# Influences of Network Latency and Packet Loss on Consistency in Networked Racing Games

Takahiro Yasui*
Department of Computer Science and Engineering
Graduate School of Engineering
Nagoya Institute of Technology
Nagoya 466-8555, Japan

Yutaka Ishibashi
Department of Computer Science and Engineering
Graduate School of Engineering
Nagoya Institute of Technology
Nagoya 466-8555, Japan
ishibasi@nitech.ac.jp

Tomohito Ikedo
Department of Computer Science and Engineering
Graduate School of Engineering
Nagoya Institute of Technology
Nagoya 466-8555, Japan
kotoris@mcl.elcom.nitech.ac.jp

## ABSTRACT

This paper investigates the influences of network latency and packet loss on the consistency among players in networked racing games. We here deal with the $\Delta$-causality control as causality control and dead-reckoning as prediction control. By experiment, we make a performance comparison among the $\Delta$-causality scheme, the dead-reckoning scheme, a scheme which carries out the $\Delta$-causality control and dead-reckoning together, and a scheme which performs neither of the two types of control. As a result, we show that the combination use of the two types of control can keep the consistency and fairness among players in good condition.

## Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Communications Applications; K.8.0 [**Personal Computing**]: General—*Games*

## General Terms

Algorithms, Performance, Human Factors, Experimentation

## Keywords

Networked racing game, Consistency, Fairness, Causality control, Dead-reckoning

## 1. INTRODUCTION

Broadband access to the Internet is enabling networked real-time games such as networked racing games and networked shooting games. Such games have severe constraints on the latency from input of information to its output [1]. Also, the consistency of the state among players and the causality of input events are important. These features largely influence the outcome of a race (i.e., victory or defeat) in the games.

However, owing to the network latency and packet loss, the causality and consistency among players may be disturbed. To solve the problem, we need to carry out consistency control [2], causality control [3]–[6], prediction control [6], [7], and so on. As the difference in network latency or in packet loss among players becomes larger, the positions of racing cars displayed at one player become more largely different from those at another player. This brings the inconsistency among the players.

In [3], the authors demonstrate that the causality can be maintained by the $\Delta$-causality control. However, since a number of packets are discarded when the network load is heavy, the consistency among players may be disturbed. Pantel and Wolf illustrate the effectiveness of dead-reckoning [8], which is one of prediction control schemes, by applying dead-reckoning to several kinds of games [7]. In [6], Diot and Gautier propose the bucket synchronization, which carries out causality control and prediction control together. However, they do not make a performance comparison between the bucket synchronization and other schemes. Also, they do not clarify how the joint use of the two types of control is superior to the individual use of each type of control. Furthermore, to the best of the authors' knowledge, there is no paper which makes a performance comparison in terms of the influences of the network latency and packet loss on the consistency. Such a comparison is very important to clarify which control scheme outperforms others.

This paper deals with the $\Delta$-causality control as causality control and dead-reckoning as prediction control. By experiment, we make a performance comparison among the $\Delta$-causality scheme, the dead-reckoning scheme, a scheme which carries out the $\Delta$-causality control and dead-reckoning together, and a scheme which performs neither of the two types of control. We also investigate the influences of network latency and packet loss on the consistency in the four schemes.

The rest of this paper is organized as follows. Section 2 discusses the consistency in networked racing games and de-

*Currently, he is with NTT West Corp., Osaka 540–8511, Japan.


*NetGames'05,* October 10–11, 2005, Hawthorne, New York, USA.

scribes its performance measures. Section 3 explains the four schemes. The method of the experiment is explained in Section 4, and experimental results are presented in Section 5. Section 6 concludes the paper.

## 2. CONSISTENCY AMONG PLAYERS

We suppose in this paper that two players (players 1 and 2) play a networked racing game. We also assume that the game is implemented based on the peer-to-peer model.

In what follows, we first handle the case in which network delays between the two players are small. Next, we deal with the case in which the network delays are large or largely different from each other. In the two cases, we assume that there is no packet loss for simplicity of explanation. Then, we discuss the case in which packet loss occurs. Finally, we introduce performance measures.

### 2.1 Influence of network latency

In Fig. 1, we show displayed images of the racing cars of players 1 and 2 in the first case. The image on the left-hand side in the figure is displayed at player 1, and that on the right-hand side is at player 2. For simplicity, we assume in the figure that the racing course is straight. Players 1 and 2 drive cars 1 and 2, respectively. At each player, the player's car and the other player's car are displayed. The information about the position of player $i$'s car ($i = 1$ or 2) arrives late at player $j$ (if $i = 1$, $j = 2$; otherwise, $j = 1$) owing to network delays. Therefore, if we update the position of car $i$ at player $j$ without any control (that is, on receiving the information), the position of car $i$ at player $i$ is different from that of car $i$ at player $j$. This means that the state at player $i$ is not consistent with that at player $j$.

In order to discuss the consistency quantitatively, we introduce the positional error of a car, which is defined as the difference between the position of car 1 (2) at player 1 and that at player 2. Let us denote the positions (in the direction of movement) of cars 1 and 2 at player 1 by $y_{1,1}$ and $y_{2,1}$, respectively, at a given time and those at player 2 by $y_{1,2}$ and $y_{2,2}$, respectively. Note that we here take account of only the direction of movement for simplicity so as to calculate the positional error. Then, the positional error $d_1$ of car 1 is given by $d_1 = y_{1,1} - y_{1,2}$, and that of car 2 is $d_2 = y_{2,2} - y_{2,1}$ (see Fig. 1)[1]. For simplicity, we here assume that $d_1 \geq 0$ and $d_2 \geq 0$. If $d_1 = d_2 = 0$, the consistency is maintained. Otherwise, the consistency is not strictly kept. However, even when $d_1 \neq 0$ or $d_2 \neq 0$, the outcome of the race (i.e., victory or defeat) at player 1 is the same as that at player 2 if which car is ahead of the other car is the same between the two players[2]. In Fig. 1, since $d_1 \neq 0$ and $d_2 \neq 0$, the consistency is disturbed; however, which car is ahead of the other car is the same between the two players; this is because the values of $d_1$ and $d_2$ are small.

Then, we handle the case in which the network delays are large or largely different from each other and there is no packet loss as shown in Fig. 2. Figure 2 (a) illustrates the case in which the positional relations of the two cars between players 1 and 2 are not the same. In Fig. 2 (b), the positional relations of the two cars between players 1 and 2 are the same; that is, car 2 is ahead of car 1 at the two players. We set the values of $d_1$ and $d_2$ in Fig. 2 (a) to the same as those in Fig. 2 (b), respectively ($d_1 > d_2$ in Fig. 2). In Fig. 2 (b), since car 2 is largely ahead of car 1 at player 2, car 2 is slightly ahead of car 1 at player 1. If car 2 is ahead of car 1 at player 2 by more than $d_1 + d_2$, the positional relations can be the same (see Fig. 2). Also, at player 1, the positional relations can be the same if car 1 is ahead of car 2 by more than $d_1 + d_2$. That is, even if the consistency of the state is violated, the consistency of the positional relations can be preserved; this depends on the positional errors of the two cars (i.e., $d_1$ and $d_2$). As the values of $d_1$ and $d_2$ increase, it becomes more difficult to keep the same positional relations.

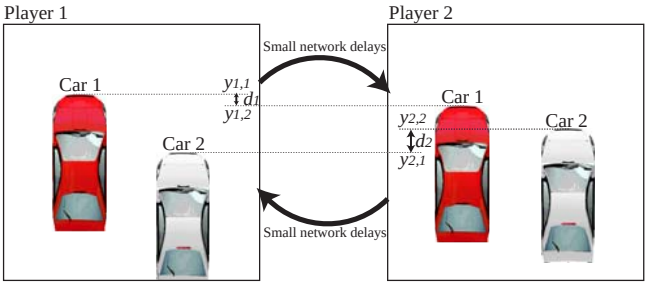



**Figure 1: Displayed images at players 1 and 2 in the case where network delays are small and there is no packet loss.**

### 2.2 Influence of packet loss

In the case where packet loss occurs, the position of a car is not updated if we do not carry out any control; that is, the car stays at the previous position. Therefore, packet loss has the influence similar to that of network latency. As shown in Fig. 3, where we assume that the network delays between the two players are the same and packets of player 1 are lost, the information about the position of player 1 (2)'s car arrives late at player 2 (1) owing to network delays. Furthermore, the packet loss delays the position update of car 1 at player 2. In Fig. 3, since $d_1 \neq 0$ and $d_2 \neq 0$, the consistency is disturbed. However, since car 1 is ahead of car 2 by more than $d_1 + d_2$ at player 1, the positional relations of the two cars between players 1 and 2 are the same.

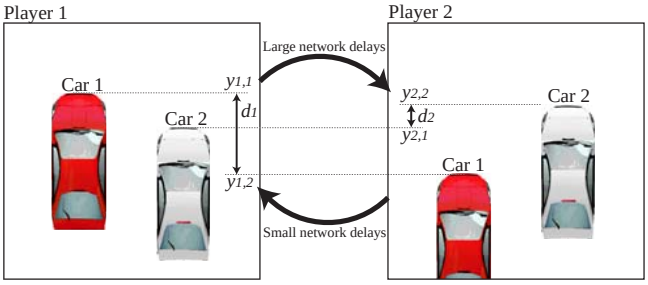
### 2.3 Performance measures

As a performance measure for the consistency, this paper introduces the *inconsistency rate of the positional relations of the two cars* (that is, which car is ahead of the other car) between the two players. We obtain the positional information about the two cars at regular intervals and compare the positional relation between the two cars at player 1 with that at player 2. The inconsistency rate is defined as the ratio of the number of disagreements between the two positional relations to the total number of comparisons. This measure is important since the positional relations are closely related to the outcome of a race (i.e., victory or defeat).

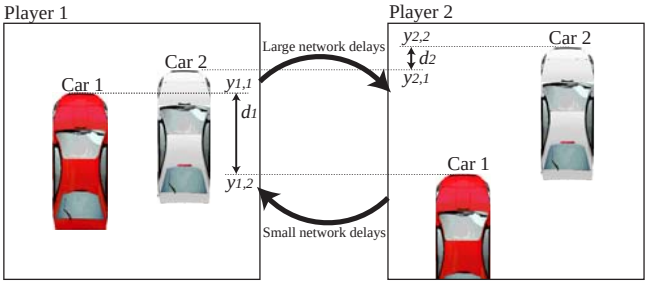
On the other hand, large differences between $d_1$ and $d_2$ lead to the unfairness between the two players. This is be-

[1]When the course is curved, we obtain the positional error by approximating the curve with multiple straight lines which have a constant length.

[2]It should be noted that generally prediction control schemes produce positional errors.




(a) Case in which the positional relations of the two cars between players 1 and 2 are not consistent




(b) Case in which the positional relations of the two cars between players 1 and 2 are consistent

**Figure 2: Displayed images at players 1 and 2 in the case where network delays are largely different from each other and there is no packet loss.**

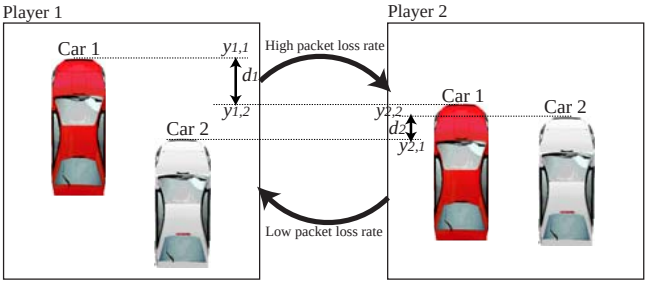



**Figure 3: Displayed images at players 1 and 2 in the case where packets of player 1 are lost.**

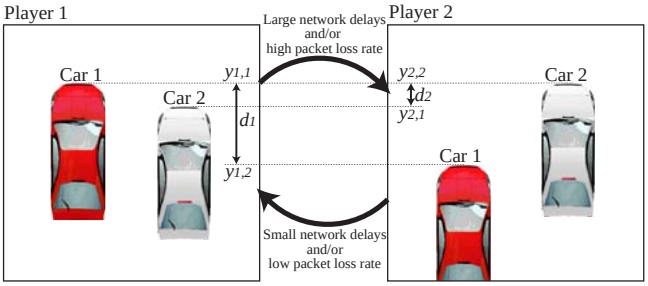



**Figure 4: Displayed images at players 1 and 2 in the case where player 1 has the same skill as player 2 (i.e., $y_{1,1} = y_{2,2}$).**

cause as the positional error becomes larger, the car of a player is displayed more largely late at the other player. To discuss the fairness in further detail, we assume that the position of car 1 at player 1 is equal to that of car 2 at player 2; that is, $y_{1,1} = y_{2,2}$ (see Fig. 4). This means that the skills of the two players are equal to each other. As described earlier, when the difference in the position between the two cars at player 1 or 2 is larger than $d_1 + d_2$, the positional relations between the two players can be the same. Thus, if player 1 can advance the position of car 1 from the current position by $d_1$, the positional relations can be the same between the two players. As in player 1, if player 2 can advance the position of car 2 by $d_2$, the positional relations can also be the same. In Fig. 4, since $d_1 > d_2$, player 1 needs to advance the position of car 1 more largely than player 2 in order to have the same positional relations between the two players. Therefore, we here choose to say that player 1 is disadvantageous in terms of the fairness in this case.

To discuss the fairness between the two players quantitatively, we employ the difference $d_1 - d_2$ in the positional error between the two cars. As in Fig. 4, when $d_1 - d_2 > 0$, player 1 is unfavorable in terms of the fairness. When $d_1 - d_2 = 0$, players 1 and 2 are even. When $d_1 - d_2 < 0$, player 2 is disadvantageous. Thus, as another performance measure, we employ the *average difference in the positional error between the two cars*, that is, the average of $d_1 - d_2$.

## 3. SCHEMES FOR PERFORMANCE COMPARISON

In this paper, we handle four schemes for a performance comparison; the $\Delta$-*causality scheme* (referred to as *Causality* in this paper), the *dead-reckoning scheme* (*DR*), and a scheme which carries out the $\Delta$-causality control and dead-reckoning together (*DR+Causality*), and a scheme which performs neither of the two types of control (the *no-control scheme*; *NC*)[3]. In the four schemes, each player obtains the positional information about the player's car at regular intervals (every 33 ms in our experimental system) and sends the information with its timestamp, which is the generation time of the information, as a computer data *media unit* (*MU*) [3], [9], which is an information unit under the two types of control, to the other player.

In DR, at each terminal (or player)[4], the current position of the other player's car is predicted by the past received (transmitted) MUs. For simplicity, we use the first-order

[3]We also dealt with a scheme which exerts the adaptive $\Delta$-causality control [9], which dynamically changes the value of $\Delta$ ($\Delta \geq 0$) in the $\Delta$-causality control according to the network load, and dead-reckoning together. In the scheme, group (or inter-destination) synchronization control [9], which adjusts the output timing among multiple destinations, is carried out to employ the same value of $\Delta$ at the two players. As a result of the experiment, we found that the scheme is slightly inferior to DR+Causality in terms of consistency. However, the scheme outperforms DR+Causality from the point of view of interactivity; the average delay of the scheme is smaller than or equal to $\Delta$ seconds, and that of DR+Causality is equal to $\Delta$ seconds. For simplicity of discussion, we do not deal with the scheme in this paper.

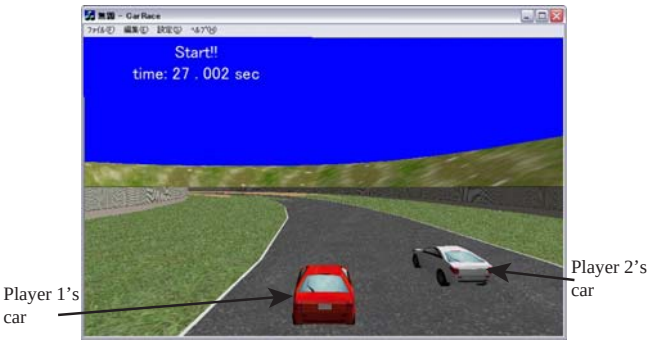
[4]We here use the words 'terminal' and 'player' interchangeably.

prediction [7]. In the first-order prediction, we calculate the predicted position by using the information about the position included in the last received (transmitted) MU and about the velocity calculated with the positional information included in the latest two received (transmitted) MUs. Then, we compare the predicted position with the actual position. If the difference between the predicted position and the actual position (i.e., the prediction error) is larger than a threshold value $T_{\rm dr}$, the information about the actual position is transmitted as an MU. Otherwise, the information is not transmitted. In the convergence technique, when an MU is received, we correct the position over several times in order to correct the position gradually until the difference becomes less than $T_{\rm dr}$. In this paper, for simplicity, we make the convergence at a time.

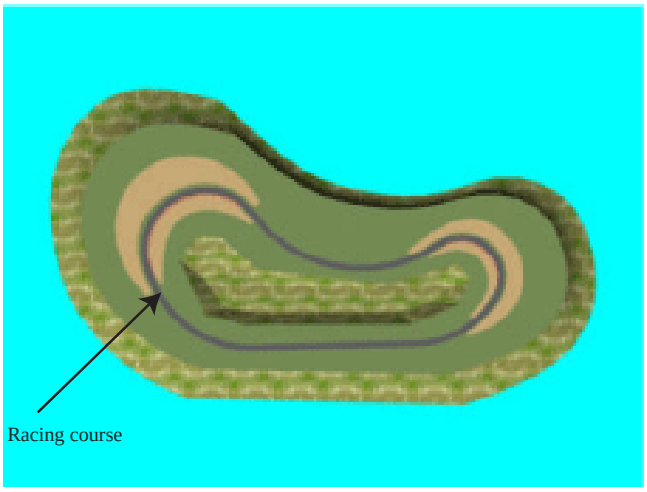
NC outputs MUs on receiving the MUs at each terminal. When we employ NC or DR, an MU which is captured at each terminal is output immediately after capturing the MU at the terminal.

In Causality, when each terminal receives an MU, the terminal saves the MU in the terminal's buffer until a time limit and then outputs it. The time limit is equal to the generation time of the MU plus $\Delta$ seconds. If the MU is received after the time limit, it is discarded. In this paper, we assume that globally synchronized clocks are employed; that is, the clock ticks at the two terminals have the same advancement, and the current local times are also the same[5].

When we use DR+Causality, if an MU is received within the time limit (i.e., its generation time plus $\Delta$ seconds) of the MU at each terminal, the MU is saved in the terminal's buffer until the time limit and then used for the prediction and convergence by the same method as that of DR. If the MU is received after the time limit, it can be used only for the prediction at the next output time. In Causality and DR+Causality, each MU which is generated at each terminal is also stored in the terminal's buffer until its time limit in order to be output.




(a) An example of displayed images at player 1




(b) An image of the racing course

**Figure 5: Displayed images of the networked racing game.**

## 4. METHOD OF THE EXPERIMENT

Figure 5 shows an example of displayed images and an image of the racing course in our networked racing game. In order to generate the computer data traffic of the same amount in the experiment, we stored the positions of two cars in files every 33 ms; we made ten files. Two players who have almost the same skill[6] drove the two cars along the racing course (see Fig. 5 (b)) for 30 seconds ten times in the case of no network latency and no packet loss. In the experiment, car $i$ ($i = 1$ or 2) is moved according to the stored files at terminal $i$. Terminal $i$ transmits MUs each of which includes the position of car $i$ to terminal $j$ (if $i = 1$, $j = 2$; otherwise, $j = 1$). When the terminal receives an MU, it updates the position of the car by using the positional information in the MU.

In the experiment, the values of $\Delta$ in Causality and DR+Causality are set to 100 ms [1][7]. The values of $T_{\rm dr}$ in DR and DR+Causality are set to 0.1, where the depth of each car is 1.

We show the configuration of the experimental system in Fig. 6. The experimental system consists of the two terminals (CPU: Pentium4 2.4 GHz, OS: WindowsXP Home Edition, RAM: 512 Mbytes, Graphic board: GeForce4 MX 420) and a network emulator (NIST Net [11]). The two terminals are connected to NIST Net via Ethernet cables (100BASE-T). NIST Net generates an additional delay for each MU according to the Pareto-normal distribution [11]. It also makes packet loss occur randomly.

In the experiment, we have examined the influences of network latency and packet loss on the consistency. First, in order to examine the influence of network latency, the average and the standard deviation of the additional delay from terminal 2 to terminal 1 are set to 75 ms and 10 ms, respectively. Those from terminal 1 to terminal 2 are changed. Next, as for the influence of packet loss, the loss rate for

[5]Using the Network Time Protocol (NTP) [10], we can adjust the clock ticks to each other within a few milliseconds.

[6]The reason why we employed the two players with almost the same skill is that we make the positional relation of the two cars (that is, which car is ahead of the other car) switch frequently. In the experiment, the positional relation of the two cars switches eight through thirteen times (average: 10.5 times). If the skills of the two players are largely different from each other, the more skillful player's car is largely ahead of the other player's one. In this case, the network delay does not influence the positional relation of the two cars as in Fig. 2 (b).

[7]In [1], it is shown that delays within around 100 ms are subjectively allowable in a networked racing game.

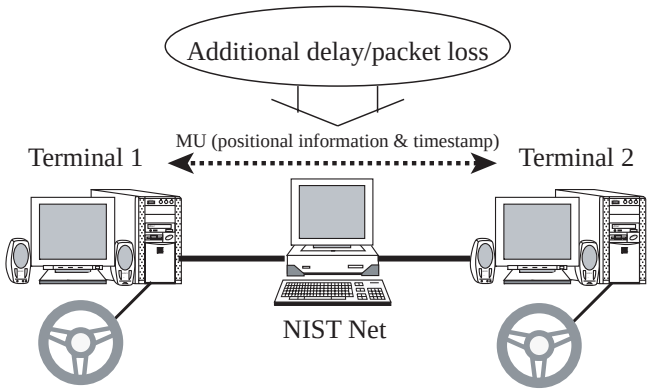



**Figure 6: Configuration of the experimental system.**

packets transmitted from terminal 2 to terminal 1 is set to 10 %, and that from terminal 1 to terminal 2 is changed. The average and standard deviation of the additional delay in both directions between the two terminals are set to 75 ms and 10 ms, respectively.

# 5. EXPERIMENTAL RESULTS

We first examine the influence of network latency on the consistency, and then we investigate that of packet loss.

## 5.1 Influence of network latency

We show the inconsistency rate of the positional relations of the two cars as a function of the average additional delay from terminal 1 to terminal 2 in Fig. 7. We also plot the average difference in the positional error between the two cars versus the average additional delay in Fig. 8. In the figures, we show experimental results when the standard deviation of the additional delay from terminal 1 to terminal 2 is 10 ms or 30 ms. We do not handle the cases in which the standard deviation of the additional delay is greater than the average additional delay in Figs. 7 and 8; therefore, experimental results in the cases are missing in the figures. Also, we plot the 95 % confidence intervals of the performance measures in the figures. However, when the interval is smaller than the size of the corresponding symbol representing the experimental result, we do not show it in the figures.

In Fig. 7, we can confirm that the inconsistency rate of NC is the largest in the whole range of the average additional delay considered here. We also see that the inconsistency rate of Causality is the smallest when the average additional delay is less than or equal to around 75 ms and the standard deviation of the additional delay is 10 ms. The reason is that Causality can output each MU at almost the same time at both terminals by buffering. However, the inconsistency rate of Causality starts to increase largely when the average additional delay exceeds about 75 ms. This is because the number of MUs which are discarded owing to missing their time limits increases largely. In the figure, we notice that the inconsistency rates of DR and DR+Causality are not heavily dependent on the average additional delay. The reason is that even when an MU arrives late, we can output the predicted MU by the prediction. Furthermore, in the two schemes, the predicted position of each car was not largely different from the actual position.

In Fig. 7, the inconsistency rate of DR+Causality is smaller than that of DR. In addition, from Fig. 7, we find that DR+Causality has almost the second smallest inconsistency rate among all the schemes when the average additional delay is less than or equal to around 50 ms or 75 ms; beyond this range, DR+Causality has the smallest. This is because DR+Causality can compensate for larger additional delay jitters by buffering MUs. Hence, DR+Causality can predict the position of each car more precisely than DR.

Then, we note in Fig. 7 that the inconsistency rates of NC, DR, and DR+Causality are hardly dependent on the standard deviation of the additional delay. However, when the average additional delay is 100 ms or 125 ms, the inconsistency rate of Causality at the standard deviation of 30 ms is smaller than that of 10 ms; when the average additional delay is 50 ms or 75 ms, the inconsistency rate of Causality at the standard deviation of 30 ms is larger than that of 10 ms. The reason is that when the average additional delay is 100 ms or 125 ms (50 ms or 75 ms), the number of MUs which arrive within (beyond) their time limits increases as the standard deviation becomes larger.

From Fig. 8, we can see that the average difference in the positional error of NC changes from negative into positive when the average additional delay exceeds 75 ms. The reason is as follows. Since the average additional delay from terminal 2 to terminal 1 is set to 75 ms, the positional error of car 1 is smaller than that of car 2 when the average additional delay from terminal 1 to terminal 2 is smaller than 75 ms. However, the positional error of car 1 becomes larger than that of car 2 when the average additional delay exceeds 75 ms. We also observe in Fig. 8 that the average difference in the positional error of Causality suddenly jumps up when the average additional delay exceeds 100 ms. This is because as described in the case of Fig. 7, the number of discarded MUs increases largely at terminal 2 since the MUs do not arrive until their time limits. In Fig. 8, we further find that the average differences of DR and DR+Causality are almost zero. Therefore, the fairness among players is almost perfectly maintained in the two schemes.

From the above considerations, we can say that DR+Causality, which is a scheme which carries out the $\Delta$-causality control and dead-reckoning together, can retain the consistency and fairness among players in better condition than the other three schemes even when the network delay is large. By using smaller values of $T_{\rm dr}$ and by employing high-order prediction and convergence over multiple times, we can further improve the consistency and fairness of the scheme.

## 5.2 Influence of packet loss

We show the inconsistency rate of the positional relations of the two cars and the average difference in the positional error between the two cars versus the packet loss rate from terminal 1 to terminal 2 in Figs. 9 and 10, respectively.

Figure 9 reveals that NC has the largest inconsistency rate among the four schemes, and DR+Causality almost the smallest. Also, we can confirm that the inconsistency rates of DR and DR+Causality are hardly dependent on the packet loss rate. On the other hand, the inconsistency rate of Causality increases slightly as the average additional delay becomes larger; it should be noted that the increasing

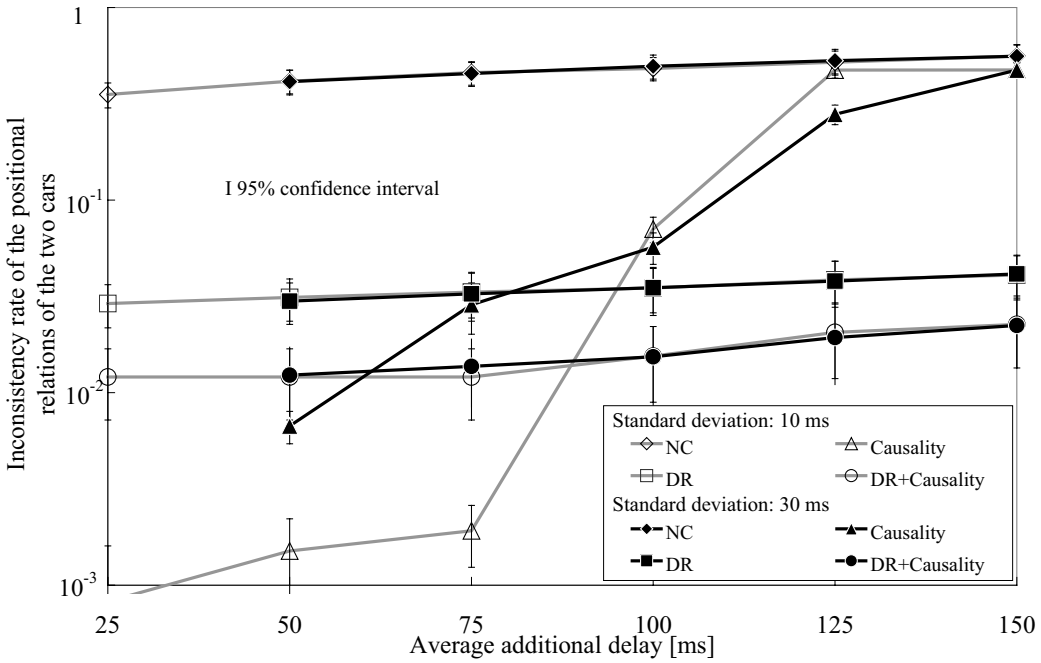



**Figure 7: Inconsistency rate of the positional relations of the two cars versus the average additional delay.**

amount of Causality is almost the same as that of NC. The reason is as follows. When MUs (packets) are lost, DR and DR+Causality can output the lost MUs by prediction, while Causality cannot output the MUs. Therefore, in Causality, as the packet loss rate increases, the influence of MUs which are lost becomes larger; thus, the inconsistency rate also becomes higher.

In Fig. 10, we observe that the average differences in the positional error of NC and Causality change from negative into positive when the packet loss rate exceeds 10 %. The reason is as follows. Since the packet loss rate from terminal 2 to terminal 1 is set to 10 %, the positional error of car 1 is smaller than that of car 2 when the packet loss rate from terminal 1 to terminal 2 is smaller than 10 %. However, the positional error of car 1 becomes larger than that of car 2 when the packet loss rate exceeds 10 %. We also notice in the figure that the average differences of DR and DR+Causality are almost zero. Therefore, the fairness among players is almost perfectly maintained in DR and DR+Causality in the experimental system.

# 6. CONCLUSIONS

In this paper, we discussed the consistency among players for networked racing games. By experiment, we made a performance comparison among four schemes and examined the influences of network latency and packet loss on the consistency. The four schemes are the $\Delta$-causality scheme (Causality), the dead-reckoning scheme (DR), a scheme which carries out the $\Delta$-causality control and dead-reckoning together (DR+Causality), and a scheme which performs neither of the two types of control (NC). As a result, we found that DR+Causality can keep the consistency and fairness among players in good condition.

As the next step of our research, we need to investigate the performance in the case where there exist a number of players and/or in a variety of network environments by simulation. We also plan to make a performance comparison between the client-server model and the peer-to-peer model as in [9]. Furthermore, we will study consistency control as in [2] and [12] so as to improve the performance, and we will handle other types of networked real-time games such as networked shooting games.

# 7. ACKNOWLEDGMENT

This work was supported by the Grant-In-Aid for Scientific Research (C) of Japan Society for the Promotion of Science under Grant 16560331.

# 8. REFERENCES

[1] L. Pantel and L. C. Wolf. On the impact of delay on real-time multiplayer games. In *Proceedings of ACM NetGames'02*, pages 23–29. April 2002.

[2] J. Vogel and M. Mauve. Consistency control for distributed interactive media. In *Proceedings of ACM Multimedia'01*, pages 221–230. September/October 2001.

[3] Y. Ishibashi and S. Tasaka. A media synchronization scheme with causality control in network environments. In *Proceedings of IEEE LCN'99*, pages 232–241. October 1999.

[4] Y. Lin, K. Guo, and S. Paul. Sync-MS: Synchronized messaging service for real-time multi-player distributed games. In *Proceedings of IEEE ICNP'02*, pages 155–164. November 2002.

[5] K. Guo, S. Mukherjee, S. Rangarajan and S. Paul. A fair message exchange framework for distributed

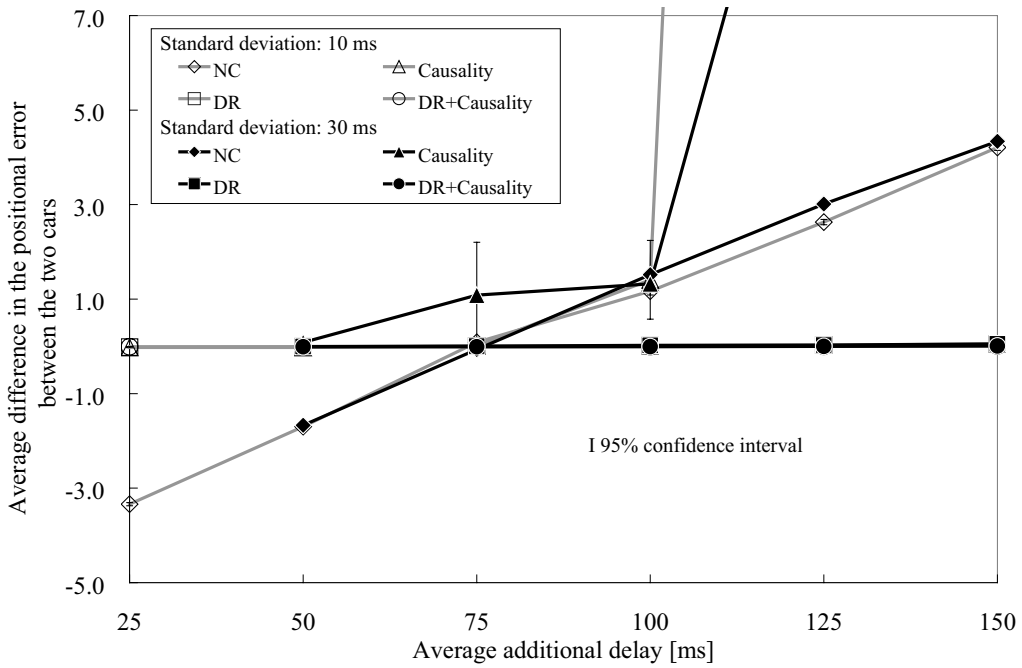



**Figure 8: Average difference in the positional error between the two cars versus the average additional delay.**

multi-player games. In *Proceedings of ACM NetGames'03*, pages 21–33. May 2003.

[6] C. Diot and L. Gautier. A distributed architecture for multiplayer interactive application on the Internet. *IEEE Network*, 13(4):6–15, July/August 1999.

[7] L. Pantel and L. C. Wolf. On the suitability of dead reckoning schemes for games. In *Proceedings of ACM NetGames'02*, pages 79–84. April 2002.

[8] S. Singhal and M. Zyda. *Networked virtual environments: Design and implementation*. ACM Press, SIGGRAPH Series, pages 127–144. 1999.

[9] Y. Ishibashi and S. Tasaka. Causality and media synchronization control for networked multimedia games: Centralized versus distributed. In *Proceedings of ACM NetGames'03*, pages 34–43. May 2003.

[10] D. L. Mills. Network time protocol (version 3) specification, implementation and analysis. RFC-1305, March 1992.

[11] M. Carson and D. Santay. NIST Net - A Linux-based network emulation tool. *ACM SIGCOMM Computer Communication Review*, 33(3):111–126, July 2003.

[12] E. Cronin, B. Filstrup, A. R. Kurc and S. Jamin. An efficient synchronization mechanism for mirrored game architectures. In *Proceedings of ACM NetGames'02*, pages 67–73. April 2002.

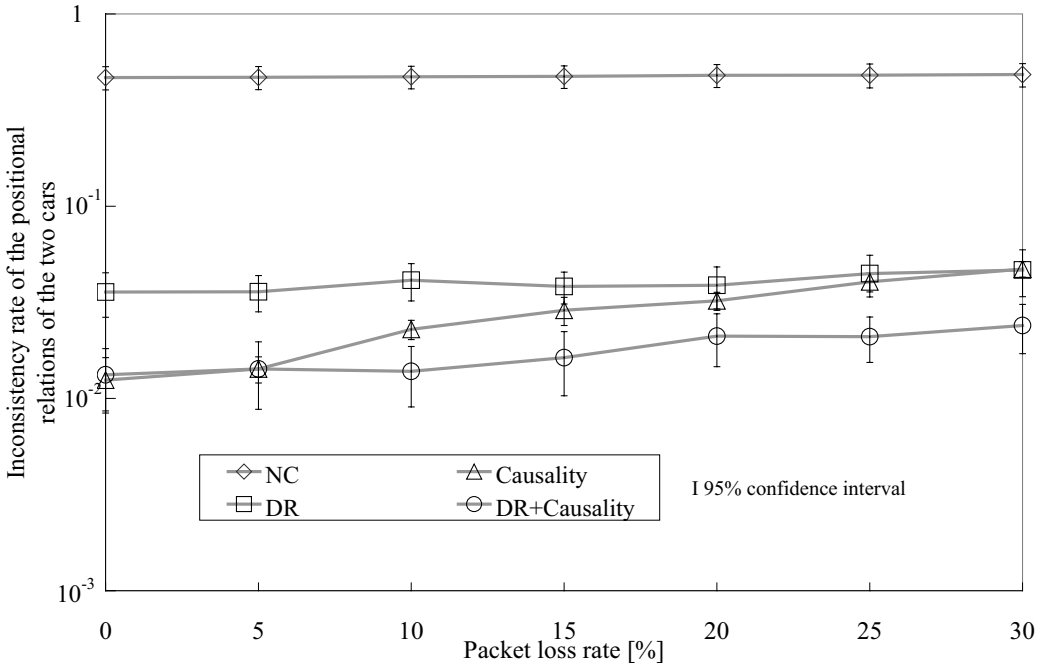



**Figure 9: Inconsistency rate of the positional relations of the two cars versus the packet loss rate.**

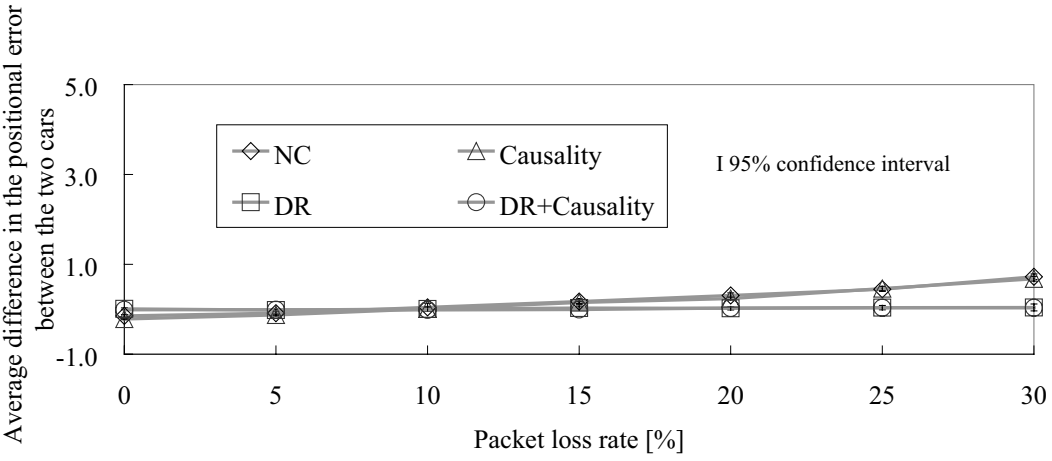



**Figure 10: Average difference in the positional error between the two cars versus the packet loss rate.**