CONCEALING NETWORK DELAYS IN DELAY-SENSITIVE ONLINE INTERACTIVE GAMES BASED ON JUST-NOTICEABLE DIFFERENCES

Jingxi Xu and Benjamin W. Wah

Department of Computer Science and Engineering
The Chinese University of Hong Kong
Shatin, Hong Kong
{jxxu@cse.cuhk.edu.hk, bwah@cuhk.edu.hk}

ABSTRACT

Online delay-sensitive games with fast interactive actions, like fighting games (FTG) and sports games, require the synchronization of coupled multi-player actions. With network impairments, action commands from other players can be delayed or lost, leading to compromised real-time perception of these games. In contrast to slower strategy games, online fighting and sports games studied in this paper require real-time judgment and instant feedbacks. Traditional methods for optimizing the delay effects of these games are focused on quantifying round-trip delays, without examining the perceptual effects of players. In this paper, we develop a new criterion using just-noticeable differences (JND) for optimizing the duration of actions and responses. Our approach aims to reduce the probability of players perceiving the delay effects, when compared to a reference game with zero network delay. Using statistics collected in offline subjective tests, the timing of actions is modified at run time. Experimental results show significant reduction of players' awareness of network delays using our approach when compared to existing delay-concealment schemes.

Index Terms— Games, Internet, human factors, real-time, delay, quality of experience.

1. INTRODUCTION

With the availability of fast and reliable Internet, online games have changed the way they were played in the past decade. Instead of gathering in a room connected by a local network, players living afar can now share a game with a similar experience like the offline version. Existing techniques have provided smooth and fair environments for massively multiplayer online games (MMO) and many real-time strategy games (RTS), as these games can tolerate hundreds of millisecond network delay and still maintain synchronization [1]. However, delay-sensitive games like fighting (FTG) and sports games cannot yet provide satisfying player experience over the Internet. These games have two unique properties.

a) They require instant and precise judgments. For example, in a fencing game, it is necessary for a judgment immediately after a sabre hits an opponent. A late judgment will allow the loser to continue and violate the rule of the game.

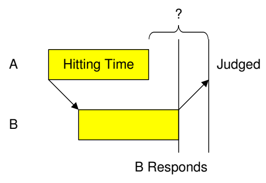
b) The speed is fast, making any lag or freeze perceptible and reduces the game's controllability and interactivity. Hence, we cannot wait a long period for the response from the other player before making a judgment.

In this paper, we focus on two-player delay-sensitive games, which are found in fighting and sports games. These games operate in a hit-and-response model in Figure 1(a).

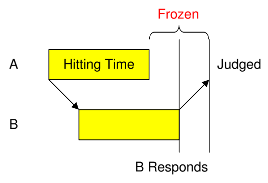
In this model, the game-playing process is divided into rounds, where a round is a period for one player (player A) to make an action and the other player (player B) to respond to the action. For instance, in a fighting game, an action can be one in which A attacks B. We define the *hitting time* to be the period from the start of A's action (appearance) to the end of A's action (reaching the end of play). In delay-sensitive games, the action is so fast that hitting times can be less than 500 ms and are of the same magnitude as network delays.

Due to network latency and jitter-buffer delays (for smoothing the late arrivals of packets), A's action will start later in B's reality. B can respond to the action (such as guarding against it) only after receiving it through the network. B's response is then sent back to A to let A know the result of the judgment (whether B has succeeded to guard against it). Figure 1(a) shows a *blank period* in A's reality equal to the round-trip latency. In this period, A has completed the action but does not know the outcome until the response from B has arrived. A cannot make another action because this may violate synchronization (such as A continuing to attack B, who has already countered the attack and has disabled A). Instead, A has to wait until the end of the blank period. To enhance player experience, it is essential to conceal this blank period.

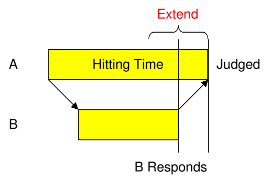
A naive method for covering the blank period is to freeze the game until A has received B's response (Figure 1(b)). This works in delay-insensitive games like MMO and RTS; however, it does not work in delay-sensitive games with hitting times of similar magnitude as network latency.



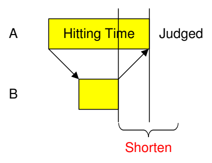
(a) Blank period



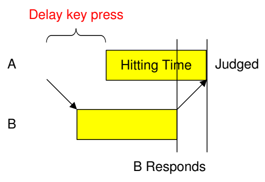
(b) Frozen action in blank period



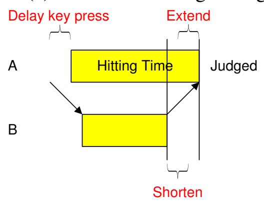
(c) LPF1: Extending hitting time



(d) LPF2: Shortened time



(e) Local lag



(f) Proposed scheme

Fig. 1. (a) Network delay introduces a blank period before A can get the result of the judgment. (b) To keep synchronization, a traditional method will freeze the movement of A until B's response has been received. (c) Local perception filter 1: A's hitting time can be extended in A's reality in order to cover the blank period. (d) Local perception filter 2: B's response time can be shortened in B's reality in order to have B's response at A earlier. (e) Local lag: A's pressing the key to start the action can be acted later in A's reality in order to hide the blank period. (f) By combining the local perception filters and the local lag, the blank period can be divided into smaller periods that can be hidden more readily. The important question is how to modify the timing of events so that they are imperceptible to both players when compared to a reference scenario with zero network delay.

To gracefully conceal the blank period, recent methods have proposed to modify the reality in games. Local Perception Filters (LPFs) [2] are methods for changing the hitting time in order to cover the blank period. Figure 1(c) shows the extension of the hitting time of the action in A's reality to cover the blank period. In other words, A will see a slower action while B still sees the action at normal speed. This method was employed to maintain synchronization in shooting games where players are generally far from each other in virtual space and missiles are needed to hit the other players. In A's reality, by making the missile speed normal near A but slower near B (who is far from A), A cannot easily perceive the difference. Alternatively, Figure 1(d) shows the LPF approach that keeps the action speed normal in A's reality but accelerates the action in B's reality.

However, the LPF approach does not work well for compensating the blank period in FTG and sports games. Firstly, since A and B are close to each other in virtual space, they can clearly see the action sequence and notice any acceleration or deceleration. Secondly, the modification of the duration of the hitting time is significant when it is very short, which is the case in games studied in this paper. The side-effects of LPFs, therefore, make the change in hitting times *perceptible*.

To demonstrate the effect of LPFs, we have built a simple two-player FTG prototype (see Figure 2) where A hits B from either the upper or the lower side, and B defends against the attack by moving a paddle up or down. We ran the game at 60 frames/sec (fps) to make it smooth and set the hitting time in our *reference setting* to 20-frame duration = $20 \times \frac{1}{60}$ sec. We then extended the hitting time in A to 26-frame duration to simulate the case where LPF was used to cover a 100-ms blank period (*LPF1 setting*). Alternatively, we shortened the response time in B to 14-frame duration (*LPF2 setting*).

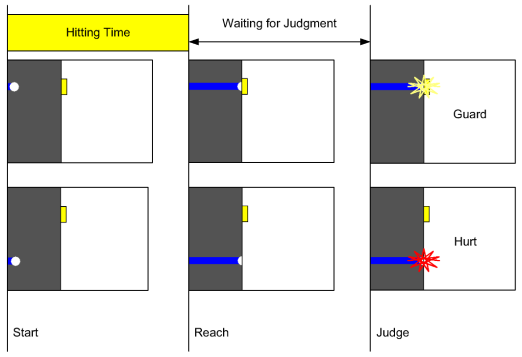


Fig. 2. Prototype of delay-sensitive fighting game. A attacks B from either the upper or the lower side, and B guards against the attack by moving the yellow paddle up or down.

We then conducted subjective tests by asking 10 subjects to play the game as A (or as B in *LPF2 setting*) in both the reference and the two LPF settings and to report the one with a longer hitting time. Each subject was asked to press specific keys on the keyboard and was allowed to play the game twice to assure consistency. Not surprisingly, 90% (*resp.* 100%) of the subjects correctly figured out that the hitting time under the LPF1 (*resp.* LPF2) setting was longer (*resp.* shorter).

The results show that the side-effects of the LPF approach are *perceptible* and will make players indirectly perceive the network latency. One may argue that the perception of the side-effects is due to the large modified hitting time. To test this argument, when we let subjects compare the 20-frame and 22-frame versions (which is not sufficient to compensate for the blank period) under LPF1, only 70% of them could correctly find the longer one.

Another method call Local Lag [3] (Figure 1(e)) attempts to delay the start of the hitting time in A's reality, so that A will share the same reality with B on the action. This is done by delaying the key event in A and by making the action start a little later after the key command is triggered by A.

We find the Local Lag method perform similarly as LPF. Using a hitting time of 20-frame duration as before, we asked subjects to compare between the *reference setting* with no key delay and the *local-lag setting* with 6-frame (*resp.* 2-frame) key delay. The result shows that 90% (*resp.* 70%) of the subjects can correctly find the one with key delay.

In short, previous approaches have perceptible side-effects when the network latency is long. In those cases, they have to sacrifice interactivity for synchronization and precise judgment. Their perceptible side-effects can be attributed to a lack of perception-driven objective that leads to improper tuning of timing of events for concealing the side effects.

To address the issue in previous approaches, we propose in this paper a perception-driven objective for optimizing the concealment of the blank period. Since our goal is to determine any perceptual effect of artifacts in a proposed scheme with non-zero latency when compared to the reference scheme with zero latency, only comparative ranking is needed. Our approach is based on *just-noticeable difference* (JND) to discern the difference between the two alternatives.

JND [4] is a concept studied extensively in psychophysics and has been used to measure the point where physical intensities between two alternatives lead to perceptual differences. For example, when comparing two lines, the difference between a reference line with 20 cm and another with 20.01 cm is not perceptible, but one with 21.5 cm is perceptible. JND in this case measures the shortest line over 20 cm that can be differentiated from the reference.

All JND methods today are based on Fechner [5] who discovered methods for measuring JND and Weber's laws that JND follows. In computer science, JND has been used in signal compression, water-marking, voice-over-IP, and measuring video quality. To our knowledge, the concept has never been used for optimizing blank periods in online games.

Based on the limitations in previous approaches, this paper studies a new problem for concealing the effects of network delays that are perceptible by users in fast-paced delay-sensitive interactive games. The scientific challenge is to arrange the action sequences in order to "hide" the delays and to make users "believe" that the game operates like one without delay. Since user perception cannot be mathematically modeled, JND is our metric to help adjust the actions sequences. In contrast to existing work on JND, this work belongs to a new research area in JND called delay concealment.

Figure 1(f) shows our key idea that combines the previous approaches (LPFs and Local Lag) to compensate for the blank period and that uses JND to drive their relative setting in order to minimize the probability for users to perceive the difference with respect to the case of zero network latency. The use of JND is critical because it guides the adjustments of delayed key press and hitting and response times in order to minimize the probability for the changes to be perceived. JND requires offline subjective tests, whose results can be used at run time to set the timing of actions.

To demonstrate the idea, we use the same setup as before but delay A's key event, extend A's hitting time, and shorten B's response time, each by a 2-frame duration. (A more elegant scheme is shown in Section 4.) This time, at most 70% of the subjects can identify the delayed case.

Problem statement. In this paper, we study the statistical modeling of users' perception of side-effects when compensating for the blank period caused by waiting for judgment and synchronization in delay-sensitive interactive games. By conducting offline subjective tests, the results will help set the timing of various actions in order to conceal the blank period and to make it not perceptible to both players.

Our problem is studied with respect to fast-paced and delay-sensitive games with hitting times less than 1 sec. We assume tight synchronization in which a judgment has to be made before the next action can be carried out.

The rest of the paper is organized as follows. Section 2 summarizes methods for concealing losses in network transports. In Section 3 we present our proposed JND concept and methods for measuring it efficiently. We then demonstrate in Section 4 the use of JND for concealing delay effects, before concluding the paper.

2. INTERNET LOSS AND DELAY BEHAVIOR

In this section, we show the range of delays (due to jitter buffers and loss recovery) that our method aims to conceal.

Online games played over the Internet use its best-effort packet transport for sending messages. When messages are lost in transmission or arrive after the judgment deadline, a loss of synchronization may occur. Since its consequence depends on the game logic and speed, only the number of losses of synchronization per second is important. However, as synchronization commands are transported in packets, their loss rate is similar to the rate that packets are lost or arrive late in the receiver (the *unconcealed packet rate* or UCPR).

Two types of network impairments will affect UCPR. Firstly, delay jitters, which are sudden and short-term increases of network delay, will postpone the arrival of packets. A playout buffer can be used to smooth jitters, although its length is limited by the judgment deadline. A packet arriving later than the judgment deadline will be considered lost. Secondly, packet losses will render packets unavailable. In delay-sensitive games, retransmissions that require additional round-trip times, such as those in TCP, cannot be tolerated.

A general approach for concealing losses of packets with

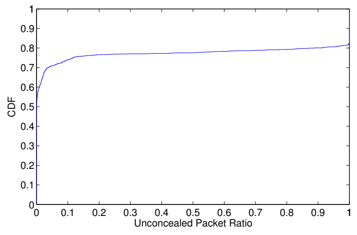


Fig. 3. Cumulative distribution of UCPR showing that 75% of the 1,200 traces with $t^{\text{overall}} \leq 200$ ms can provide UCPR ≤ 0.1. The remaining traces have long delays whose effects are perceptible in delay-sensitive games.

small payload (those carrying action commands) and without retransmission is piggybacking. This transmits copies of a previous message along with a new message in a UDP packet. If a packet is lost, the data it carries can be recovered as long as the piggybacked version arrives before the deadline. With buffering and piggybacking, the round-trip delay is

$$t^{\text{overall}} = \left(t^{\text{propagation}} + \frac{PiggyDegree}{PacketRate} + t^{\text{buffering}} \right) \times 2$$

where $t^{\text{propagation}}$ is the network delay, $PiggyDegree$ is the number of subsequent packets that carry a previously transmitted message, $PacketRate$ is the packet transmission rate, and $t^{\text{buffering}}$ is the time for buffering packets. The up- and down-links are assumed to be symmetric for simplicity.

We find that with $t^{\text{overall}} \leq 200$ ms, satisfactory UCPR (with packet buffering and piggybacking) can be achieved for most connections. This result was obtained by traces collected in the PlanetLab. Using a UDP probe, we sent 500-byte UDP packets every 20 ms and collected more than 1,200 1-minute packet traces between 46 node pairs (including short-haul and long-haul connections) chosen randomly from a set of 180 nodes. We then calculated UCPR for the 1,200 traces, while assuming symmetric up and down links. Figure 3 depicts that more than 75% of these traces have UCPR < 0.1 when $t^{\text{overall}} \leq 200$ ms. The remaining traces generally have network latency $>$ 200 ms. In short, most of the connections can support delay-sensitive games if $t^{\text{overall}} \leq 200$ ms.

3. STATISTICAL CHARACTERIZATION OF JND

In this section, we characterize JND statistically and present methods for conducting JND subjective tests.

Definition of JND. In our subjective tests, subjects are presented two interactive game sessions in a random order. One of the sessions runs under no delay with hitting time $t^{\text{hit}}_{\text{ref}}$ and the other operates with delayed $t^{\text{hit}} = t^{\text{hit}}_{\text{ref}} + t^{\text{modified}}$ if an extended hitting time is used (and $t^{\text{hit}} = t^{\text{hit}}_{\text{ref}} - t^{\text{modified}}$ if a shortened response time is used). Subjects are then asked which session has a relatively longer delay. Let p be the fraction of subjects who correctly answer the question. Because the original and the modified settings are presented in a random order multiple times, those cases with $p < 0.5$ are folded into $p > 0.5$. We define the awareness of delay as follows.

Definition 1. The $c\%$ JND (or *awareness*) of the session with hitting time t^{hit} is the maximum extent of t^{modified} that makes $p \leq c\%$.

We find the *awareness* metric important for measuring the effect caused by the modified hitting time. Unlike the absolute change in hitting time which is a quantitative metric, awareness is a subjective metric that relates to human perception in a statistical sense. In the following, we describe some of the fundamental properties of JND.

Axiom 1. p is monotonically non-decreasing with t^{modified} under given $t^{\text{hit}}_{\text{ref}}$. (This means that when a larger modification is made on t^{hit}, more subjects will perceive the change.)

Axiom 2. p is monotonically non-increasing with $t^{\text{hit}}_{\text{ref}}$ under given t^{modified}. (This means that the modification is less significant when it is made to a slower motion.)

Axiom 3. p is a continuous function of t^{modified} and $t^{\text{hit}}_{\text{ref}}$. (When there is a large pool of subjects, this signifies that small changes in t^{modified} or $t^{\text{hit}}_{\text{ref}}$ will only be perceived by a small fraction of the subjects.)

Efficient measurements of JND. To measure JND under various t^{hit} and to make the results statistically valid, an exceedingly large number of subjective tests will be needed. To reduce the number of tests, we make the following assumptions.

Assumption 1: IID. The subjects have the same level of expertise, and their ability to perceive differences in delay is independent and identically distributed (IID). This assumption allows the statistics of responses to be obtained by repeated tests using multiple subjects. Further, the estimated $\tilde{p}$ with limited tests approaches the actual p asymptotically.

Assumption 2: Monotonicity. p is monotonically increasing with t^{modified} under a given $t^{\text{hit}}_{\text{ref}}$ for $p \in (0.5, 1]$. Note that p will not be less than 0.5 in relative comparisons since the response is a random guess when a subject cannot perceive the difference. This is more restricted than *Axiom* 1 and eliminates those cases in which p stays unchanged when t^{modified} is increased. It allows us to simplify *Definition* 1 as follows.

Definition 1.* The $c\%$ JND (*awareness*) of the session with hitting time t^{hit} is the extent of t^{modified} that makes $p = c\%$.

Based on these simplifications, the measurement of JND can be done as follows:

1. With a given $t^{\text{hit}}_{\text{ref}}$, we measure $\tilde{p}$ under several t^{modified}.
2. We repeat *Step* 1 with different $t^{\text{hit}}_{\text{ref}}$.
3. According to *Axiom* 3, we interpolate $\tilde{p}$ at those $t^{\text{hit}}_{\text{ref}}$ and t^{modified} that are not tested.

The above procedure does not consider errors in $\tilde{p}$ that can occur with limited subjective tests. This happens because the number of subjects who correctly respond to the question follows a binomial distribution $n \sim B(N,p)$, and $\tilde{p} = n/N$ converges to p as $N \to \infty$. Moreover, the 60-fps rate in a real-time game introduces uncertainties in $\tilde{p}$, since any motion that happens in plus or minus one frame time ($\pm\frac{1}{60}$ sec) cannot be detected. Basing on this understanding, we revise the process into an iterative procedure. Instead of attempting to get a precise result (which is actually unnecessary, considering the fuzziness in human perception), we want $\tilde{p}$ at different $(t^{\text{hit}}_{\text{ref}}, t^{\text{modified}})$ to satisfy the monotonicity properties in *Axioms* 1 and 2. The new procedure is as follows.

1. Find $\tilde{p}_{i,j}$ at different $(t^{\text{hit}}_{\text{ref},i}, t^{\text{modified}}_{i,j})$ by subjective tests.
2. Stop and output the $\tilde{p}_{i,j}$'s if their monotonicity is satisfied within the awareness range $\tilde{e}_{i,j}$ defined by the difference of awareness within $\pm$1-frame time:

$$\tilde{e}_{i,j} = \pm \frac{1}{n^{\text{modified}}_i - 1} \sum_{k \neq j} \left| \frac{\tilde{p}_{i,j} - \tilde{p}_{i,k}}{t^{\text{modified}}_{i,j} - t^{\text{modified}}_{i,k}} \right| .$$

   Because we have not conducted subjective tests at ± 1 frame intervals for $(t^{\text{hit}}_{\text{ref},i}, t^{\text{modified}}_{i,j})$, we estimate their awareness by a linear interpolation of the awareness found by subjective tests conducted on the remaining $t^{\text{modified}}_{i,k}|_{k\neq j}$ modified hitting times under $t^{\text{hit}}_{\text{ref},i}$ ($n^{\text{modified}}_i - 1$ of them). In the above summation, each term represents the awareness at 1-frame interval based on interpolating the difference in awareness ($= \tilde{p}_{i,j} - \tilde{p}_{i,k}$) with respect to $t^{\text{modified}}_{i,j} - t^{\text{modified}}_{i,k}$. The equation then averages all the estimated $\pm$1-frame awareness.
3. Otherwise, for each of the $\tilde{p}_{i,j}$'s that violates monotonicity, perform more subjective tests in order to get a more precise $\tilde{p}_{i,j}$. Go to *Step 2*.

Note that $\tilde{e}_{i,j}$ in Step 2 is improved iteratively as better $\tilde{p}_{i,j}$ is found by more tests. This ensures the eventual termination of the procedure in which monotonicity is achieved. Further, we only perform measurements in *Step* 3 for those $\tilde{p}_{i,j}$'s that violate monotonicity, thus saving the number of tests.

4. EXPERIMENTAL RESULTS

In this section, we compare the awareness found for the previous delay-concealment schemes and our proposed scheme.

We conducted our subjective tests with 20 subjects using our prototype in Figure 2. Each subject had 2-3 tests (depending on whether the error is small enough) for each combination of hitting and modification times. We measured their awareness for hitting times ranging from 10 ms to 40 ms and for changes in hitting/response times between 2-frame and 6-frame periods. For fairness, subjects were not told which method the prototype was using to conceal delays.

JND in previous delay-concealment schemes.

a) *Local Perception Filter 1 with extended hitting time for A.* We asked subjects to compare the case with an extended hitting time and that with the original hitting time (shown in a random order). Figure 4(a) shows that a 6-frame extended hitting time is easily perceptible by subjects regardless of the original hitting time. For example, if we try to cover a 100-ms (6-frame time) round-trip network delay, the modified hitting time is perceptible by more than 85% of the subjects when the original hitting time is less than 35-frame times. (Note that 85% is not directly attained but is calculated with the uncertainty and monotonicity consideration mentioned above.) We did not show modified hitting times of 12-frame times (in order to cover a 200-ms round-trip delay), as the awareness level is 100%. The results show that this scheme does not work well in real-time delay-sensitive interactive games.

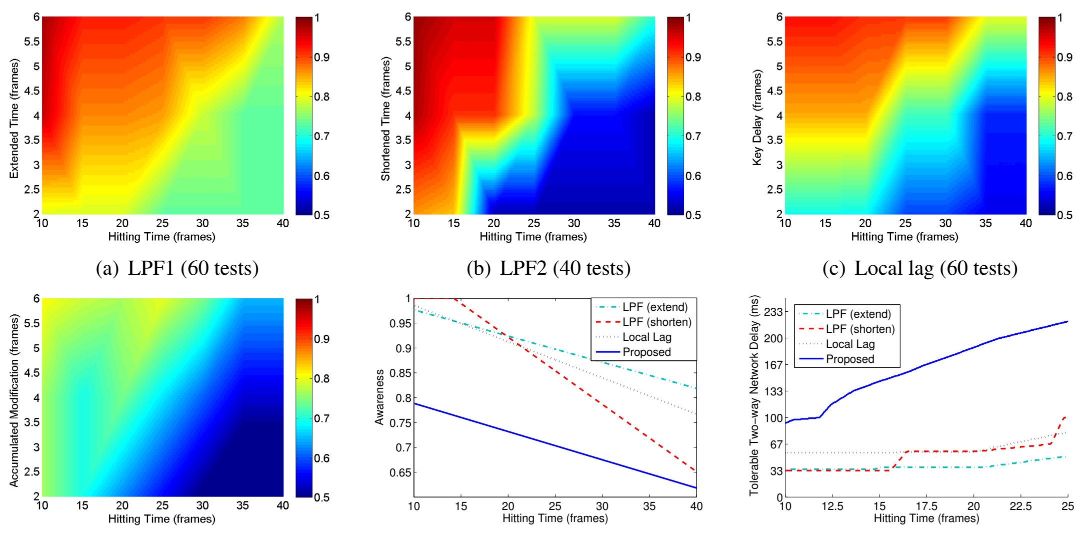
b) *Local Perception Filter 2 with shortened response time for B.* We asked subjects to compare the case with shortened response time in B and that with the original response time (again shown in a random order). Figure 4(b) shows that, when the original hitting time is short, subjects are more sensitive to the modification, and the delay effects can be easily perceived. Hence, this scheme is not suitable for concealing network delays in games with fast motions. However, when the original hitting time is longer, the modification is less perceptible, and the scheme outperforms LPF1.

c) *Local Lag.* We asked subjects to compare the case with key command lag at A and that without (in a random order). Figure 4(c) shows that the delay effects are less perceptible than LPF1 and LPF2 when the hitting time is short. However, there are still more than 80% of the subjects who can perceive the delay when the hitting time is less than 30-frame times and the network delay to be covered is 100 ms. Hence, the scheme is inadequate for concealing delays in fast-moving games.

Proposed Scheme. We propose to combine the three previous methods to disperse the network delay to be covered in order to make its effect less perceptible to both players. Instead of the naive method that divides the delay into three equal parts and conceals them by the three previous methods, we use JND to guide their apportionment. There are two considerations here. Firstly, the modification to the two players should lead to the same awareness in delay for fairness. Otherwise, the player with higher awareness will perform poorer than the other. Secondly, the overall awareness should be minimized.

In Player A (who initiated the action), we employ both the LPF with extended hitting time and the Local Lag schemes. Because the two schemes are dependent, the awareness due to their combined effects cannot be separated easily. Instead, we perform subjective tests to measure the overall awareness by assigning equal delays to both schemes.

Next, we employ LPF with shortened response time at B. This is independent of the schemes employed at A because B does not know the game scene there. For fairness, we let the awareness of both A and B to be the same. The overall awareness is the same as the individual awareness at A and B due to the independence of the schemes.



(a) LPF1 (60 tests) (b) LPF2 (40 tests) (c) Local lag (60 tests)

(d) Proposed (60 tests) (e) Comparison of awareness (f) Tolerable network delays

Fig. 4. (a) Awareness due to delay effects of LPF1 (with extended hitting time for A) is easily perceptible by subjects, where 1-frame time = $\frac{1}{60}$ sec. (b) Awareness due to delay effects of LPF2 (with shortened response time for B) that are more (*resp.* less) significant when the hitting time is short (*resp.* long). (c) Awareness due to the local lag scheme has a less perceptible delay effect when the hitting time is short, but still cannot conceal the effects when network delay is long. (d) Our proposed scheme has the smallest awareness among the four schemes. (e) Comparison of awareness when there is a 100-ms blank period to cover. (f) Tolerable two-way network delays under different hitting times when the required awareness is 80%.

The offline results found are then stored in an awareness table that maps tuples of hitting time and network latency to tuples of timing changes of actions. At run time, based on the hitting time and network delay to be covered (known to both sides), both A and B look up the corresponding modified action timing that is sufficient to conceal the blank period.

Figure 4(d) shows that the overall awareness of our proposed scheme is significantly better than that of the previous methods. Figure 4(e) further compares the awareness of the various schemes when used to cover a 100-ms blank period. It shows that our method can reduce the awareness by more than 10% in most cases. Because our method balances the awareness in both players, it is fair to both players.

Our proposed approach can also help measure the network condition that supports a required level of awareness. Figure 4(f) shows the maximum tolerable round-trip network delays under different hitting times if the awareness to differentiate between the modified version and the original version is 80%. The results can help developers fine tune the hitting times under different network conditions in order to achieve good quality of experience (QoE) to players. For instance, they can increase the hitting times so that QoE can be more robust in connections with long delays. Figure 4(f) shows significant improvements of our proposed scheme when compared to the previous schemes. For instance, when the hitting time is 20 ms, our method can conceal the delay effects when the round-trip delay is near 200 ms, whereas previous schemes can do the same only when the round-trip delay is less than 67 ms.

In this paper, we have studied delay-concealment schemes for delay-sensitive interactive games over the Internet that require precise synchronization. Our major contribution is on the use of JND (in terms of an awareness metric) to quantify user perception on changes in hitting and response times in order to conceal network delays.

5. REFERENCES

[1] M. Dick, O. Wellnitz, and L. Wolf, “Analysis of factors affecting players’ performance and perception in multiplayer games,” in *Proc. of 4th ACM SIGCOMM workshop on Network and system support for games*. ACM, 2005, pp. 1–7.

[2] J. Smed, H. Niinisalo, and H. Hakonen, “Realizing the bullet time effect in multiplayer games with local perception filters,” *Computer Networks*, vol. 49, no. 1, pp. 27–37, 2005.

[3] D. Stuckel and C. Gutwin, “The effects of local lag on tightly-coupled interaction in distributed groupware,” in *Proc. ACM Conf. on Computer Supported Cooperative Work*, 2008, pp. 447–456.

[4] S. Hecht, “The visual discrimination of intensity and the Weber-Fechner law,” *The Journal of General Physiology*, vol. 7, no. 2, pp. 235–267, 1924.

[5] G.T. Fechner, “Element of psychophysics,” *New York: Holt, Rinehart and Winston (Original work published 1860)*, 1966.