and Claypool, M. The effects of loss and latency on user performance in Unreal Tournament 2003, *Proc. NetGames 2004*, 144-151.
3. Chen, H., Chen, L., Chen, L., and Chen, G., Effects of local-lag mechanism on task performance in a desktop CVE system. *J. Com. Sci. Tech.* 20, 3, 2005, 396-401.
4. Chen, L., Chen, G., Chen, H., March, J., Benford, S., and Pan, Z., An HCI method to improve human performance reduced by local-lag mechanism, *Interacting with Computers*, 19(2), 2007, 215-224.
5. Claypool, M. and Claypool, K., Latency and player actions in online games. *CACM*, 49, 11, 2006, 40-45.
6. Cronin, E., Kurc, A., Filstrup, B., Jamin, S., An efficient synchronization mechanism for mirrored game architectures. *Multimedia Tools Appl.*, 23, 1, 2004, 7-30.
7. Fraser, M., Glover, T., Vaghi, I., Benford, S., Greenhalgh, C., Hindmarsh, J., and Heath, C. Revealing the realities of collaborative virtual reality. *Proc. CVE 2000*, 29-37.
8. Gautier, L., Diot, C., and Kurose, J., End-to-end transmission control mechanisms for multiparty interactive applications on the Internet. *Proc. IEEE Infocom 1999*, 1470-1479.
9. Gutwin, C. The effects of network delay on group work in shared workspaces, *Proc. ECSCW 2001*, 299-318.
10. Gutwin, C., Benford, S., Dyck, J., Fraser, M., Vaghi, I., and Greenhalgh, C. Revealing delay in collaborative environments. *Proc. CHI 2004*, 503-510.
11. Gutwin, C., Dyck, J., and Burkitt, J., Using Cursor Prediction to Smooth Telepointer Jitter. *Proc. ACM Group 2003*, 294-301.
12. Hart, S., and Staveland, L., Development of NASA-TLX (Task Load Index): Results of empirical and theoretical research., in Hancock, P., and Meshkati, N., eds., *Human Mental Workload*, North Holland, 139-183.
13. Ishibashi, Y., Nagasaka, M., and Fujiyoshi, N. Subjective assessment of fairness among users in multipoint communications. *Proc. ACE 2006*, 69.
14. Jefferson, D. Virtual time. *ACM Trans. Programming Language Systems*, 7, 3, 404-425.
15. Mauve, M. Consistency in replicated continuous interactive media. *Proc. CSCW 2000*, 181-190.
16. Mauve, M. How to keep a dead man from shooting. *Proc. IDMS 2004*, 199-204.
17. Oliveira, M., and Crowcroft, J. Perceptual network metaphors: Breaking the network transparency paradigm, *Proc. MIPS 2003*, 207-221.
18. Pantel, L. and Wolf, L. On the impact of delay on real-time multiplayer games. *Proc. NOSSDAV 2002*, 23-29.
19. Quax, P., Monsieurs, P., Lamotte, W., De Vleeschauwer, D., and Degrande, N. Objective and subjective evaluation of the influence of small amounts of delay and jitter on a recent first person shooter game. *Proc. NetGames 2004*, 152-156.
20. Smed, J., Niinisalo, H., Hakonen, H. Realizing the bullet time effect in multiplayer games with local perception filters. *Comput. Networks*, 49, 1, 2005, 27-37.
21. Svensson, M., Heath, C., and Luff, P., Instrumental action: the timely exchange of implements during surgical operations, *Proc. ECSCW 2007*, 41-60.
22. Vaghi, I., Greenhalgh, C., and Benford, S. Coping with inconsistency due to network delays in collaborative virtual environments. *Proc. VRST '99*, 42-49.