Шаклин Дмитрий Нилович

ПРОБЛЕМА СИНХРОНИЗАЦИИ ПО ВРЕМЕНИ КЛИЕНТ-СЕРВЕРНЫХ СЕТЕВЫХ ВИДЕОИГР

В данной статье рассказывается о проблемах, выявленных в сетевых видеоиграх. Рассматриваются логические проблемы, возникающие вследствие того, что пропускная способность сети ограничена, и время прохождения пакетов от сервера к клиентам различается. Разбирается система с авторитарным сервером с одним клиентом и с несколькими. Предлагаются методы решения вышеуказанных проблем, в том числе метод предсказания и некоторые его вариации.

Адрес статьи: www.gramota.net/materials/1/2016/1/39.html

Источник

Альманах современной науки и образования

Тамбов: Грамота, 2016. № 1 (103). С. 129-133. ISSN 1993-5552.

Адрес журнала: www.gramota.net/editions/1.html

Содержание данного номера журнала: www.gramota.net/materials/1/2016/1/

7. **Чеснокова Л. В.** Национально-культурная специфика немецкого концепта Heimat (Родина) // Исторические, философские, политические и юридические науки, культурология и искусствоведение. Вопросы теории и практики. Тамбов: Грамота, 2015. № 5 (55). Ч. 2. С. 204-207.
8. **Элиаде М.** Мифы, сновидения, мистерии / пер. с англ. А. П. Хомик. Киев: Рефл-бук; Ваклер, 1996. 288 с.
9. **Bernet R.** Heimweh und Nostalgie // Utopie Heimat. Psychiatrische und kulturphilosophische Zugänge / hrsg. von M. Heinze, D. Quadflieg und M. Bührig. Berlin: Parodos Verlag, 2011. S. 87-102.
10. **Dorn T., Wagner R.** Die deutsche Seele. München, 2011. 560 S.
11. **Heinze M.** Heimat und Sozialpsychiatrie? // Utopie Heimat. Psychiatrische und kulturphilosophische Zugänge / hrsg. von M. Heinze, D. Quadflieg und M. Bührig. Berlin: Parodos Verlag, 2011. S. 11-22.
12. **Joisten K.** Auf der Suche nach Heimat // Utopie Heimat. Psychiatrische und kulturphilosophische Zugänge / hrsg. von M. Heinze, D. Quadflieg und M. Bührig. Berlin: Parodos Verlag, 2011. S. 103-124.
13. **Quadflieg D.** Begegnungen mit dem Unheimlichen – Heidegger trifft Freud // Utopie Heimat. Psychiatrische und kulturphilosophische Zugänge / hrsg. von M. Heinze, D. Quadflieg und M. Bührig. Berlin: Parodos Verlag, 2011. S. 125-136.
14. **Schlink B.** Heimat als Utopie [Электронный ресурс]. URL: http://www.bookdepository.com/Heimat-als-Utopie-Bernhard-Schlink/9783518066133 (дата обращения: 08.04.2014).

PHILOSOPHICAL MEANINGS OF NOSTALGIA AS YEARNING FOR HOME

Chesnokova Lesya Vladimirovna
Omsk State Pedagogical University
L.Tchesnokova@mail.ru

The article examines the feeling of nostalgia as yearning for home, for the period of happy childhood. In existential philosophy (Heidegger) the experience of nostalgia is connected with yearning for genuine existence, a human's craving for freeing oneself from not being grounded. Not only spatial, but also insurmountable temporal moving away from native land, which reminds of human finiteness and mortality, is associated with nostalgia. That's why the cause of nostalgia is a wish to stop the flight of time.

Key words and phrases: nostalgia; homeland; home; feeling of security; nostalgic yearning; existential homelessness; human's mortality.

УДК 004.946
Технические науки

В данной статье рассказывается о проблемах, выявленных в сетевых видеоиграх. Рассматриваются логические проблемы, возникающие вследствие того, что пропускная способность сети ограничена, и время прохождения пакетов от сервера к клиентам различается. Разбирается система с авторитарным сервером с одним клиентом и с несколькими. Предлагаются методы решения вышеуказанных проблем, в том числе метод предсказания и некоторые его вариации.

Ключевые слова и фразы: синхронизация по времени; клиент – сервер; видеоигры; *Dead Reckoning*; интерполяция; метод предсказания; лаго-компенсация.

Шаклин Дмитрий Нилович
Национальный исследовательский университет «Московский институт электронной техники»
shaklin.ru@mail.ru

ПРОБЛЕМА СИНХРОНИЗАЦИИ ПО ВРЕМЕНИ КЛИЕНТ-СЕРВЕРНЫХ СЕТЕВЫХ ВИДЕОИГР

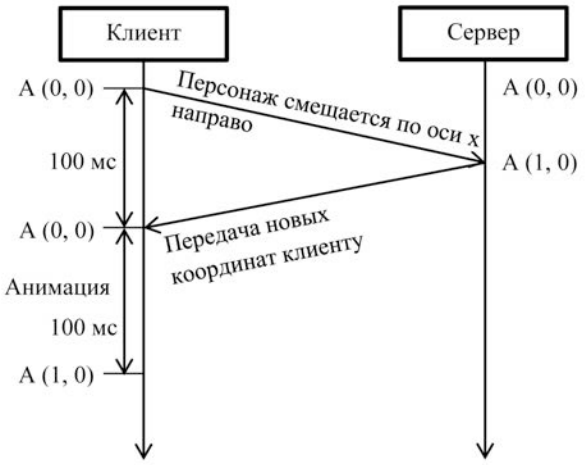
Популярность видеоигр в мире продолжает неуклонно расти. При всей своей первоначальной развлекательной направленности, целью многих игр на данный момент являются анализ, обучение, мотивация и развитие людей. Сегодня существуют такие игры, которые помогают изучать сложные социальные и экологические проблемы, – начиная от загрязнения Чесапикского залива и заканчивая судьбой беженцев в суданской провинции Дарфур. Используются видеоигры также и в медицине, например, как дополнительное болеутоляющее средство, так как погружение в мир пришельцев и монстров психологически отвлекает от боли. И с колоссальными темпами развития Интернета на нашей планете растет такими же темпами и количество сетевых видеоигр.

В общем виде структура сетевой видеоигры состоит из сервера и подключенных к нему клиентов. Сервер является главной структурой, определяющей поведение всех клиентов. Такой сервер называется авторитарным, так как клиенты без команды от сервера не совершают никаких действий.

По причине того, что пропускная способность сети ограничена, и время прохождения пакетов данных от сервера к клиенту отличается, появляются различные логические проблемы, вследствие чего игроки могут чувствовать себя некомфортно при использовании в игре клиент-серверной модели.

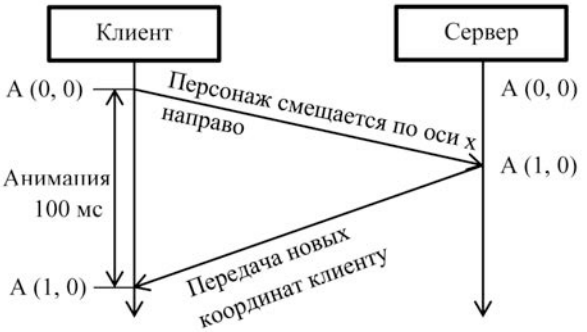
Предсказание

Представим следующую ситуацию. В игре клиент передвигается из одной позиции в другую. В данной работе будем рассматривать задержку (пинг (*ping*, англ.) – время реакции на отправленный пакет) сети от 100 мс и выше, так как это уже достаточно большое время для появления проблем в работе игры по сети. Сама анимация передвижения персонажа клиента также занимает 100 мс.



Когда клиент отправляет команду на движение серверу, подтверждение команды он получит лишь через 100 мс, и, соответственно, только через еще 100 мс (время, равное времени анимации передвижения) управляемый клиентом персонаж прибудет на нужную позицию.

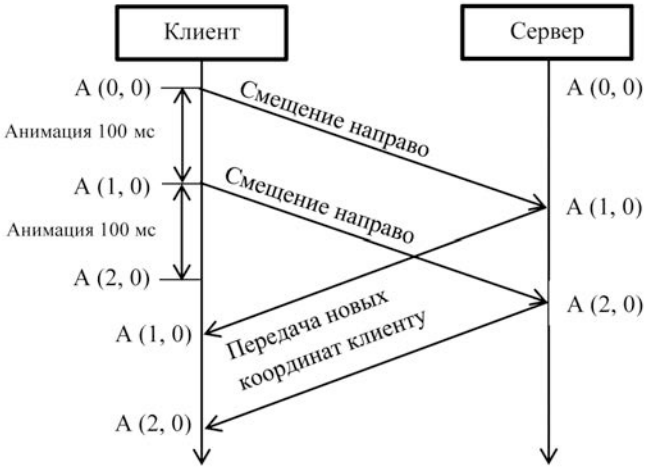
Задержка между отправкой команды на движение и его отображением визуально создает неестественное чувство. Вместо того чтобы дожидаться от сервера информации об изменении собственной позиции, клиент может предсказывать результаты собственных пользовательских команд с помощью точно такого же кода, который использует для этой цели сервер [2]. То есть клиент для предсказания должен иметь запущенную копию игры на своей локальной машине, а не только сообщать и получать от сервера данные [3].



В этом случае нет никакой задержки между клиентским вводом и отображением результата на экране.

Проблемы предсказания

В рассмотренном выше примере все числа подобраны специально таким образом, чтобы не возникало проблем. Но рассмотрим другой случай. Пусть задержка составляет 250 мс, а анимация движения персонажа – так же 100 мс. Предположим, что клиент два раза подряд дал команду двигаться персонажу направо.



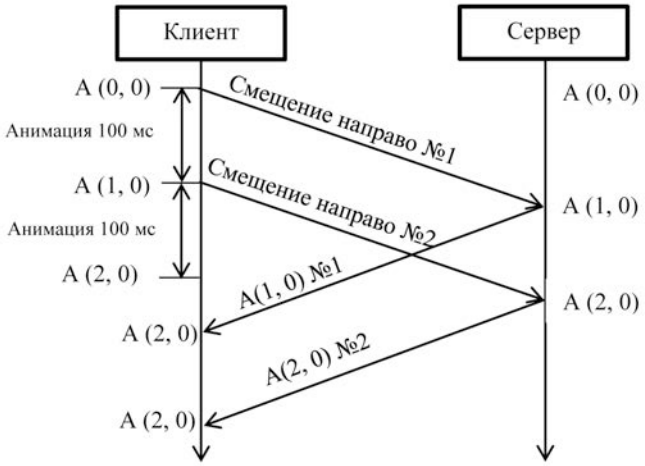
Проблема появляется в момент времени t=250 мс, когда от сервера приходит первый ответ. К этому времени предсказанное самим клиентом свое положение x=2, но от сервера приходит сообщение, указывающее,

что положение клиента x=1. И так как все решения принимает сервер, клиент должен передвинуть персонажа обратно на позицию x=1. Но когда в момент времени t=350 приходит сообщение от сервера с новым состоянием персонажа x=2, персонаж снова должен совершить скачок.

С точки зрения игрока подобные скачки выглядят совершенно неестественно.

Согласование с сервером

Ключ решения этой проблемы скрывается в том, что клиент видит все свои действия в настоящем времени, но, из-за задержки во времени, от сервера приходит состояние игры в прошлом.



Для решения данной проблемы первое, что нужно сделать, – это к каждому пакету, отправляемому от клиента к серверу, добавлять порядковый номер. Например, на диаграмме выше первый запрос к серверу помечен № 1, второй, соответственно, – № 2.

Теперь при t=250 мс сервер отвечает, что, основываясь на данных, полученных в сообщении № 1, позиция клиента x=1. И так как сервер играет решающую роль в принятии решений, он устанавливает позицию для персонажа в x=1.

Далее нужно сделать так, чтобы клиент запоминал каждое свое перемещение, формируя таким образом историю своих состояний.

И в момент, когда приходит от сервера первый ответ, на стороне клиента можно восстановить следующее положение после отправки сообщения серверу. Если же они совпадают, то в момент времени t=250 мс исправлять значение x на 1 не нужно. То же самое и для момента времени t=350 мс.

Таким образом, согласование с сервером решает эту проблему.

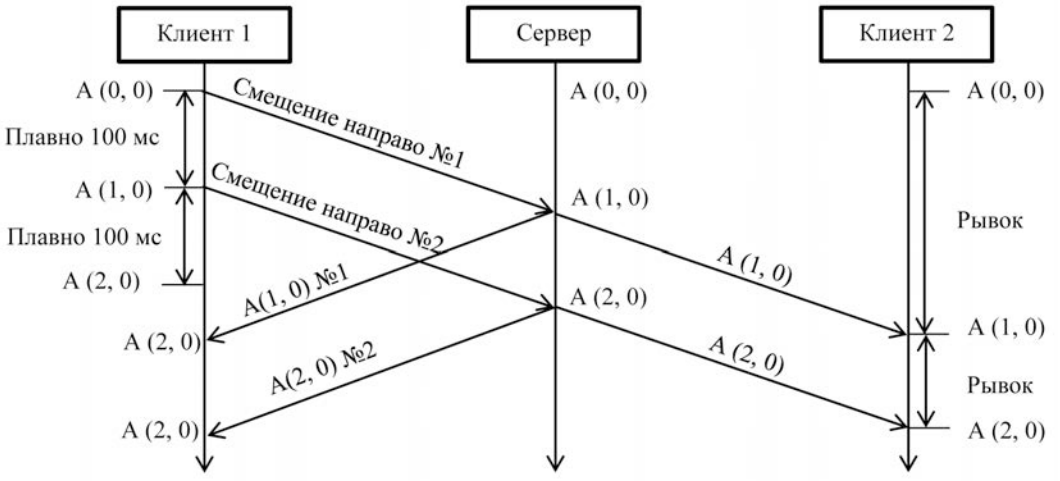
Сервер и несколько клиентов

Предсказывать можно свои действия и действия остальных клиентов. Что касается предсказания своих действий, – все немного проще. Ведь можно у себя на клиенте эмулировать тот же мир, что эмулирует и сервер. Например, можно эмулировать физику, но только ту, которая не влияет на других клиентов.

Есть и другие подходы. Можно воспользоваться экстраполяцией на основе предыдущих координат. Или, например, на основе координат и скорости или ускорения можно просчитать параметры следующего положения.

Но гораздо тяжелее прогнозировать движение других клиентов. Практически это – невозможно. Мы можем предсказать только то, что уже знаем. Нельзя точно сказать, будет ли другой клиент прыгать или пойдет налево [1].

Рассмотрим на следующей схеме пример того, как видит один клиент другого.



Таким образом, с точки зрения первого клиента, все выглядит достаточно гладко, так как прогноз для самого себя работает стабильно. Но то, как клиент 2 видит клиента 1 сложно назвать приемлемым. Напомним, что мы рассматриваем системы с задержкой не менее 100 мс. Клиент 2 будет видеть, что положение клиента 1 меняется рывками. В начальный момент времени другой игрок находится в одной позиции и через 250 мс рывком перемещается в другую позицию, которая была только что продиктована сервером. В зависимости от типа игры, есть несколько способов решения этой проблемы.

Один из них называется *Dead Reckoning* (счисление координат). Суть этого метода заключается в том, что мы пытаемся предсказать действие объекта, исходя только из его характеристик. Например, зная начальное положение и скорость объекта, мы можем построить его траекторию.

Представим симулятор гоночных автомобилей. Специфика этой игры в том, что гоночные автомобили двигаются достаточно быстро и поэтому предсказуемы. То есть, если автомобиль мчится с огромной скоростью, то за секунду он может или немного замедлиться, или немного ускориться, или немного повернуть направо или налево. Ключевое слово здесь «немного». Маневренность на высоких скоростях очень сильно зависит от предыдущей позиции, скорости и направления вне зависимости от того, что делает игрок в данный момент.

В этом случае те данные, которые клиент получает от сервера о других игроках и о себе, не будут сильно отличаться от реальных. И при какой-то небольшой разнице позицию автомобиля будет несложно откорректировать.

Но, с другой стороны, если клиент врежется во что-нибудь, ошибка при предсказании будет достаточно серьезной.

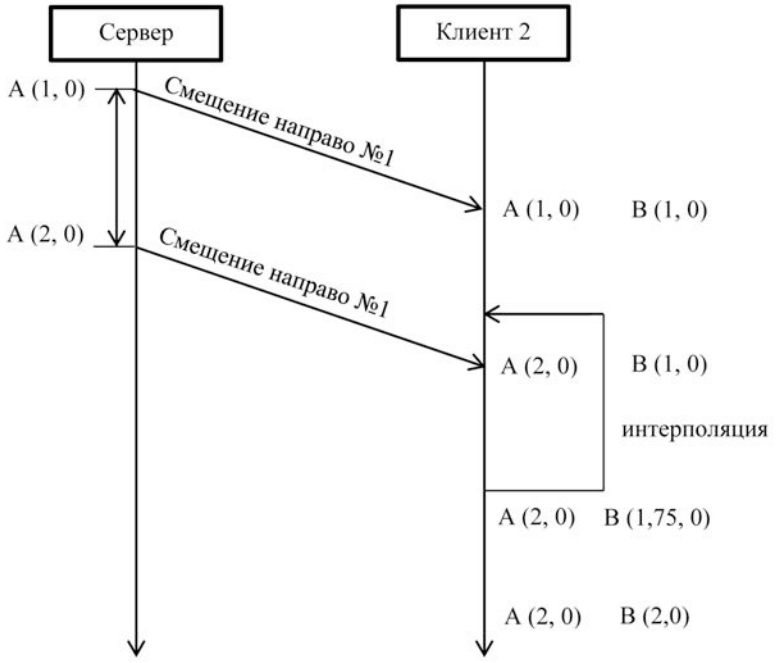
Таким образом, методика *Dead Reckoning* отлично подойдет для медленно изменяющихся ситуаций. Например, для симуляторов военных кораблей. Это понятие и появилось из морской навигации. Оно означает метод определения места (текущих координат) корабля по известным исходным координатам и параметрам движения.

Но есть ситуации, в которых *Dead Reckoning* не может быть применена совсем, – ситуации, когда направление или скорость персонажа клиента могут быстро измениться. Например, в 3D-шутерах игроки обычно бегают, поворачиваются, прыгают на высоких скоростях, и в этом случае *Dead Reckoning* является практически бесполезной, так как позиции и скорость уже невозможно предсказывать, основываясь на истории параметров.

И, в то же время, нельзя просто обновлять позиции других игроков каждые 100 мс (если взять это время за задержку). Будет казаться, что другие игроки постоянно дергаются.

Все, что мы имеем, – это позиции игроков каждые 100 мс. И задача заключается в том, как показать, что происходит между этими моментами. Чтобы достичь этого, весь рендеринг происходит в прошлом так, что клиент непрерывно вычисляет движение объектов между двумя последними принятыми пакетами [2].

Например, если клиент получает данные в момент времени t= 500, значит, у клиента уже есть данные о координатах других клиентов в момент времени t=400. Получается, что в промежуток времени между t=500 и t=600 можно показывать, что делали остальные игроки в промежутке с t=400 по t=500, используя для этого, например, различные методы интерполяции. Таким образом, мы показываем действительное положение других игроков, правда, немного запаздывая по времени.



На диаграмме выше можно проследить, как на втором клиенте происходит отображение первого клиента из его предыдущих положений.

Таким образом, мы получаем следующую картину. Клиент свои собственные передвижения наблюдает в настоящем, а действия других клиентов он видит в прошлом. И так каждый клиент видит всех остальных клиентов в их прошлых состояниях.

Лаго-компенсация

Для рассмотренного выше метода интерполяции есть исключения, если важна точность. Например, это – игра-шутер. И когда один игрок стреляет в другого, то получается, он стреляет в прошлое состояние другого игрока, в то положение, где он был 100 мс назад. Может так получиться, что клиент, который сделал выстрел, будет считать себя попавшим в другого клиента. Но тот, в кого стреляли, на момент попадания уже находится в других координатах, и поэтому посчитает, что попасть в него не могли. Как же разрешить серверу эту ситуацию? В этом случае можно воспользоваться методом, который носит название «лаго-компенсация». Заключается он в том, что сервер перемещает всех игроков на позиции, которые они занимали в момент исполнения команды. Пользовательская команда исполняется, и попадание засчитывается [Там же].

Тем не менее, клиенту, в которого попали, может казаться это несправедливым. Ведь с его стороны в него явно не могли попасть. И здесь нет конкретного решения. Пакеты с данными не могут путешествовать со скоростью света. Поэтому задержка между клиентом и сервером должна быть в тех пределах, которые не дают подстреленному игроку «заподозрить неладное». Клиентов же с задержкой, которая совсем не дает им и другим игрокам «нормально» чувствовать себя в игре, даже с учетом предложенных методов, не стоит допускать к игровой сессии.

Выводы

Были рассмотрены некоторые из основных проблем синхронизации клиент-серверных сетевых игр.

Разобраны различные методы разрешения логических проблем, возникающих вследствие ограниченности пропускной способности сети, в том числе метод предсказания с дополнениями. Среди рассмотренных решений – методы интерполяции, метод *Dead Reckoning*. Также освещен метод лаго-компенсации.

На основании данного исследования можно сделать вывод, что с помощью различных методов есть возможность добиться хороших результатов при некоторых условиях. Например, при постановке ограничений на скорость передачи пакетов. Но, в общем случае, нет способа со стопроцентной вероятностью предотвратить все проблемы, которые могут возникнуть при синхронизации клиентов.

Многие ситуации придется решать с помощью различных приемов, основанных на неточностях и несовершенствах человеческого восприятия.

Список литературы

1. **Карлов А.** Player.IO. Интерполяция или удивительный мир обмана [Электронный ресурс]. URL: http://www.ant-karlov.ru/PlayerIO-interpolyatsiya-ili-udivitelniy-mir-obmana.html (дата обращения: 10.11.2015).
2. **Сетевое программирование в Source** [Электронный ресурс]. URL: https://developer.valvesoftware.com/wiki/Source_Multiplayer_Networking:ru (дата обращения: 15.11.2015).
3. **Fiedler G.** What Every Programmer Needs to Know about Game Networking [Электронный ресурс]. URL: http://gafferongames.com/networking-for-game-programmers/what-every-programmer-needs-to-know-about-game-networking/ (дата обращения: 17.11.2015).

PROBLEM OF TIMING OF CLIENT-SERVER NETWORK VIDEO GAMES

Shaklin Dmitrii Nilovich
National Research University of Electronic Technology
shaklin.ru@mail.ru

This article describes the problems identified in network video games. The author considers the logical problems occurring because the network bandwidth is limited and the time of packets passage from the server to the client is different. The paper also clarifies the system with the authoritarian server with one or several clients. The methods of solving the above mentioned problems are proposed including the method of prediction and some of its variations.

Key words and phrases: timing; client-server; video games; Dead Reckoning; interpolation; method of prediction; lag compensation.