مجله پژوهش های معاصر در علوم و تحقیقات
سال اول، شماره ۷، بهمن ۱۳۹۸
ISSN: ۷۶۷۶-۵۷۶٤
www.jocrisar.ir

بهبود زمان پاسخ بازی‌های ابری با کمک الگوریتم‌های لایه برداری و پیش بینی حرکت

فاطمه بهاری نسب [۱] ، مهرداد مائین [۲]

[۱] کارشناس ارشد مهندسی فناوری اطلاعات گرایش شبکه های کامپیوتری دانشگاه علوم تحقیقات تهران
[۲] دکترای تخصصی عضو هیئت علمی دانشگاه آزاد اسلامی واحد یادگار امام خمینی (ره)

چکیده

با رشد صنعت بازی‌های ویدئویی، بازارهای جدید و فن آوری‌ها در حال ظهور هستند. بازی‌های الکترونیکی نسل جدید به طور فزاینده‌ای نیاز به پردازش بیشتر و کارت‌های قدرتمند ویدئویی دارند. راه حلی که بیشترین توجه را به خود جلب می‌کند، بازی ابری است. بازی ابری روشی است که بازیکن یک فرمان را اجرا می‌کند واطلاعات فرستاده شده و پردازش شده از راه دور بر روی ابر قرار می گیرد و پس از آن تصاویر به عنوان یک جریان ویدیوئی با استفاده از اینترنت پخش می شود. نتیجه بازی از طریق مدیر بازی به نام Cloud Manager بیان شده است. مدیر ابر چارچوبیست که از تکنیک‌های ذخیره سازی لایه در پس زمینه استفاده می‌کند. این بازی باید در لایه‌های مختلف کد گذاری شود تا مدیر ابر بتواند از تکنیک‌های ذخیره‌سازی لایه‌ای در پس‌زمینه و برای پیش‌بینی وضعیت آینده،از ماتریس پیش‌بینی در لایه ی کارکتر استفاده کند. تکنیک ذخیره سازی برای کاهش حجم کار کدگذاری بر روی سرور و استفاده از پهنای باند شبکه با انتقال پیام بیت کمتر است. با توجه به نیاز به بهبود زمان پاسخ در بازی‌ها و زمان پردازش کمتر در آن ها، در این مقاله سعی بر آن شده است که با استفاده از الگوریتم لایه برداری به بهبود این نقص کمک شود.

واژه‌های کلیدی: محاسبه، ابر، بازی، پیش پردازش، ذخیره سازی لایه، پیش بینی حرکت

۱. مقدمه

با توسعه صنعت بازی‌های فریمی ، بازارهای جدید و فن‌آوری‌ها در حال ظهور هستند . بازی‌های الکترونیکی نسل جدیدی که بطور فزاینده نیازمند پردازش و کارت فریمی قوی تری هستند. راه‌حلی که برجستگی بیشتری دارد، بازی ابری است. بازی‌های ابری به عنوان نوعی از خدمات تعریف شده است که در آن بازی الکترونیکی، در زمان میزبانی یک ابر، به کاربران امکان می‌دهد تا از طریق دسترسی به اینترنت از انجام یک بازی در دستگاه‌های ناهمگن لذت ببرند. یک چالش برای همه سیستم‌های بازی ابری این است که آنها باید مدت زمان تاخیر سیستم را حفظ کنند. زمان بین ارسال فرمان توسط مشتری و دریافت جریان ویدئو از بازی برایبهبود کیفیت تجربه حیاتی است [۱]، [۲]. با پیشرفت های اخیر در تکنولوژی ابری ایده بازی ابری به واقعیت تبدیل شده است، محاسبات ابری میتواند توسط برنامه های مختلف قابل استفاده باشد، در میان آنها بازی های کامپیوتری که تشنه منابع هستند به عنوان برنامه قاتل برای محاسبات ابری شناخته شده اند، بازی ابری یا بازی بر اساس تقاضا یک نوع از بازی های آنلاین است که اجازه میدهد بر اساس تقاضا محتوی بازی بر روی دستگاه های غیر تخصصی مثل تلویزیون هوشمند وغیره، جریانسازی شود (معمولا صدا و تصویر) یکی از مزایای مهم این است که هیچ دانلود و نصب بازی مورد نیاز نیست با توجه به اینکه بازی واقعی روی سرور شرکت ارائه دهنده ذخیره و اجرا شده و فقط خروجی آن به سمت مشتری جاری می شود. در این عمل همچنین دسترسی به بازی تقریبا فوری است از مزایای دیگر این است که هیچ محدودیتی برای دستگاه های پایانی (قابلیت سخت افزاری، سیستم عامل و غیره) وجود ندارد، توسعه دهندگان از این نظر که لازم نیست نسخه های متعدد بازی را برای دستگاه های مختلف توسعه دهند این موضوع بازیکن ها را قادر میسازد که تقریبا بر روی هر دستگاهی بازی کنند.علی رغم ویژگی‌های فوق العاده بازی‌های ابری، چندین چالش اساسی در مسیر توسعه این دسته از بازی‌های آنلاین وجود دارد. مهمترین و اولین چالش موجود در مورد سیستم عامل و بستر تست بازی‌های ابری است که باید برای ارزیابی‌های جامع از عملکرد آنها اعمال گردد. ارزیابی‌ها شامل اندازه گیری معیارهای کیفیت خدمات از قبیل مصرف انرژی و معیارهای شبکه، و همچنین معیارهای کیفیت تجربه مثل تجربه حاصل از بازی برای بازی کننده می‌باشد. مسئله ژانرهای گوناگون بازی و همچنین بهینه سازی اجزای مختلف از طریق سیستم عامل بازی‌های ابری از دیگر موضوعات چالش برانگیز این حوزه است.یکی دیگر از چالش‌های مهم در بحث بازی‌های ابری و بطور کلی در محاسبات ابری، زمان پاسخ برنامه می‌باشد.در این سیستم‌ها منطق بازی و عملیات رندرینگ از راه دور در یک سرور ابری پردازش می‌شود و خروجی‌های صوتی و تصویری به یک مشتری ضعیف با قابلیت‌های محاسباتی محدود مانند تلفن‌های همراه، رایانه‌های کم حجم و حتی تلویزیون‌های هوشمند روانه می‌شود.سیستم باید اقدامات یک بازیکن را جمع آوری کند ، آنها را به ابر منتقل کند ، اقدامات را پردازش کند ، نتایج را ارائه دهد ، تغییرات بازی را رمزگذاری و فشرده سازی کند ، و ویدیو (صحنه های بازی) را به پخش کننده منتقل کند. چالش‌های زیادی در عملیات اجرا و تامین بازی‌های ابری وجود دارد، بالاخص زمانی که تلاش برای کاهش زمان رمزگذاری سرور و میزان ارسال بیت ویدئو مد نظر قرار داشته باشد. در این مقاله، روش جدید بهینه سازی بر اساس لایه‌های تصویری که ذخیره سازی می‌شود، می باشد. در بخش دوم این مقاله، مراحل الگوریتم لایه برداری پیشنهادی و همچنین مراحل پیش بینی حرکت با استفاده از تعریف ماتریس پیش بینی شرح داده می‌شوند و در بخش سوم اعتبارسنجی و تحلیل نتایج و نهایتا در بخش چهارم نتیجه گیری ارائه شده است[۳-۸].

۱,۲ روش پیشنهادی

رویکرد جدید مد نظر این مقاله، نوعی لایه برداری را پیشنهاد می کند که تکنیک ذخیره سازی برای کاهش حجم کار کدگذاری بر روی سرور و استفاده از پهنای باند شبکه با انتقال پیام بیت کمتر است. بنابراین لازم است تا لایه های مختلف بازی به طور جداگانه رمزگذاری شده و نتیجه ذخیره گردد بطوریکه در فریم های بعدی، اگر یک لایه اصلاح نشده باشد مانند یک پس زمینه ثابت، دوباره توسط سرور رمزگذاری مجدد نشود. آخرین اطلاعات برای هر لایه نیز در سمت سرویس گیرنده ذخیره می شود، بنابراین وقتی لایه

تغییر نمی کند، نیازی به انتقال مجدد نیست. دلیل رمزگذاری جداگانه پس زمینه و پیش زمینه بازی، امکان ذخیره سازی لایه‌های تغییر نیافته و رمزگذاری و جریان سازی پس زمینه برای زمان‌هایی که آن تغییرات قابل توجه هستند.

در ادامه مراحل الگوریتم لایه برداری پیشنهادی و همچنین مراحل پیش بینی حرکت با استفاده از تعریف ماتریس پیش بینی شرح داده می‌شوند.

۲,۲ تکنیک لایه بندی پیشنهادی

دلیل رمزگذاری جداگانه پس زمینه و پیش زمینه بازی، امکان ذخیره سازی لایه‌های تغییر نیافته و رمزگذاری و جریان سازی پس زمینه برای زمان‌هایی که آن تغییرات قابل توجه هستند. تکنیک لایه بندی مورد نظر بر روی لایه‌هایی کار می‌کند که قبلا کدگذاری شده اند تا از طریق استفاده مجدد ، حجم بار رمزگذاری را کاهش دهد. علاوه بر این، فقط لایه‌های جدید رمز شده جریان سازی می‌شوند که این امر میزان بیت کلی ارسال شده از سرور به مشتری را کاهش می‌دهد.بطور کلی در این پژوهش جهت بررسی اعتباریک حافظه کش فرض شده که لایه‌های ذخیره شده دارای تنها اشیاء استاتیک بدون حرکت هستند، به طوری که می‌توان تنها بر روی پارامترهای دوربین تکیه کرد.

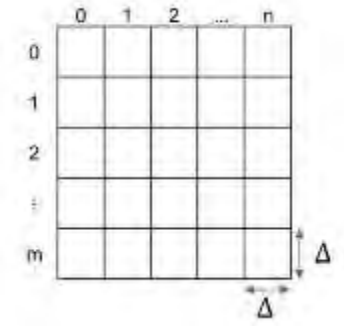
۳.۲ ایجاد ماتریس پیش بینی

الگوریتم پیش بینی حرکت در فضای ویژگی از طریق یک ماتریس پیش بینی که حرکت بعدی را ذخیره می‌کند، استفاده می کند. ماتریس پیش بینی از محیط بازی ساخته شده است و در این کار ماتریس پیش بینی T نامیده می‌شود.در محیط بازی، مقادیر پارامتر ارتفاع G و طول J از مختصات اجزاء در بالا، پایین، سمت چپ و سمت راست نقشه بازی بدست می‌آید. پارامتر ارتفاع G فاصله بین مولفه در انتهای بالا و جزء در انتهای پایین است. پارامتر طول J فاصله بین مولفه در انتهای سمت چپ و مولفه در انتهای سمت راست است. مقدار پارامتر Δ تغییر واحد کاراکتر در یکی از چهار جهت است: راست، چپ، بالا یا پایین.

همانطور که گفته شد، مقدار پارامتر Δ جابجایی واحد شخصیت در یکی از چهار جهت است: سمت راست، چپ، بالا یا پایین است .با مقادیرG، J و Δ، می‌توان ماتریس پیش بینیT را ایجاد کرد. تعداد ستون هایN و تعداد خطوط M می‌تواند با فرمول زیر محاسبه شود:

$$N=[(\frac{J}{\Delta})] \text{ و } M = \left[\left(\frac{G}{\Delta}\right)\right] \qquad (۱)$$

در هر آرایه، تعداد ردیف‌ها و ستون‌ها باید یک عدد صحیح باشد، بنابراین نتایج M و N گرد شده اند. ماتریسT در شکل ۱ نشان داده شده است، با توجه به اینکه اندیس خانه های ماتریس از صفر شروع می شود پس اندیس آخرین خانه در سطر و ستون برابر $m = M-۱$ و $n = N-۱$ است.



شکل ۱_ نمایش ماتریس پیش بینیT

برای انجام پیش بینی حرکت ، یک بردار $v_{i,j}$ به هر سلول اختصاص داده می شود ماتریس پیش بینی حرکت T، نشان دهنده حرکت بعدی شخصیت است .مقدار اولیه $v_{i,j}$ به این بستگی دارد که مدیر ابر از آموزش قبلی استفاده می کند و خانه های ماتریس از قبل پر شده است یا نه ، در صورتی که ماتریس پیش بینی از آموزش قبلی استفاده نکرده باشد مقادیر سلول های ماتریس با مقدار (۰،۰) پر می شود. بردار $v_{i,j}$ در سلول (i , j) به صورت $v_{i,j} = (x_{i,j}, y_{i,j})$ می تواند نشان داده شود .

۱.۳.۲ حرکت کاراکتر

با هربار حرکت کاراکتر مقدار بردار $v_{i,j} = (۰,۰)$ به روز می‌شود. از آنجا که محل کاراکتر در سلول $(i۱,j۱)$ متناسب است با مقدار بردار $v_{i۱,j۱} = (x_{i۱,j۱}, y_{i۱,j۱})$، با رفتن به سلول $(i۲,j۲)$ مقدار جدید بردار $v'_{i۱,j۱} = (x'_{i۱,j۱}, y'_{i۱,j۱})$ بصورت ذیل محاسبه می‌شود:

$$x'_{i۱,j۱} = x_{i۱,j۱} + (i۲ - i۱)$$

$$y'_{i۱,j۱} = y_{i۱,j۱} + (j۲ - j۱) \quad (۲)$$

۲.۳.۲ پیش بینی حرکت

برای انجام پیش بینی حرکت، سلول (i , j) برای وجود کاراکتر، در نظر گرفته خواهد شد. در حالی که مقدار سلول بصورت $v_{i,j} = (x_{i,j}, y_{i,j})$ محاسبه شده است مقدار زاویه و طول θ از مختصات دکارتی به مختصات قطبی تبدیل می‌شود.به منظور پیش بینی حرکت اگر مقدارr چیزی غیر از صفر باشد، حرکت ایجاد می‌شود و در غیر این صورت، اگر r برابر صفر باشد، پیش بینی حرکت ایجاد نخواهد شد. طول r نشان دهنده سطح اطمینان برداری $\overrightarrow{v_{i,j}}$ است، و بزرگتر و کوچکتر شدن مقدار در اثر مجموع بردار جدید و زاویه θ برای پیش بینی حرکت بعدی استفاده می‌شود. از آنجا که کاراکتر می‌تواند در تعداد مسیر محدودی حرکت کند، این هشت حرکت (بالا، پایین،چپ، راست و قطر)، توسط زاویه حرکت متناسب با T ↑ پیش بینی می‌شوند.

یک ماتریس پیش بینی در شکل ۲، نشان داده شده است که در آن ماتریسT دربرگیرنده مقداری از وضعیت حاصل از چند حرکت شکل گرفته توسط کاراکتر در فضای بازی

	۰	۱	۲	۳	۴	۵	۶	n...
۰	(۲,-۹)	(۱,-۵)	(۳,-۴)	(-۵,-۸)	(۰,۰)	(۰,۰)	(۵,-۲)	...
۱	(۴,-۸)	(۵,-۵)	(-۱,۹)	(-۲,-۵)	(۰,۰)	(۰,۰)	(۲۳,۳)	...
۲	(۹,-۲)	(۲۳,-۲)	(۳۰,-۲)	(۳۴,-۴)	(۳۴,۰)	(۳۴,۰)	(۳۲,۲)	...
۳	(۱۱,۳)	(۱۳,-۵)	(۴,۲۰)	(-۳,۲۱)	(۰,۰)	(۰,۰)	(۳,-۹)	...
m:	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋱

اندیس خانه ی مقصد (۱،۱)

اندیس خانه ی مقصد (۱،۲)

مقدار بردار : (-۱،۹)

زاویه θ : درجه ۹۶،۳۴

	۰	۱	۲	۳	۴	۵	۶	n...
۰	(۲,-۹)	(۱,-۵)	(۳,-۴)	(-۵,-۸)	(۰,۰)	(۰,۰)	(۵,-۲)	...
۱	(۴,-۸)	(۶,-۵)	(-۱,۹)	(-۲,-۵)	(۰,۰)	(۰,۰)	(۲۳,۳)	...
۲	(۹,-۲)	(۲۳,-۲)	(۳۰,-۲)	(۳۴,-۴)	(۳۴,۰)	(۳۴,۰)	(۳۲,۲)	...
۳	(۱۱,۳)	(۱۳,-۵)	(۴,۲۰)	(-۳,۲۱)	(۰,۰)	(۰,۰)	(۳,-۹)	...
⋮ m	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋱

	۰	۱	۲	۳	۴	۵	۶	n...
۰	(۲,-۹)	(۱,-۵)	(۳,-۴)	(-۵,-۸)	(۰,۰)	(۰,۰)	(۵,-۲)	...
۱	(۴,-۸)	(۵,-۵)	(۰,۸)	(-۲,-۵)	(۰,۰)	(۰,۰)	(۲۳,۳)	...
۲	(۹,-۲)	(۲۳,-۲)	(۳۰,-۲)	(۳۴,-۴)	(۳۴,۰)	(۳۴,۰)	(۳۲,۲)	...
۳	(۱۱,۳)	(۱۳,-۵)	(۴,۲۰)	(-۳,۲۱)	(۰,۰)	(۰,۰)	(۳,-۹)	...
⋮ m	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋱

اندیس خانه ی مقصد (۲،۳)

مقدار بردار : (۳۴,-۴)

زاویه θ : درجه ۶،۷۱

پیش بینی حرکت : به سمت راست

	۰	۱	۲	۳	۴	۵	۶	n...
۰	(۲,-۹)	(۱,-۵)	(۳,-۴)	(-۵,-۸)	(۰,۰)	(۰,۰)	(۵,-۲)	...
۱	(۴,-۸)	(۵,-۵)	(۰,۸)	(-۲,-۵)	(۰,۰)	(۰,۰)	(۲۳,۳)	...
۲	(۹,-۲)	(۲۳,-۲)	(۳۰,-۲)	(۳۵,-۴)	(۳۴,۰)	(۳۴,۰)	(۳۲,۲)	...
۳	(۱۱,۳)	(۱۳,-۵)	(۴,۲۰)	(-۳,۲۱)	(۰,۰)	(۰,۰)	(۳,-۹)	...
⋮ m	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋱

اندیس خانه ی مقصد (۲،۴)

مقدار بردار : (۳۴،۰)

زاویه θ : درجه ۰

پیش بینی حرکت : به سمت

شکل ۲ – نمونه‌ای از الگوریتم پیش بینی حرکت را بکار گرفته است. در این شکل ماتریس‌ها برای (الف) موقعیت اولیه کاراکتر. (ب) شخصیت به سمت راست حرکت می‌کند. (ج) شخصیت به مورب راست پایین‌تر حرکت می‌کند. (د) شخصیت به سمت راست حرکت می‌کند، مقدار دهی شده اند.

توجه داشته باشید که هر سلول مقدار منحصر به خود را دارد و آنها مستقل از یکدیگر هستند، و پیش بینی حرکت تنها به سلولی بستگی دارد که کاراکتر درآن موجود است. با اضافه شدن کاراکترها به فضای بازی، مقادیر بیشتری به ماتریس T اضافه خواهد شد و در نتیجه دقت پیش بینی حرکت بهتر می‌شود.در این الگوریتم جهت پیشرفت در پیش بینی حرکت برای بهبود الگوریتم پیش بینی حرکت، دو پیشنهاد ارائه شده است که در قسمت‌های ذیل به آنها اشاره می‌شود. پیش پردازش و ماتریس آموزش یا آموزش ماتریس پیش بینی

۱.۲.۳.۲ پیش پردازش

عملیات پیش پردازش یک بخش مهم و ابتدایی است. به عنوان مثال برای شکل ۲مورد زیر را در نظر بگیرید، کاراکتر ۱۰ حرکت را انجام می‌دهد و الگوریتم پیش بینی بواسطه عملیات پیش پردازش فریم‌ها به درستی حرکت‌های ۲، ۳، ۵، ۷ و ۹ را پیش بینی می‌کند. در حالی که میانگین زمان پاسخ کمتر است، هنگام استفاده از الگوریتم پیش بینی، تغییر زمان پاسخ می‌تواند موجب ایجاد jitter و در نتیجه احساس ناراحتی برای کاربر شود. همانطور که میدانیم delay و jitter بر عملکرد بازیکنان تاثیر منفی می گذارد.در فرایند پیشنهادی تنها با انجام یک بار پیش پردازش قبل از پردازش، تعداد مشخصی از حرکات همزمان بواسطه آن پیش بینی می‌شود.

۲.۲.۳.۲ ماتریس آموزش

راه دیگر برای بهبود الگوریتم پیش بینی حرکت این است که ماتریس آموزش با ماتریس پیش بینی ترکیب شود. ماتریس آموزش در این پژوهش توسط ماتریس U با ابعاد ماتریس T ارائه شده است. برای هر سلول، مقدار بردار $p_{i,j}$ اختصاص خواهد یافت.ماتریس آموزش را می‌توان با روش‌های مختلفی ایجاد کرد که نتیجه‌های مختلفی را ایجاد می‌کنند:

- پرتکرار ترین مسیر پیمایش شده توسط همه بازیکنانی که دوره را به پایان رسانده اند.
- مجموع تمام ماتریس T تولید شده توسط بازیکنان که دوره بازی را طی نموده اند
- مسیر با حداقل تعداد حرکت از نقطه شروع تا رسیدن به انتهای مسیر .

پس از ایجاد ماتریس آموزشی U، لازم است که ماتریس U در کنار ماتریس پیش بینی T تطابق یابد.

۱.۳ اعتبار سنجی

برای اعتبار سنجی نتایج، چندین و چند تست با بازی "ربات های خشمگین[۱]" از ابتدا تا پایان به روش های مختلف برای هر آزمون انجام شد .در آزمایشات ، تعداد فریم های پردازش شده توسط رمزگذار JPEG و زمان پردازش هر فریم ثبت شد . ابعاد پنجره ی بازی در همه ی تست ها ۶۴۰*۴۸۰ بوده .

۲.۳ بازی ربات خشمگین

ربات خشمگین یک بازی تک تیرانداز از بالا به پایین است، که در راستای تایید الگوریتم پیشنهادی ما از این بازی استفاده کردیم. این بازی با ژانر اکشن و دیدگاه همه جانبه است. کد این بازی در یونیتی به لایه های پس زمینه و شخصیت تقسیم می شود و اجرای آن توسط الگوریتم لایه برداری آسان و ممکن می شود .در طول بازی برای انجام شبیه سازی، دشمنان حذف شده اند و این امکان داده شده تا فقط روی پیش زمینه ی استاتیک و شخصیت تجزیه و تحلیل کنیم.در ضمن عملکرد تیراندازی هم غیرفعال است. یک مسیر ممکن که شخصیت بتواند بر روی نقشه حرکت کند توسط خط آبی که برای آموزش استفاده میشود، نشان داده می شود و کارکتر از نقطه سبز شروع به حرکت می کند و با رسیدن به نقطه نارنجی، تست پایان می یابد. ماتریس پیش بینی تولید شده برای تست ها دارای عرض J برابر ۱۵۸,۰۱۷۱ وارتفاع Gبرابر با ۱۰۲,۱۸۶۲ و مقدار Δ معادل ۱,۲۳۱۱ است بنابراین طبق فرمول ها مقدار M برابر ۸۳ و N برابر ۱۲۹ است. برای هردور بازی انجام شده، شخصیت اصلی از نقطه شروع به نقطه پایانی منتقل شد و شماره های فریم پردازش شده توسط رمزگذار JPEGو زمان پردازش هر فریم ثبت شد. میانگین زمان پاسخ بر حسب میلی ثانیه برای هر مورد بطور متوسط بدست آورده شده و در جدول شماره ۱ نشان داده شده است.

جدول ۱_ میانگین مقدار زمان پاسخ

موارد آزمایش شده	میانگین زمان پاسخگویی(ms)	انحراف	درصد بهبود زمان پاسخ نسبت به مورد اول (٪)
مورد اول	۶۷,۹۸	۱,۹۸	۰
مورد دوم	۶۱,۸۱	۱,۸۷	۹,۰۸
مورد سوم	۶۳,۹۸	۱,۸۸	۵,۸۸
مورد چهارم	۶۵,۲۲	۱,۹۹	۴,۰۶
مورد پنجم	۴۶,۹۸	۱,۸۹	۳۰,۸۹
مورد ششم	۵۳,۰۷	۱,۹۰	۲۱,۹۳
مورد هفتم	۶۵,۱۰	۱,۹۵	۴,۲۴

و همینطور تعداد پیش بینی حرکت و دقت پیش بینی در هر مورد در جدول ۲ نشان داده شده است .طول بردارها در ماتریس آموزش بصورت تجربی پنج تعریف شده است .

۷ مورد زیر درآزمایش ها مورد بررسی قرار گرفت که عبارتند از :

.مورد ۱: بدون چارچوب مدیر ابر.

[۱] Angry Bots

مورد ۲: استفاده از پیش پردازش به همراه پیش بینی حرکت.

مورد ۳: استفاده از پیش پردازش پس از پیش بینی سه حرکت پی در پی

مورد ۴: استفاده از پیش پردازش پس از پیش بینی پنج حرکت پی در پی

مورد ۵: استفاده از پیش آموزش و پیش پردازش با پیش بینی حرکت

مورد ۶: استفاده از پیش آموزش و پیش پردازش پس از پیش بینی سه حرکت پی در پی

مورد ۷: استفاده از پیش آموزش و پیش پردازش پس از پیش بینی پنج حرکت پی در پی

جدول ۲ _ دقت الگوریتم پیش بینی حرکت.

موارد آزمایش شده	یانگین تعداد حرکت در هر تست	تعداد فریم های از قبل پردازش شده	تعداد حرکات با پیش بینی درست	دقت پیش بینی حرکت
مورد اول	۲۷۶	۳۵	۳۰	۰,۸۵۷۱
مورد دوم	۲۵۹	۱۹	۱۵	۰,۷۸۹۵
مورد سوم	۲۶۷	۸	۵	۰,۶۲۵
مورد چهارم	۲۸۰	۸۰	۷۶	۰,۹۵
مورد پنجم	۲۶۴	۶۶	۵۹	۰,۸۹۳۹
مورد ششم	۲۵۹	۳۵	۳۳	۰,۹۴۲۹
مورد هفتم	۲۷۶	۳۵	۳۰	۰,۸۵۷۱

میانگین کل زمان پاسخ، بطور کلی هنگام استفاده از چارچوب مدیر ابر کاهش می یابد، زیرا هنگام پیش بینی حرکت فقط برای شخصیت موجود در بازی، فقط پیش پردازش فریم بعدی مورد نیاز است و به پیش پردازش کل فریم ها احتیاجی نیست .

نتایج همانطور که انتظار می رفت، در موارد ۳ و ۴ میانگین زمان پاسخ طولانی تر از مورد ۲ مشاهده شد زیرا آن ها فقط وقتی به شرایط بازدیدهای متوالی و پی در پی می رسند، تازه شروع به پردازش می کنند برخلاف مورد ۲.

در مورد مقایسه ی موارد ۶و۷ نیز با مورد ۵ همین طور است. در موارد ۶ و ۷ میانگین زمان پاسخ طولانی تر از مورد ۵ مشاهده شد.

هنگام استفاده از چارچوب مدیر ابر بدون آموزش ،پیش پردازش تقریبا بلا استفاده است، زیرا الگوریتم پیش بینی حرکت هنگامی که بازیکن برای اولین بار از آن مکان عبور می کند ، اجرا نمی شود و بازیکن به ندرت از همان نقطه بار دیگر عبور کند و معمولا همان یکبار خواهد بود.

همانطور که در شکل۳ مشخص است، بهترین نتیجه برای میانگین زمان پاسخ سناریوی شماره پنجم است که با اعمال چارچوب ابری و اعمال پیش آموزش و پیش پردازش قبل از پیش بینی حرکت انجام شده است

شکل ۳_ میانگین زمان پاسخ گویی ۷ سناریوی مورد آزمایش

درصد بهبود زمان پاسخ همانطور که در نمودار ۲ قابل مشاهده است زمان پاسخ در مورد پنجم ۳۰,۹۸ درصد بهبود پیدا کرده است در مقایسه با مورد اول که بدون اعمال چارچوب پیشنهادی یعنی مدیر ابر است.

دقت پیش بینی حرکت از دقت متوسط ۰/۸۴۰۳ برخوردار بود.اینطور که از آزمایش ها برآورد شده، اگرچه در مورد ۷، زمان پاسخگویی طولانی تر بوده است و بهبودش نسبت به مورد ۶ که زمان کوتاه تری داشته، کمتر است اما پیش بینی می شود جیتر کم تر بوده و این یعنی کاربر از کیفیت تجربه ی بهتری بهره برده است .[۹]

دقت پیش بینی برابر است با : تعداد حرکات با پیش بینی درست / تعداد فریم های پیش پردازش شده

۴. ۱ نتیجه گیری

در این کار، ما یک روش ذخیره‌سازی لایه برای پخش بازی‌های مبتنی بر ابر را معرفی کردیم. این تکنیک شامل ذخیره‌سازی لایه‌های پس‌زمینه در حافظه نهان سرور برای اجتناب از رمزگذاری و انتقال مجدد آن از طریق شبکه است. در بهترین حالت، ما حدود ۲۳ ٪ کاهش اندازه جریان با کم‌تر از ۵ ٪ زمان رمزگذاری زمان به دست آوردیم. با این حال، پیشنهاد ما هنوز برای بازی با پس‌زمینه‌های متغیر مناسب نیست. در این حالت، ما هیچ کاهش اندازه جریان و بیش از ۱۵ درصد کاهش زمان کدگذاری را نداریم. دستاوردهای عملکرد به طور خاص در سیستم‌های ابر، ارایه خدمات برای تعداد بسیار زیادی از مشتریان، که در آن کاهش اندازه فردی برای مشتریان چندگانه وجود دارد، ضروری است. همچنین مهم است که اشاره کنیم که بازی‌های ۲ بعدی اغلب دارای چند تغییر پس‌زمینه هستند، که کاندیدهای طبیعی برای روش ما هستند.

۴. ۲ پیشنهادات

میتوان با استفاده از لایه مه[۲] کارایی روش پیشنهادی را به گونه ای بهبود بخشید که باعث افزایش کیفیت تجربه بخصوص در ژانرهایی از بازی رایانه ای که حساس به تاخیر هستند شود و همچنین باعث صرفه جویی در پهنای باند مصرفی زیرساخت ابر گردد.

۵. مراجع

[۱] M. Armbrust, A. Fox, R. Griffith, A.D. Joseph, R.H. Katz, A. Konwinski, G. Lee, D.A. Patterson, A. Rabkin, I. Stoica e M. Zaharia. ۲۰۱۰. A view of cloud computing. Communications of the ACM, ۵۰-۵۸.

[۲] M. Shiraz, A. Gani, R. H. Khokhar, R. Buyya.۲۰۱۳. A Review on Distributed Application Processing Frameworks in Smart Mobile Devices for Mobile Cloud Computing. IEEE Communications Surveys & Tutorials, ۲۹۴-۳۱۳

[۳] C. Huang, C. Hsu, Y. Chang, K. Chen. ۲۰۱۳. Gaminganywhere: An open cloud gaming system. In: Proceedings of ACM Multimedia Systems. p. ۳۶-۴۷., ۲,۶.

[۴] M. Luo, M. Claypool. ۲۰۱۵. Uniquitous: Implementation and evaluation of a cloud-based game game system in unity. Proc. of IEEE GEM.

[۵] Y. W.Bernier. ۲۰۰۱. Latency Compensating Methods in Client/Server In-game Protocol Design And Optimization. Game Developers Conference.

[۶] Y. Chang, P. Tseng, K. Chen, C. Lei. ۲۰۱۱. Understanding the Performance of Thin-client Gaming. Proceedings of IEEE CQR.

[۷] K. T. Chen, Y. C. Chang, P. H. Tseng, C. Y. Huang, C. L. Lei. ۲۰۱۱. Measuring the latency of cloud gaming systems. In Proceedings of ACM Multimedia.

[۸] M. Claypool, K. Claypool. ۲۰۰۶. Latency and Player Actions in Online Games. Communications of the ACM, vol. ۴۹, no. ۱۱.

[۹] D. M. Powers, ۲۰۱۱. Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. Journal of Machine Learning Technologies.

[۲] Fog