Софийски университет „Св. Климент Охридски”

Факултет по математика и информатика

Катедра „Софтуерни технологии”

ДИПЛОМНА РАБОТА

на тема

Изследване на методи за адаптиране на симулаторни видео игри спрямо кривата на учене на играча

Дипломант: **Иван Илков Найденов**
Специалност: **Софтуерни технологии**
Факултетен номер: **25393**

Научен ръководител:
проф. Д-р Боян Бончев

София, 2018 г.

Съдържание

1. Увод

1.1. Актуалност на проблема и мотивация

През последните десетилетия, с напредването и развитието на технологиите, напредна и развитието на видеоигрите, правейки ги по интересни, привлекателни и забавни за всички групи (възрастни, млади, мъже, жени и много други) хора. Именно заради това се е увеличил броят на хората интересуващи се от тези видеоигри. Повечето видеоигри ориентирани към играча (player-centric) [29], в зависимост от жанра, разчитат на няколко фактора които ще поддържат интересът му към играта, ще го забавляват и ще продължи да я играе. Някои от тези фактори са историята разказана в играта, предизвикателствата и целите представени в играта с които играчът трябва да се справи и постигне, дизайнът, визуалните и звукови ефекти в играта, различни нива на трудности за играчи с различни нива на умения нужни за да се справят с предизвикателствата, възможността да се играе в мултиплейър (групова игра)[30] и др.

Типът видеоигри на които ще обърнем внимание са симулаторните на физически апарат или процес които са едни от най-популярните както сред развлекателните така и сред сериозните игри. Симулаторните видеоигри, представляват игри опитващи се да подобават на дейности от истинският живот, като повечето се използват за трениране, анализ или предвиждане (пример за такива игри са SimCity 2000, Grand prix Simulator (1986), Touch Surgery и др.) [3]. Проблемът при повечето такива игри е че управлението на трудността и адаптирането на играта спрямо тази трудност става само в началото на играта, преди играчът да започне да играе на нея. Играчът, преди да започне с някои от нивата на самата игра, може да избере степен на трудността, според която се определя например с колко противника ще се сблъска и трябва да победи, какви предимства ще има изкуственият интелект (AI – artificial intelligence или така наречените виртуални (компютърни) играчи или NPC (Non-Player Characters) [30], герои/актьори които не са управлявани от играча) срещу който ще трябва да се изправи играча, как ще се променя локалната среда в играта или какъв праг на точки трябва да постигне за да осъществи целта на играта. Този начин с промяна на трудността и адаптирането на играта не е подходящ, ефективен и ефикасен за симулаторни игри, защото играчът не придобива нужният опит. Решението на този проблем е да променяме динамично трудността на играта, чрез адаптиране на характеристиките на игровия процес (като например вид, сложност и динамична трудност на задачите за управление) спрямо модела на самия играч [34] [35], включващ показаните до момента резултати. Класическият метод на адаптиране е базиран на задаване на нива на дадени метрики на играча (описващи резултата от играта) и промяна на игровия процес при достигането на тези нива [33]. Друг възможен подход би представлявало управлението на адаптивността на игровия процес при откриване на конкретен тип на крива на учене (например линейна, експоненциална и др.), описана от метриките за резултата и разпозната с анализ на шаблони [31] [32] [33]. За да бъдат изследвани предложените методите за адаптиране на видео игри от тип симулатор, ще се разработи игра от тип симулатор на управление на автомобил с три версии (първата версия няма да използва методи за адаптация, втората версия ще използва класическият метод за адаптиране и последната версия ще използва метода с откриване на крива на учене). За да се валидират тези методи, игрите ще бъдат дадени на случайни хора за провеждане на игрови сесии. Всеки направил подобна сесия ще бъде анкетиран, като това ще има за цел да определи до колко методите са били ефективни и ефикасни.

Тези методи на адаптация на игрите може да се използват за бъдещи разработки на симулаторни игри, подобрявайки не само времето за придобиването на нужният опит и

умения, които са целта на дадената симулаторна игра, но и подобрявайки креативността и знанията на играча който ги използва.

1.2. Цел и задачи на дипломната работа

Целта пред настоящата дипломна работа е да се извърши изследване на методите за адаптиране на 3D видеоигра от тип симулатор на управление на автомобил, а също така тя е разработена в рамките на научно-изследователския проект APOGEE (http://apogee.online), който планира да създаде решение, което от една страна да запълни липсата на проста, но ефективна софтуерна платформа с отворен код за лесно изграждане на образователни видеоигри от ИТ специалисти, а от друга страна – да отговори на нуждите от огромно количество специализирани адаптивни видеоигри за обучение по различни учебни дисциплини. Един от методите който ще изследваме е класическият метод на адаптиране, който е базиран върху задаване на граници на резултата постигнат от играча и промяна на игровият процес (като например промяна на гъстотата на мъглата, хлъзгавостта на терена (пътната настилка), усложняване на видимостта чрез намаляване на дневната светлина и др.) при достигане на тези граници. Друг метод, който ще бъде изследван, е откриване на крива на учене, описана от метриките за резултата и разпозната динамично (в реално време) чрез анализ на шаблони, спрямо която ще се извърши адаптация на динамичната сложност на играта по подходящо избрани характеристики на околната среда и за управление на автомобила. В допълнение към изследването на методите трябва да се валидира практически (чрез игрови сесии и анкетно проучване) ефективността и ефикасността на създадените методи за адаптиране на видео игри от тип симулатор на управление на автомобил.

Главните задачи, произлизащи от изследването на методите за адаптация на симулаторни видеоигри са следните:

- Да се анализират характеристиките на видеоигри от тип симулатор на управление на автомобил и възможностите за адаптирането на характеристики на игровия процес (като например динамична трудност на управлението на колата при различни атмосферни условия и осветеност на пътното платно) спрямо модела на играча.
- Да се извърши сравнителен анализ на технологиите и софтуерните платформи нужни за разработката на игри от тип симулатор. Тази задача е от значение тъй като играта трябва да върви бързо, да е лесна за схващане и управление, и да може да се играе както на компютър така и онлайн през браузър.
- Да се анализират и проектират три различни версии на онлайн 3D видеоигра от тип симулатор на управление на автомобил. Първата версия няма да има адаптивни компоненти променящи характеристиките на игровият процес. Втората версия на играта ще е с компонент за откриване на криви на учене на играча (т.е. криви на резултатни метрики) и адаптиране на характеристиките на игровия процес според тези криви. Третата версия на играта ще е с компонент за следене на резултата на играча спрямо който ще се адаптират характеристиките на игровия процес спрямо зададени нива до които може да достигне резултата.
- Да се извърши количествено изследване за определяне на възможните криви на учене на играчите. За намерените типове криви да се зададе адаптиране на динамичната сложност на играта по избрани характеристики за динамична трудност на управлението на колата при различни атмосферни условия и осветеност на пътното платно.

- Да се извърши количествено изследване за практическо валидиране (чрез игрови сесии и анкетно проучване) на ефективността и ефикасността на създадените методи за адаптиране на видео игри от тип симулатор на управление на автомобил.
- Да се анализират получените резултати и различията между подходите за адаптация: по криви на учене, по нива и без адаптация и направят изводи за приложимостта им при видео игри от тип симулатор на управление на автомобил.

1.3. Очаквани ползи от реализацията

В областта на видеоигрите както при тези за развлечение, така и при сериозните игри почти не се използва динамично адаптиране на трудността на играта. С прилагането на различните методи на адаптиране играчът може да изпита по добро преживяване докато играе. Също така би подобрило неговите умения и опит в променящата се среда спрямо това как се справя с поставените му задачи и препятствия.

Компонентът за адаптация спрямо откритите криви на учене в една игра [31] [32] [33] [34] [35], ще може да бъде лесно използван и внедрен във всяка една игра. Което от своя страна не само ще я подобри но и би станала по привлекателна и интересна за самите играчи, което би довело до повече хора които да си закупят играта следователно би донесло по-големи печалби за компанията разработила играта.

При сериозните игри които са ориентирани да помагат на ученици и студенти, една такава адаптация би била добър стимул да придобият нужните знания докато се забавляват. При симулатори, които имат за цел да тренират хората използващи ги за свършване на дадена работа по добре и успешно, такава адаптация би ги подготвила за всяка една възникнала ситуация. При хората с увреждане подобна игра симулация с динамично адаптиране би подобрило тяхното състояние и когнитивни умения [31] [32] [33] [34] [35].

1.4. Структура на дипломната работа

По своята структура, дипломната работа е разделена логически на отделни теми, които са оформени като глави.

Първата глава служи като въведение на проблема и мотивацията за изследването на методи за адаптиране на симулаторни видео игри спрямо кривата на учене. Описани са основната цел и произтичащите от нея задачи пред дипломния проект, както и очакваните ползи от реализацията му.

Втората глава е посветена на запознаване и преглед на предметната област. В нея се засягат основните подходи и методи при решаването на проблемите, свързани с реализацията на изследването и прилагането на методи за адаптация на симулаторни видео игри спрямо кривата на учене. Прави се кратко описание на симулатора който ще се използва за изследване на тези методи.

В трета глава се прави описание на съществуващите технологии, анализира необходимостта от използването на конкретни такива. За по-голяма прецизност технологиите се разделят по типове и се дава точно описание за всяка една от тях. Заедно с това се прави сравнителен анализ. На базата на получените резултати и анализа на изискванията се взема решение относно използването на конкретните технологии.

Четвъртата глава прави анализ на изискванията. Описват се основните процеси и роли. Същевременно се документират функционалните и нефункционални изисквания към системата. Детайлизира се необходимостта от интеграция с външни системи. Прави се подробна документация на симулатора.

В пета глава се прави описание на архитектурата на симулатора на управление на автомобила. Създават се съответно диаграми на поведението, както и декомпозицията на модули и под модули. Заедно с това графично се изобразява и описва моделът на данните, както и връзките между отделните елементи.

Шеста глава разглежда използваните конкретни методологии и подходи по време на имплементация. Описват се различни алгоритми и нуждата от тяхното използване. Прави се обосновка за техния избор. Съпоставят се и се анализират първоначалните цели и постигнатите резултати. Преглеждат се различните начини за тестване на системата: модулно, интеграционно, за производителност и лесна навигация. Прави си обобщение на резултатите от тестването.

В седма глава се съдържа обобщение на свършената работа, обсъждат се идеи за нейното развитие.

2. Преглед на предметната област на изследване на методи за адаптиране на симулаторни видео игри спрямо кривата на учене на играча

2.1. Същността на методите за адаптиране на симулаторни видео игри

В днешно време видеоигрите са станали голяма част от нашата култура и ежедневие, и са едни от най-популярните форми и начини за забавление. Те пристрастяват, вдъхновява и могат да държат вниманието ти с часове. Но игрите не са само за забавление, те също така се използват от психолози, преподаватели, учени и маркетинг специалисти за да подобрят начина на учене, да привлекат внимание, за анализиране поведението на мозъка. Такива игри които се използват не за забавление, се наричат сериозни игри (термин въведен от Sawyer и Rejeski) [3]. Чрез сериозните игри хората придобиват знания и умения, също така имат позитивно влияние в областите обвързани с когнитивни, мотивационни, емоционални и социални проблеми. А повечето сериозни игри са от тип симулаторни, поради простата причина че те имат възможността максимално да представят реалния свят чрез виртуалният такъв, който предоставят.

2.1.1. Механизъм на работа

Представеният метод за адаптиране, в дипломната работа, крива на учене (наричана още крива на опита или крива на производителността) предоставя графично изображение на напредъка на ученето за даден период от време. Във видео игрите, кривата на учене илюстрира как играчът прекарва време за овладяване на играта, чрез развиване на (игрови) умения, познавателни способности и знания за разрешаване и справяне с предизвикателствата на играта. Кривите на учене биват различни, както за всяка игра, така и за различните играчи. Кривата на учене представлява напредъкът в ученето и оттам специфичното представяне на отделния играч. Това означава, че е от голямо значение разпознаването на кривата на учене на играчът за реализацията на ефективен игров процес, защото това отразява мотивацията и ангажираността на играча и така тя може да се приложи за промяна на сложността на играта и нейните предизвикателства.

За лесното и автоматичното разпознаване на кривите на учене при играчите като поведенчески шаблони за постиженията на играча, в симулаторната игра за управление на автомобил, ще се използва софтуерният компонент „Player-centric rule-and-pattern-based adaptation asset“ [4] [15] [19] създаден от екип на ФМИ в рамките на европейския проект RAGE (http://rageproject.eu/). Интеграцията му в играта ще помогне за динамично откриване на специфични криви на учене на играча представляващи цялостното му представяне за определено време. Практическият експеримент включва разпознаване характеристиките на шаблон в индивидуалните криви на учене, като след това се дефинират в компонента и след това се продължи с разпознаването им динамично по време на играене на играта.

Чрез внедряването на този компонент в игра симулатор на управление на автомобил ще можем да анализираме и тестваме ползата от адаптации ориентирани към играча базирани на резултатите от играта. Целта на играта не е само да се използва като симулатор за каране на автомобил, но и за подобряване уменията на каране на кола на

играчите по ефективен, лесен и бърз начин. За това самият автомобил и околността в играта трябва да изглеждат възможно най реалистично и да представляват адекватно репрезентация на ситуации от реалният свят. Също така играта трябва да отчита прогреса или регреса на уменията за каране на играча и спрямо тях да предприеме нужните мерки за да подобри техните постижения.

2.1.2. Функционалности

Видеоигри подобни на симулатори за управление на автомобили се характеризират със следните функционалности и механизми за работа:

- Каране на различни видове коли, за постигане на максимална скорост, извършване на сложни маневри за справяне с препятствията по пътя. Това би създало незабравимо преживяване, мотивация и забавление за играча.
- Графичното представяне на околната среда трябва да е максимално близо с тази в реалният свят. (реалистичен дъжд, мъгла, път, при преминаване през подутина колата да подскача, появяване на дим при боксуване, разплискване на локва когато се премине през нея, реалистични звуци при смяна на предавки или сблъсък и много др.)
- Показване на текущата развита скорост по време на игра.
- Показване прогреса на обиколката, чрез индикатор, показващ колко остава до завършека на текущата обиколка.
- Възможността да се играе както с онлайн десктоп версия, така и онлайн през браузър. Играчите ще имат възможност да играят от всякъде и по всяко време. (Поддръжка на играта на различни платформи и операционни системи)
- Адаптиране на сложността на играта спрямо постигнатите резултати на играча. Адаптирането се състои в промяна характеристиките на околната среда, като промяна на видимостта чрез намаляване или увеличаване слънчевата светлина, промяна на гъстотата на мъглата, промяна хлъзгавостта на пътя и чрез промяна гъстотата и силата на дъжда.
- Записване на резултатите на сървър с цел изследването им и по нататъшното им използване с цел подобряване резултатите на играчите. Спрямо тези резултати ще могат да бъдат открити различните криви на учене и до колко адаптирането на трудността на играта спрямо тях е помогнало на играчите да се научат да карат автомобил по-добре.

2.1.3. Бизнес модел

Бизнес моделът на подобни симулатори на управление на кола се базира не само върху нуждата от забавление или обучение на играча, но и на извличане на печалба от продаването на играта или предлагането на микро-транзакции с цел подобряването на положението или състоянието на играча, ако не иска да влага много усилия за да постигне целите на играта [29]. Има много подобни симулаторни игри за управление на автомобили и по-голяма част от тях, които са високо качествени, са платени, а тези които не са платени, не успяват да предоставят нужните качества на дадени аспекти, като реално и атрактивно представяне, инструменти и приспособления за измерване на различни характеристики, подходящи предизвикателства, реалистични механики на автомобила и др. С цел маркетинг този род видеоигри предлагат демо версия която е непълна но достатъчна за да придобият хората представа за това какво предлага играта, за да могат да си я закупят ако придобият интерес към нея[29].

2.1.4. Предимства и недостатъци

Ето някои от предимствата и недостатъците при разработката на симулаторни видеоигре от този тип.

Предимства:

- По-малко стрес при ученето и повече забавление. Някои ситуации в реалният свят могат да бъдат доста стресиращи за младите и неопитни шофьори и могат да предизвикат нежелани реакции които до доведат до злополуки на пътя. Същите ситуации представени чрез симулаторна среда дават възможност на учащите се шофьори да разберат по-добре дадена ситуация и как да действат спрямо нея, в по спокойна обстановка, без да ги излага на опасност, докато в същото време се и забавляват.
- Справяне с реални ситуации. Повечето модерни симулатори могат да наподобят над 400 различни сценарии които обучават вариации от познавателни умения за справяне с натоварен трафик или тежки пътни условия. Това позволява на новите шофьори да подобрят времето си за реагиране. Симулаторите могат да бъдат използвани също за подготвяне на младите шофьори как да се справят в живото застрашаващи ситуации, които иначе биха били опасни за прилагане и упражнение в реални условия.
- Точно събиране на данни. В симулатора, са дадени точни данни, чрез които инструктора може да даде по точни инструкции какво да промениш и с колко до го промениш. Такава обратна връзка би била от голямо значение за учещия се, да се поправи бързо за секунди.
- Контрол над ситуацията. Симулаторите на управление на автомобил предлагат на учащите се доста предимства спрямо карането с истински автомобил. Едно такова предимство е управляемостта на дадена ситуация. Учещият се млад шофьор може да упражнява дадена маневра (паралелно паркиране, правилно изпреварване и пристрояване и др.) докато я научи как да я извършва правилно.

Недостатъци:

- Чувството да караш истинска кола не може да бъде симулирано. Фалшивото чувство за безопасност което се създава от симулатора дори може да навреди на младият шофьор. Някои учащи се шофьори могат да не приемат на сериозно обучението си в симулатора, вярвайки че като сгрешат и катастрофират, с натискането на бутон могат да рестартират положението си и да продължат със симулацията. Истинските опасности и последствия от действията на шофьора не се случват в симулаторите на управление на автомобил, което от своя страна развива у тях фалшива представа за безопасност, отговорност и компетентност.
- Докато симулаторите на управление на кола предоставят уникални предимства пред карането на истинска кола, няма много проведени научни изследвания показващи дали придобитите знания от симулатора се прехвърлят при каране р реални условия с истинска кола.
- За да се използва пълният потенциал на симулатора за каране на кола, трябва да се създаде 360 градусово преживяване (да се създаде 3D виртуална реалност абсолютно наподобяваща реалният свят) за да може шофьорът да се потопи в преживяването. Слагането на три монитора за изобразяване на реалността не са подходящи за ученето на техники като поглеждането назад за да се види дали има коли в сляпото петно на шофьора. По напреднала система би струвала много повече и би заела по голямо място, докато пътищата предоставят нужното място за упражнение на каране на автомобил.

- За по възрастните и опитни шофьори симулатора представлява дискомфорт. То може да им причини замаяност докато изпитват някои от визуалните и звукови ефекти на симулатора. [16]

2.1.5. Проблеми

Съществуват ред предизвикателства и проблеми, пред които се изправят подобни видеоигри по време и след тяхното създаване.

- Разработването на дизайна на виртуална среда която е максимално близко с реалния свят. Това включва имплементирането на реални звукови и визуални ефекти (звукът на мотора, при смяна на скоростите, при катастрофа или боксуване на гуми, появата на ди от ауспуха на виртуалната кола, отраженията на пътя при дъжд, заслепяване от светлините от насрещно движение, шума от дъжда и др.).
- Разработването на реално поведение на колата в самият симулатор.
- Събирането и анализирането на коректни данни от проведени практически тестове.
- Разработването на един такъв симулатор може да бъде много скъпо не само спрямо използваният софтуер но и хардуерът който ще е нужен за да се подкара самият симулатор.

2.2. Съществуващи решения за симулатори на управление на превозно средство

По време на разработката на тази дипломна работа ще бъдат разгледани и сравнени различни симулатори за управление на превозно средство. [1]

2.2.1. rFactor 2

rFactor 2 е реалистична, лесно разширяема състезателна симулация разработена от Studio 397. Предлага най-новите автомобили, страхотна графика, групова игра (multiplayer), и реалистично състезаване. rFactor 2 се отличава със смесена състезателна класация със върх реалистична динамика, завладяваща звукова среда и зашеметяваща графика, идеална както за групова игра, така и за самостоятелна.

Може да се прави състезание с изкуствен интелект или с други играчи. Поддържа пълен цикъл ден-нощ, както и динамично променящи се метеорологични условия, като дъжд, който динамично се вгражда в локви. Трасетата разполагат с "реална пътна" технология, която променя сцеплението, при преминаване на много коли по тях. Всичко може да бъде персонализирано от общността, включително добавяне на нови трасета, за провеждане на състезания, както и нови автомобили.

rFactor 2 се използва за много състезателни събития провеждани по състезания. Също така се използва от ветерани от формула едно за трениране. [5] [6]

Фиг. 1 – играта rFactor2

2.2.2. Assetto Corsa Competizione

Assetto Corsa Competizione е новата официална видео игра на Blancpain GT Series. Благодарение на изключителното качество на симулацията, играта ще позволи, на играча, да изпита истинската атмосфера на шампионата на GT3, като се състезава срещу официални шофьори, отбори и коли, възпроизведени по време на игра с най-високото ниво на точност.

Assetto Corsa Competizione е създаден, за да пресъздаде шофирането на истински автомобили от серията Blancpain GT Series чрез сложен математически модел, който точно възпроизвежда сцеплението на гумите, въздействието на аеродинамичността, параметрите на двигателя, окачването и електронните системи, които определят баланса на автомобила, както и влиянието на механичните повреди върху управляемостта на автомобила.

Unreal Engine 4 гарантира фотореалистично изобразяване и точно представяне на сценарии, автомобилни материали и климатични условия.

Благодарение на технологията Laserscan, всяко официално трасе се пресъздава с максимална точност: всички бордюри и детайли напълно се вписват в истинските си места, за да дадат възможно най-вероятен геймърски опит.

Добре структурирана система за класиране ще оценява индивидуалното представяне и поведението при шофиране, за да възнагради най-добродетелните шофьори и да насърчи честната игра в онлайн състезанията.

Режимите на кариера, шампионат и безплатна игра предлагат пълно и адаптивно игрално изживяване, също благодарение на уроците и прогресивните нива на помощ, които позволяват да се приспособят трудностите на противника на AI и контрола на стабилността на автомобила спрямо способностите на играча. [7] [8]

Фиг. 2 – играта Assetto Corsa Competizione

2.2.3. iRacing

iRacing е водеща онлайн състезателна симулация. Разработена от самото начало като централизирана състезателна услуга, iRacing организира и хоства състезания по виртуални трасета представящи реални такива по целия свят. iRacing използват най-новите технологии за пресъздаване на все по-разширяваща се гама от прочути състезателни автомобили и писти. Всички детайлни подробности на автомобили и писти, са практически неразличими от истинското - давайки на състезателите несравнимо потапяне, когато те вземат зеленото знаме в онлайн състезанията които предлага iRacing.

iRacing се използва не само от професионални шофьори, но и от случайни геймъри. Въпреки че iRacing е състезателен симулатор в центъра на своята същност, стойността като инструмент за обучение не може да засенчи вълнуващия състезателен опит, който очаква геймърите, търсещи опит в шофирането. Всичко нужно за да започне един играч е компютър, контролер и добра връзка с интернета. Също така този симулатор предлага открита практика, квалификация, тестване, състезание за време. Предлага над 80 трасета за състезание и също толкова и коли от които играчът може да избира. [11]

Фиг. 3 – играта iRacing

2.2.4. Euro Truck Simulator 2

Euro Truck Simulator 2 дава възможност на играча да кара камиони по пътищата на Европа, доставяйки товари на впечатляващи разстояния. С десетки градове, които да бъдат изучени от играча като Великобритания, Белгия, Германия, Италия, Холандия, Полша и много други, издръжливостта, уменията и бързината ще бъдат тласнати до краен предел. Играча може да стартира свой собствен бизнес, който продължава да расте, дори след като играчът е приключил с доставките на стоки. Играчът има възможност да изгради свой собствен автопарк от камиони, да купува гаражи, наема шофьори за да управлява неговата компания за да постигне максимална печалба. Също така му е дадена възможността да персонализира своя камион. Хиляди километри от реални пътни мрежи със стотици известни забележителности и постройки е друго което предлага симулатора. Представяйки множество камиони с безброй възможности за персонализиране и усъвършенствана физика на движението, играта осигурява несравнимо шофиране, което я поставя на мястото на най-популярния тренажори за шофиране на камиони на пазара. В симулатора са вградени правилата и законите по пътищата, които ако не се спазват от играча бива наказван като му се отнема от виртуалните пари представени в играта. [12] [13] [14]

Фиг. 4 – играта Euro Truck Simulator 2

2.2.5. X-Plane 11

X-Plane 11 е симулатор за управление на самолети. Симулатора предлага реалистични модели на самолети, които са внимателно изобразени и до най малкият детайл, от предавателните кутии до нивите. Всяко въздухоплавателно средство предложено в симулатора е с красива, използваема 3-D пилотска кабина, като също така всяка такава кабина е използваема за инструментален полет. Играчът може да гледа как други самолети получават обслужване или може да поиска обслужване за неговия собствен самолет. X-Plane 11 включва детайлна триизмерна сцена за повече от 3000 летища по целия свят. Терминали, хангари, писти и много други правят тези летища да се чувстват като истински такива. Също така симулаторът дава възможност за контрол над времето което предоставя възможността на играча да лети в различни метеорологически условия. [9] [10]

Фиг. 4 – играта X-Plane 11

2.3. Сравнителен анализ на съществуващите решения симулатори на управление на превозно средство

Съществуващите решения ще сравним като използваме следните критерии за оценка:

- Цена – може би един от най-важните критерии, тъй като се отнася до всички потребители. Повечето симулатори имат демоверсии които могат да се тестват за да преценят хората дали да си закупят пълната версия.
- Поддръжка на различни платформи. Може ли симулатора да се игра на операционни системи като Windows, Mac, Unix, Android, iOS или на конзолна платформа (като например PS, Xbox, Nintendo и др.).
- Адаптивност на дизайна. Дали симулаторът поддържа работа с различни по размер на екрана устройства (поддръжка на различни резолюции).
- Симулатора с отворен код ли е? Това означава дали е дадена възможността на потребителите да пригодят и модифицират симулатора според собствените си нужди или да правят подобрения които ще са общодостъпни за всички за да могат да бъдат използвани.
- Поддържани режими на играта (сингъл-плейър, мулти-плейър).
- Системни изисквания. Какви са минималните и максималните системни изисквания за симулатора да върви на платформите на които се поддържа.
- Нужда от специален хардуер. Повечето симулатори изискват специални устройства (джойстик, кормилна уредба, контролер и др.) за да могат да бъдат използвани, възможно е и само с клавиатура и мишка да се играе.

Симулатор *Критерий*	**rFactor 2**	**Assetto Corsa Competizione**	**iRacing**	**Euro Truck Simulator 2**	**X-Plane 11**
Цена	29,99€ еднократно заплащане	24,99€ еднократно заплащане	13$ месечно, 33$ за 3 месеца, 66$ за 1 година или 199$ за 2 години	19,99€ еднократно заплащане	59,99€ еднократно заплащане
Демо версия	Да	Не	Не	Да	Да
Поддръжка на ОС	Windows 7 and later versions, поддръжка на VR(HTC Vive, Oculus Rift и Windows смесена реалност)	Windows 7 and later versions	Windows 7 and later versions, поддръжка на VR(HTC Vive, Oculus Rift) (Уеб-базирана)	Windows 7 and later versions, Mac OS X 10.9 (Mavericks) и Mac OS X 10.10 (Yosemite), Linux Ubuntu 12.04	OS X: OS X 10.10 or newer (e.g. Yosemite, El Capitan, or Sierra), Windows: Windows 7, 8 или 10, Linux:Varies
Адаптивност на дизайн	Да	Да	Да	Да	Да
Open Source	Да	Не	Не	Да	Да
Single player	Да	Да	Да	Да	Да
Multi player	Да	Да	Да	Не	Не
Минимални Системни изисквания	Процесор: 2.8 GHz Intel Core 2 Duo or 3.0 GHz AMD Athlon II x2 Памет: 4 GB памет Видеокарта: nVidia GTS 450 or AMD Radeon 5750 DirectX: версия 9.0c Мрежа: Широколентова интернет връзка Пространство за съхранение: 30 GB достъпно	Процесор: Intel Core i5-4460 or AMD FX-8120 Памет: 4 GB памет Видеокарта: GeForce GTX 460 2GB, Radeon HD 7770 DirectX: версия 11 Пространство за съхранение: 50 GB достъпно пространство Звукова карта: Integrated	Intel Core i3, i5, i7 or better or AMD Bulldozer or better 8 GB of RAM NVidia GeForce 2xx series or better, 1GB+ dedicated video memory, AMD 5xxx series or better, 1GB+ dedicated video memory or better Integrated Intel HD Graphics 4200+ or better, with 8GB of system RAM 10 GB of free disk space Microphone optional, required for voice chat Controller: Steering wheel, analog gamepad, joystick, mouse, or any version of Windows supporting touch screen driving.	Processor: Dual core CPU 2.4 GHz Memory: 4 GB RAM Graphics: GeForce GTS 450-class (Intel HD 4000) Hard Drive: 3 GB available space	Процесор: Intel Core i3, i5, or i7 CPU with 2 or more cores, or AMD equivalent Памет: 8 GB памет Видеокарта: DirectX 11-capable video card from NVIDIA or AMD w/512 MB VRAM DirectX: версия 11 Пространство за съхранение: 20 GB достъпно пространство

	пространство Звукова карта: DirectX Compatible		Internet Browser Requirements: Firefox 15, Internet Explorer 8, Google Chrome 20 or newer JavaScript enabled Cookies enabled Flash Player Internet Connection Speed: DSL, Cable, Fiber, 128K or faster – Supported. Satellite is not supported.		Звукова карта: Default
Максимални системни изисквания	Процесор: 3.0 GHz Intel i5 or 4.0 GHz AMD FX Памет: 8 GB памет Видеокарта: nVidia GTX 760 or AMD 7870 DirectX: версия 11 Мрежа: Широколентова интернет връзка Пространство за съхранение: 60 GB достъпно пространство Звукова карта: DirectX Compatible	Процесор: Intel Core i5-8600K or AMD Ryzen 5 2600X Памет: 16 GB памет Видеокарта: GeForce GTX 1070 8 GB, Radeon RX 580 8GB DirectX: версия 11 Пространство за съхранение: 50 GB достъпно пространство Звукова карта: Integrated	Няма посочени	Процесор: Quad core CPU 3.0 GHz Памет: 6 GB RAM Графика: GeForce GTX 760-клас (2 GB) Твърд диск: 3 GB свободно място	Процесор: Intel Core i5 6600K at 3.5 GHz or faster Памет: 16 GB памет Видеокарта: DirectX 12-capable video card from NVIDIA or AMD w/4 GB VRAM DirectX: версия 12 Мрежа: Широколентова интернет връзка Пространство за съхранение: 65 GB достъпно пространство Звукова карта: Default
Нужда от специален хардуер	Поддръжка за контролер, Поддръжка за аксесоари за виртуална реалност	Поддръжка за контролер	Поддръжка за контролер, Поддръжка за волан, Поддръжка за аксесоари за виртуална реалност	Поддръжка за контролер,	Поддръжка за контролер, Поддръжка за аксесоари за виртуална реалност

Фиг. 5 – сравнителен анализ на симулаторни решения за управление на превозно средство

От така направеният анализ, представен в Фиг. 5 може да се установи, че всички разгледани симулатори разполагат със сходни характеристики по отношение на поддръжката на различните операционни системи, адаптивността на дизайна към различните размери на екрана устройства, с които могат да ги достъпват. Сходство има също така в системните изисквания които се стремят да се използва най-новия хардуер съществуващ на пазара. Друго сходство е интегрирането на различни аксесоари даващи възможността да се играе а тези симулатори по различни начини, удобни за играча било то с мишка и клавиатура, или с контролер, или с волан. Но нито един от тези симулатори не използва методите за динамично адаптиране на сложността спрямо резултатите на играча, което е и предмет на разработка на дипломната работа. Ето защо е нужно провеждането на изследването на тези методи и прилагането им върху симулаторната игра разработена по време на дипломната разработка.

2.4. Изводи

От така направения преглед на съществуващи решения става ясно, че при разработването на симулатора на управление на автомобил трябва да се наблегне на някои функционалности и механизми: играта да поддържа всички възможни операционни системи, да предлага възможността да се играе с различни резолюции, да предоставя възможност да се играе, както на по нови системи, така и на по стари такива. Също така да се интегрират методите за адаптиране на играта, както и възможността да се играе чрез различни устройства. От голямо значение ще е играта да е общодостъпна за всички, без да се заплаща за нейното използване.

3. Избор на технологии и платформи за разработка

3.1. Изисквания към средствата за реализация и предварителна оценка

Предварителен анализ и оценка на функционалните и нефункционалните изисквания към симулатора за управление на автомобил показаха необходимостта от използването на технологии които ще позволят разработване на реалистична виртуално околна среда, както и реализъм спрямо доста аспекти свързани с колата, как ще се държи в различни условия на пътя или при сблъсък с друг обект. Друг фактор на който трябва да се наблегне е изборът на подходящи технологии позволяващи симулаторът да се играе на различни операционни системи и платформи, както и да поддържа различни резолюции като за стари системи така и за по нови такива. Потребителският интерфейс трябва да е интуитивен и възможно най много опростен. Още от водещите изисквания към набора от технологии са бързодействието, надеждността, осигуряването на бърза разработка и лесната разширяемост. Други важни изисквания за избора на технологии са възможността за бързо изпращане на резултатите, от симулатора докато играчът го използва, към подходящ сървър където да бъдат обработени и записвани. Друг критерий за избор на технологии е методите за адаптиране на сложността на симулатора. При избора на технологии е важно и тяхната популярност, която да гарантира тяхната бъдеща поддръжка и лесна интеграция с други популярни технологии и компоненти.

Предварителен анализ на технологиите и тяхната евентуална реализация включва следните под категории:

- Платформа за разработка на симулатора – основната задача е да се избере подходяща платформа за имплементиране на симулатора и различните компоненти които го изграждат. Изборът на платформа също трябва да се съобрази с технологиите които ще се изберат за външните компоненти взаимодействащи със симулатора. Друга характеристика по която да се избере платформа ще е предоставянето на готови компоненти които ще могат бързо и лесно да се имплементират и интегрират. Друг основен фактор е езиците за програмиране които поддържа платформата. И не на последно място, дали платформата поддържа технология позволяваща направените игри и приложения на нея да се използват на различни операционни системи и платформи.
- Технология за изпращане на данни от симулатора до даден сървър – Поради причината, че трябва постоянно да се изпращат резултатите по време на играене на симулатора, трябва да се избере технология която да поддържа връзка към даден сървър, към който да се изпращат и записват данните за всеки един играч, в отделни файлове.
- Методи за адаптиране трудността в симулатора – както се установи по време на анализа на съществуващите решения в разглежданата проблемна област, ще има различни методи за адаптиране на симулатора за управление на автомобил. Поради тази причина ще се направят различни версии на симулатора, като всяка версия ще използва различен метод за адаптация. Това ще е и единствената разлика между версиите на симулатора, всичко останало като имплементация ще е едно и също. Целта е да можем да тестваме всяка

една версия и след това въз основа на резултатите от тестовете да преценим кои от методите е най-удачен за промяна на сложността при симулаторите.

3.2. Избор на платформа за разработка на симулатора на управление на автомобил

От направеният анализ на нуждите от избор на технологии и платформи и по специално частта с избор за платформа за разработка, че ще бъде сведена до избор до конкретен програмен език за разработка който се използва от платформите. Неговият избор е продиктуван, както от съществуващи познания в областта, а също така и от предишен опит в тяхното използване.

Използваният език ще бъде C# - това е обектно-ориентиран език за програмиране, разработен от Microsoft, като част от .Net, с общо предназначение. За по голямо улеснение при разработката на симулатора трябва да се направи избор за конкретна платформа която го използва.

3.2.1. Налични платформи

Constructor (Scirra, Studio 414 The LightBulb**)** – Constructor е платформа за разработване на уеб базирани приложения. Използва HTML5 и JavaScript като езици за програмиране. HTML5 е сравнимо със Adobe's Flash технологията която доминира интернет игрите. За разлика от Flash обаче не е нужно да се инсталира допълнителен софтуер за играеш игра направена на HTML5, тъй като е вграден в браузъра. Поради тази причина игрите направени на Constructor могат да вървят и на мобилни устройства. JavaScript се използва за подобряване бързината на играта и за писане на допълнения (plugins) към играта които могат да бъдат използвани и от други хора. Също така Constructor предлага offline режим на играене използвайки HTML5 AppCache, позволявайки на игрите да се играе дори когато няма достъп интернет. [20]

GameMaker: Studio (YoYo Games 2007) - GameMaker: Studio е платформа за разработване както на 2D уеб базирани игри, така и на 2D десктоп и конзолни игри. Платформата използва вграден език за програмиране GML, както и 'Drag and Drop' програмиране. Платформата предлага внедряване на играта на различни операционни системи, конзоли и платформи (Windows, Mac, Ubuntu, Android, iOS, HTML5, PlayStation 4, Xbox one и др.). Предлага голям набор от помощни инструменти за персонализиране на почти всички аспекти в играта. Също така дава възможност за разработване на собствени модули както и използвани и на чужди такива. [21]

Unity (Unity Technologies) – Unity е платформа за разработка на 2D и 3D игри, които могат да бъдат уеб базирани, десктоп или конзолни. Основни езици които са интегрирани в платформата и могат да бъдат използвани за програмиране на игри са C# и JavaScript. Освен че предлага внедряване на игрите за почти всички операционни системи и платформи, също така предлага набор от мощен инструментариум за създаване на двуизмерни и триизмерни обекти, които да бъдат използвани в играта, за разработване поведението на тези обекти, за разработване на отлични графични среди. Има съвместимост със сорс контрол технологии които помагат за управлението на разработката на играта и лесното и споделяне между хора работещи колективно за разработването на дадена игра. Възможност за програмиране на игри за Виртуална реалност с новият вграден Vuforia Engine 7.5. Също така компанията е направила магазин за добавки (plugins/assets), достъпни за всички използващи тази платформа. Други технологии които се предлагат е възможността игрите да са мулти-плейър, а също и сингъл-плейър.[22]

Unreal Engine (Epic Games, Inc. 2004) - Unreal Engine е платформа за разработка на 2D и 3D игри, които могат да бъдат уеб базирани, десктоп или конзолни. Езиците

вградени в тази платформа са C++ и BluePrints Visual Scripting. Това позволява максимално представяне на игрите направени на тази платформа. Поддържа внедряване на почти всички операционни системи и платформи. Също така предлага инструменти за разработка и модифициране на графични обекти. [23]

3.2.2. Сравнителен анализ

Целта на сравнителния анализ е да се даде оценка и аргументация при избора на платформа за разработване на симулатора. За да бъде възможна аргументация, първо трябва да се заложат основните критерии, от които ще се определи използваната платформа. Ще направим сравнителен анализ по следните критерии:

- Цена за използване на платформата.
- Възможност за безплатно използване.
- Вградени програмни езици в платформата.
- Тип игри които могат да се разработят чрез тази платформа (2D, 3D)
- Поддържани платформи за внедряване (publishing a game).
- Възможност за разработване на собствени модули.
- Съвместимост с външни инструменти (Tools).
- Системни изисквания.

Платформи Критерий	**Constructor**	**GameMaker: Studio**	**Unity**	**Unreal Engine**
Цена	Цени започващи от 89,99€	Цени започващи от 39,99€	Цени започващи от 30,00€	Безплатен за ползване, и 5% печалбата ако тя надхвърля 3000$ на четиримесечие
Безплатна версия	Временно ползване	Временно ползване	Да	Да
Вградени езици	HTML5, JavaScript	GML	C#, JavaScript	C++
Типове поддържани игри	2D, уеб базирани	2D, Отчасти 3D, Онлайн и офлайн десктоп игри	2D,3D, уеб базирани, десктоп игри конзолни игри	2D,3D, уеб базирани, десктоп игри конзолни игри
Поддръжка платформи за внедряване (publishing)	Windows, Mac, Linux, iOS, Android, Windows phone, Wii U	Windows, Mac, Ubuntu, PS 4, Xbox One, iOS, Android, Windows phone, Tizen	Windows, Mac, Ubuntu, Linux, PS 4, Xbox One, iOS, Android, Facebook	Windows, Mac, Ubuntu, Linux, PS 4, Xbox One, iOS, Steam, HTML5, Nintendo switch, PlayStation VR, Oculus Rift VR, VIVEPORT VR, Daydream
Възможност за разработване на собствени модули	Да	Да	Да	Да

Съвместимост с външни тулове	Да	Да	Да	Да
Single player	Да	Да	Да	Да
Multi player	Да	Да	Да	Не
Системни изисквания	CPU: Quad Core CPU or Dual Core CPU (Intel Core 2.8 GHz, AMD Athlon 64 X2 4400+ or faster) CPU SPEED: Info RAM: 8 GB OS: Windows 7, 8, 10 VIDEO CARD: DirectX 9 Compatible - Nvidia Geforce 8800GT / ATI Radeon 4850 or faster with Shader Model 3 and 512 MB VRAM FREE DISK SPACE: 10 GB	CPU: Info CPU SPEED: 64bit Intel compatible Quad Core CPU RAM: 8 GB OS: Microsoft 64bit Windows 10 VIDEO CARD: DX11 based graphics card FREE DISK SPACE: 3 GB	CPU: SSE2 instruction set support. GPU: Graphics card with DX10 (shader model 4.0) capabilities. The rest mostly depends on the complexity of your projects. Additional platform development requirements: iOS: Mac computer running minimum macOS 10.12.6 and Xcode 9.0 or higher. Android: Android SDK and Java Development Kit (JDK); IL2CPP scripting backend requires Android NDK. Universal Windows Platform: Windows 10 (64-bit), Visual Studio 2015 with C++ Tools component or later and Windows 10 SDK	Processor Quad-core Intel or AMD, 2.5 GHz or faster Memory 8 GB RAM Video Card/DirectX Version DirectX 11 compatible graphics card

Фиг. 6 – сравнителен анализ на съществуващите платформи за разработване на игри

3.2.3. Заключение и обосновка

На пръв поглед изглежда че Unreal Engine е най подходящата платформа за разработване на игри. Има добре описана документация и помощни материали под формата на видеа за обучение и упражнение. Платформата е безплатна стига някоя от игрите която е създал даден човек не започне да печели повече от 3000$ на всеки четири месеца, в такъв случай Epic Games взима 5% от печалбата. Предлага и повече платформи на които е възможно да се внедри играта.

На втори поглед обаче Unreal Engine поддържа C++ като език за програмиране, което е в противоречие с изискването на дипломната работа симулаторната игра за управление

на автомобил да бъде разработена на C#. Такова решение е предложено от Unity платформата. Някои от недостатъците на тази платформа спрямо Unreal Engine са поддържаните платформи за внедряване на играта и ограничената безплатна версия .която се предлага от Unity. Като изключим малките недостатъци на Unity, тази платформа притежава всичко необходимо, за да бъде предпочитания избор за платформа на разработване на симулатора.

3.3. Избор на технология за изпращане на данните от играта до даден сървър.

Спрямо ограниченията да използваме C# разработване на симулатора и избора а използваме Unity платформата има няколко технологични решения за изпращане на данни от играта до даден сървър, които отговарят на тях. Други ограничения на които трябва да отговарят тези технологии е да не забавят играта, да могат лесно да се интегрират към играта и да са надеждни.

3.3.1. Налични технологии

Named Pipes (Named Pipes for interprocess communication) - Те предлагат повече функционалност, отколкото анонимните тръби, които осигуряват комуникацията между различни процеси на локален компютър. Наименованите тръби поддържат пълна дуплексна комуникация по мрежата и множество сървъри, комуникация на базата на съобщения и подправяне на клиенти, което позволява свързването на процесите да използват свои собствени правила на отдалечени сървъри. Тази технология обикновенно се използва заедно с нишковото програмиране. Особени характеристики на тази технология е че трябва да се разработи и поддържа клиентска и сървърна част които да си комуникират. [24]

Sockets (които комуникират чрез TCP/IP-based network**)** - В днешно време се използват все повече и повече. Те осигуряват лесен начин за обмен на данни по мрежата. Може да се използва като обмен на съобщения между потребителите, да се прехвърлят файлове ,да се играе "разпределени" игри или за комуникация с множество програми. Благодарение на мощните си функции, socket се превръщат в необходимост да се научи технология за разработчиците.

Сокетите се базират на архитектура на клиент / сървър. Накратко сървърът ще запази номер на порт. След това ще се ослуша за клиенти които искат да се вържат към него. Когато клиент се опита да се върже със сървърният сокет и връзката бъде успешна, ще могат да си обменят данни. Когато свършат, връзката ще бъде затворена. Но за до може данните да бъдат изпратени и получени, те ще трябва да се сериализират и десериализират. [25]

Уеб услуги (Web Service) – Тази технология разчита на това, че на даден сървър ще се изпращат данните под формата на заявка, която ще бъде обработена от уеб услугата. Уеб услугата представлява PHP код който се изпълнява ат страна на сървъра и изобщо не затормозява клиентската част. Единственото изискване към клиента е да направи коректна заявка към уеб услугата намираща се на дадения сървър. [26]

3.3.2. Сравнителен анализ

За да се състави обективна преценка и аргументация относно избор на технология за изпращане на данни от играта до даден сървър ще бъдат разгледани следните критерии:

- Компоненти за имплементиране
- Използване допълнителни технологии
- Възможни ефекти от използвани външни технологии

Технологии Критерий	**Named Pipes for interprocess communication**	**Sockets**	**Web Service**
Компоненти за имплементиране	Клиентска част която да изпраща съобщения, Сървърна част която да приема съобщения	Клиентска част която да изпраща съобщения, Сървърна част която да приема съобщения	Сървърна част която да обработва съобщенията, Клиентска част изпращаща съобщения
Използване допълнителни технологии	Използване на нишки(threads)	Използване сериализация на данните	PHP файл съдържащ логиката за обработка на данни
Възможни ефекти от използвани външни технологии	Възможността да се получи deadlock ситуация при нишките, което ще забави играта	Сериализацията и десериализацията може да забави данните	Ако сървъра падне, уеб услугата няма да е достъпна, но това няма да спре действието на играта нито ще я забави.

Фиг. 7 – сравнителен анализ на съществуващите технологии за пращане на данни от играта към даден сървър

3.3.3. Заключение и обосновка

На пръв поглед Named Pipes for interprocess communication и Sockets технологиите не изглеждат добро решение за изпращане и получаване на данни между играта и сървъра, тъй като има възможност това да прекъсне или забави процеса на играта. Друго което може да се забележи е, колко по трудни са за имплементиране и интегриране от колкото използването на Web Service. Дори Web Service-те като технология не представляват проблем за играта дори и да не работят.

На базата на направеният анализ и различните разгледани критерии, Уеб услугите са предпочитаната технология за използване комуникация между играта и сървъра.

3.4. Избор на методи за адаптиране на сложността в симулатора.

Изборът на методи за адаптиране на сложността в симулатора за управление на автомобил, както и при вече разгледаните други технологии за платформа за разработка на симулатора и за комуникацията между от играта и сървъра се определя от различни критерии, фактори и ограничения. Методът не трябва да забавя процеса на играта и трябва да е лесен за интегриране. Също така методите които ще се изберат за различните версии на симулатора трябва да отговарят на условията за провеждане на изследването на адаптация на симулатор спрямо кривата на учене.

3.4.1. Налични технологии

Първи метод – промяна на трудността от играча в началото на играта. Преди да започне играта самият играч може да настрои на каква трудност би искал да играе, спрямо която се настройват различните характеристики определящи трудността. Много компании разработващи игри използват този метод за адаптиране сложността на играта. По този начин трудността на играта е една и съща през цялото време докато не бъде сменена от самия играч и не предлага друга адаптация на сложността.

Втори метод – Без промяна на трудността. Някои игри предлагат различни нива като всяко ниво е по трудно от следващото. Играчът няма възможност да променя трудността на играта.

Трети метод – Промяната на трудността ще зависи от показаните резултати по време на играене. Спрямо резултатите по зададени нива, които резултата може да достигне, ще се определя каква адаптация ще се приложи в играта.

Четвърти метод – Промяната сложността се извършва по следният начин: първо се открива крива на учене, описана от метриките за резултата и разпозната динамично (в реално време) чрез анализ на шаблони. След което спрямо тази крива на учене ще се извърши адаптация на динамичната сложност на играта по подходящо избрани характеристики.

3.4.2. Сравнителен анализ

Избирането на подходящи методи за адаптиране на сложността в симулатора ще се избира по няколко критерии. Един от тези критерии е начинът, предлаган от методите, за адаптиране на сложността. Ограниченията поставени от разработката на дипломната работа са друг критерии. Също така изборът се ограничава и от избраните предишни технологии за избор на платформа и метод за комуникация между играта и сървъра.

Методи ***Критерий***	**Метод 1**	**Метод 2**	**Метод 3**	**Метод 4**
Начин на адаптиране	В самото начало на играта, правено от самият играч (Статична адаптация)	Не се прави адаптация на сложността, тя е вградена в играта и не може да се променя	Адаптация на сложността динамично по време на играта спрямо резултатите на играча	Адаптация на сложността динамично по време на играта спрямо резултатите на играча
Статично адаптиране на сложността	Да	Да	Не	Не
Отговаря ли на ограниченията на дипломната работа	Не	Да	Да	Да

Фиг. 8 – сравнителен анализ на съществуващите методи за адаптация на сложността на симулатора

3.4.3. Заключение и обосновка

След представения анализ и схема (Фиг. 8) на различните решения за избор на метод за адаптация на сложността на симулатора се стигна до следното заключение, че метод 1 не е подходящ за целите на изследването което се провежда с тази дипломна работа, тъй като ще е много сложно следенето и анализирането на резултатите от играча. Метод 2 може да се използва като основа за базово сравнение спрямо останалите модели, тъй като той не предлага адаптация на сложността на играта. Чрез него може да се съберат резултати, които да сравним с резултати от друг метод, за да се направи оценка за това дали даденият метод е подходящ за адаптация на сложността на видеоигра. Метод 3 и 4 са подходящи методи за динамично адаптиране на сложността на видеоигри.

4. Анализ на изследване на методи за адаптиране на симулаторни видео игри спрямо кривата на учене на играча

Извличането и анализа на изискванията е един от най-важните етапи от разработката на софтуер. Това е моментът, когато бизнес логиката и технологията трябва да се допрат и „разберат“.

4.1. Основни концепции

Основната концепция в изследването са методите за адаптиране на симулаторни видеоигри тяхното приложение и разработване в самите видеоигри. Методите трябва да предлагат динамично адаптиране на сложността, като промяната ѝ се извършва в реално време (по време на използване на играта). Единствено изключение прави базовият метод, който ще се използва за сравнение с другите методи, и при който няма нужда да има адаптация на трудността. Методите трябва да предлагат интуитивен начин за анализиране на резултата на играча, според който ще се адаптира сложността на видеоиграта.

4.2. Функционални изисквания

Една от двете основни цели на анализа и извличането на изисквания е дефиниране на функционалните изисквания към симулатора на управление на автомобил. Поради това че симулатора ще има три версии, като всяка версия ще представлява различен метод на адаптация, ще разделим изискванията в отделни категории за всяка версия.

4.2.1. Изисквания към симулатора

Неадаптивна версия

- Запознаване на играча с правилата и контролите на играта и въвеждане на имейла на играча за да започне да я играе.
- Следене и изобразяване скоростта на играча по време на играта.
- Следене и изобразяване оставащото разстояние за завършване на текущата обиколка.
- Автомобилът който ще бъде управляван от играча да има навигация позволяваща му да управлява колата напред, назад, завиване наляво, завиване надясно и възможността колата да скача, при ситуации в които колата е заседнала и не може да помръдне.
- Използване на аудио и визуални ефекти за максимален реализъм.
- Промяна на околната среда при стартиране на нова обиколка.
- Обработка на резултатите на играча и тяхното записване в реално време.

Адаптивна версия спрямо зададени нива

- Запознаване на играча с правилата и контролите на играта и въвеждане на имейла на играча за да започне да я играе.
- Следене и изобразяване скоростта на играча по време на играта.
- Следене и изобразяване оставащото разстояние за завършване на текущата обиколка.

- Автомобилът който ще бъде управляван от играча да има навигация позволяваща му да управлява колата напред, назад, завиване наляво, завиване надясно и възможността колата да скача, при ситуации в които колата е заседнала и не може да помръдне.
- Използване на аудио и визуални ефекти за максимален реализъм.
- Промяна на околната среда при стартиране на нова обиколка.
- Обработка на резултатите на играча и тяхното записване в реално време.
- Адаптиране на сложността на симулатора, в реално време, спрямо постигнатите резултати на играча.
- Адаптацията да се осъществява като резултатите стигнат определени зададени нива, спрямо които се определя дали сложността ще се увеличи или намали.

Адаптивна версия спрямо криви на учене

- Запознаване на играча с правилата и контролите на играта и въвеждане на имейла на играча за да започне да я играе.
- Следене и изобразяване скоростта на играча по време на играта.
- Следене и изобразяване оставащото разстояние за завършване на текущата обиколка.
- Автомобилът който ще бъде управляван от играча да има навигация позволяваща му да управлява колата напред, назад, завиване наляво, завиване надясно и възможността колата да скача, при ситуации в които колата е заседнала и не може да помръдне.
- Използване на аудио и визуални ефекти за максимален реализъм.
- Промяна на околната среда при стартиране на нова обиколка.
- Обработка на резултатите на играча и тяхното записване в реално време.
- Адаптиране на сложността на симулатора, в реално време, спрямо постигнатите резултати на играча.
- Адаптацията да се осъществява като според показаните резултати се откриват крива на учене по която да се определи промяната на сложността на симулатора.

4.2.2. Идентифициране на отделните роли в системата

След анализ на системните изисквания за симулатора на управление на автомобил можем да определим ролите в симулаторната видеоигра. Дефинирани са следните роли:

U1. Играчът – човек ползващ входни устройства за да подава команди към симулатора, според които да се извършат някакви действия в играта.

U2. I/O Devices (like, Mouse, keyboard, speaker, monitor) – Устройствата, които се използват за въвеждане на данни, се наричат входни устройства (Input Devices). Входното устройство може да чете данни и да ги преобразува във форма, която компютърът може да използва. Изходните устройства могат да генерират готовия продукт от машинната обработка във форма, удобна за използване и четее от хората. За дадена игра трябва да има силно взаимодействие между потребителя и играта, която той играе. Затова периферните устройства като мишка, клавиатура, джойстици и монитори играят основна роля, за да направят играта интерактивна.

U3. Graphics Engine – Графичният енджин е софтуер, който във връзка с приложен програмен продукт помага да се начертаят графични изображения на дисплея на компютъра. Думата "енджин" в компютърната област се отнася до софтуер, който помага за извършването на определен тип обработка на програми, като например енджин за текстови реч, енджин за бази данни, енджин за оформление и графичен енджин.

Графичният енджин помага за подобряване на графиката на играта чрез увеличаване на резолюцията и увеличаване на броя на пикселите на единица площ. Този енджин прави сцените на играта да са ясни и да работят гладко.

U4. Sound/Audio Engine - Аудио / звуковият енджин е компонентът, който се състои от алгоритми за справяне със звука, а вградените програми се записват в него, за да се справят с вградените в играта звукови ефекти. Той има способността да извършва изчисления с помощта на CPU или на всяка ASIC (Application Specific Integrated Circuit).

U5. Rendering and Vision-Input Engine – Енджина за рендърване заедно с входната система за видения (зрение) произвежда 3D анимирани графики, използвайки различни техники, като растеризация и проследяване на лъчи. Те биват програмирани и компилирани, за да бъдат изпълнени на който и да е процесор или графичен процесор, повечето енджини за рендиране се разработват на един или повече API като Direct3D и / или OpenGL, които предлагат слой за абстракция на софтуер за графичния процесор (GPU). Библиотеките от ниско ниво, като DirectX или OpenGL, са популярно включени в игрите, защото предоставят независим от хардуер достъп до различни хардуерни средства. Тези хардуерни устройства могат да бъдат входни устройства като мишка, клавиатура и джойстик.

U6. DLL files and Drivers/Device APIs - Файл с разширение DLL (библиотека за динамични връзки) е тип файл, който включва инструкции, написани под формата на програми, които могат да бъдат извикани или използвани от други програми за изпълнение на определени задачи. По този начин различните програми могат да споделят способностите и характеристиките, които са програмирани в един файл. Тези системни файлове играят поддържаща роля за изграждане на архитектурата на играта и помагат да се подобри нейната ефективност. API на устройството може да бъде дефиниран като API (Application Programming Interface), който позволява на разработчиците да създават приложение, което в крайна сметка може да взаимодейства с хардуерни устройства, свързани или инсталирани с вашата система. Приложният програмен интерфейс (API) за устройства обикновено осигурява на крайните потребители да използват своя свързан хардуер за взаимодействие със системата.

4.2.3. Идентифициране на външните системи

За външните системи с които ще комуникира симулатора, бе идентифицирана само една такава:

E1. Уеб Услуга за записване на резултатите – тази уеб услуга ще се намира на сървър. Целта и е да записва резултатите по време на игра в отделни файлове, за всеки един играч. Файловете ще имат определен формат за да могат да бъдат свалени от сървъра и анализирани.

4.3. Нефункционални изисквания

Втората от двете основни и най-важни задачи преди анализа и извличането на изискванията е дефиниране на нефункционалните изисквания към симулатора.

4.3.1. Изисквания за производителност и бързодействие

- Симулатора трябва да изобразява сцените бързо. Играта трябва да върви с 60 fps (frames per second). Това означава 60 сцени да могат да бъдат изобразени за 1 секунда или една сцена за 16.7 мили секунди.
- Симулаторът в рамките на всеки frame (ли иначе казано за 16.7 мили секунди), трябва да може да изпраща към уеб услугата резултатите на играча за да бъдат записани.

4.3.2. Изисквания за сигурност

- Симулатора и уеб услугата гарантира криптиран трафик, за да пази потребителските данни предвид тяхната защита от закона за личните данни.

4.3.3. Изисквания за надеждност и контрол на работата и дефектите

- Симулатора трябва да записва и показва 100% от грешките, възникнали по време на нейната употреба.
- В случай на възникнал срив заради неточност в кода (бъг), същата трябва да се отстрани в рамките на 24 часа от нейното установяване и да се предостави новата версия на симулатора на играчите.

4.3.4. Изисквания за преносимост

- Симулаторът поддържа следните операционни системи: Windows
- Симулаторът поддържа различни размери на екранните устройства, които я достъпват.
- Симулаторът поддържа следните уеб браузъри: Mozilla Firefox 42+, Microsoft Edge 42+

4.3.5. Изисквания за използваемост

- Симулаторът трябва да може да бъде достъпван от всички потребители. Цялата видео игра е публично достъпна и безплатна.

4.3.6. Изисквания за изпитаемост

- Симулатора трябва да бъде тестван и преди цялостното му завършване с цел дали отговаря на първоначалните изисквания.

4.4. Преглед на симулатора видеоигра

След като анализирахме и определихме изискванията към симулатора ще направим цялостен преглед на играта, включващ в себе си потокът на играта(game flow), геймплейът на играта и механиките ѝ, нива, интерфейс и др. [2] [3]

4.4.1. Кратко описание на играта

Играта е single-player и представлява 3D симулатор на управление на автомобил, като играчът бива предоставен с трудности, по време на карането на колата, с които трябва да се справи.

Фиг. 9 – screenshot (екранна снимка) от интерфейса на играта при отлични условия

Фиг. 39 – screenshot (екранна снимка) от интерфейса на играта при мъгла

Фиг. 28 – screenshot (екранна снимка) от интерфейса на играта при мъгла и дъжд

Целта на играта е да се направят 4 обиколки на трасето, като всяка обиколка ще представя различна трудност с която играчът трябва да се справи. Играта е насочена към хора които се обучават да управляват автомобил и към тези които искат да подобрят своите шофьорски умения.

4.4.2. Контроли на играта на играта

Играчът има нужда единствено от клавиатура за да управлява играта. Симулаторът предлага следните контроли за управление на автомобила:

Клавиш	**Действие при натискане на дадения клавиш**
'W' или '↑'	Придвижва автомобила напред
'S' или '↓'	Придвижва автомобила назад
'A' или '←'	Автомобила прави ляв завой
'D' или '→'	Автомобила прави десен завой
Space bar	Колата скача във въздуха

Фиг. 10 – контролите за управление на автомобила в играта

4.4.3. Интерфейс на играта

В долния ляв ъгъл има скоростомер, показващ скоростта с която кара в момента играчът, а около него има зелен индикатор, показващ прогреса на текущата обиколка.

4.4.4. Аудио ефекти

Различни обекти в играта издават различни аудио ефекти, като например двигателя при каране на колата, звукът от дъжда, заглушаване на шума от колата при преминаване през тунел, издаване на звук при хлъзгане, шумът от буксуващи гуми и др.

4.4.5. Поток на играта (Game Flow)

Фиг. 11 – Game Flow диаграма изобразяваща различните процеси през които минава симулатора

След като играча е стартирал приложението се появява началният екран. На него е описано какви са контролите за управление и каква е целта на играта. За да започне играта, играчът трябва да въведе валиден имейл адрес. След въвеждането на имейла си играчът стартира своите обиколки около трасето. Спрямо това коя версия играе играча се случват различни неща по време на различните обиколки.

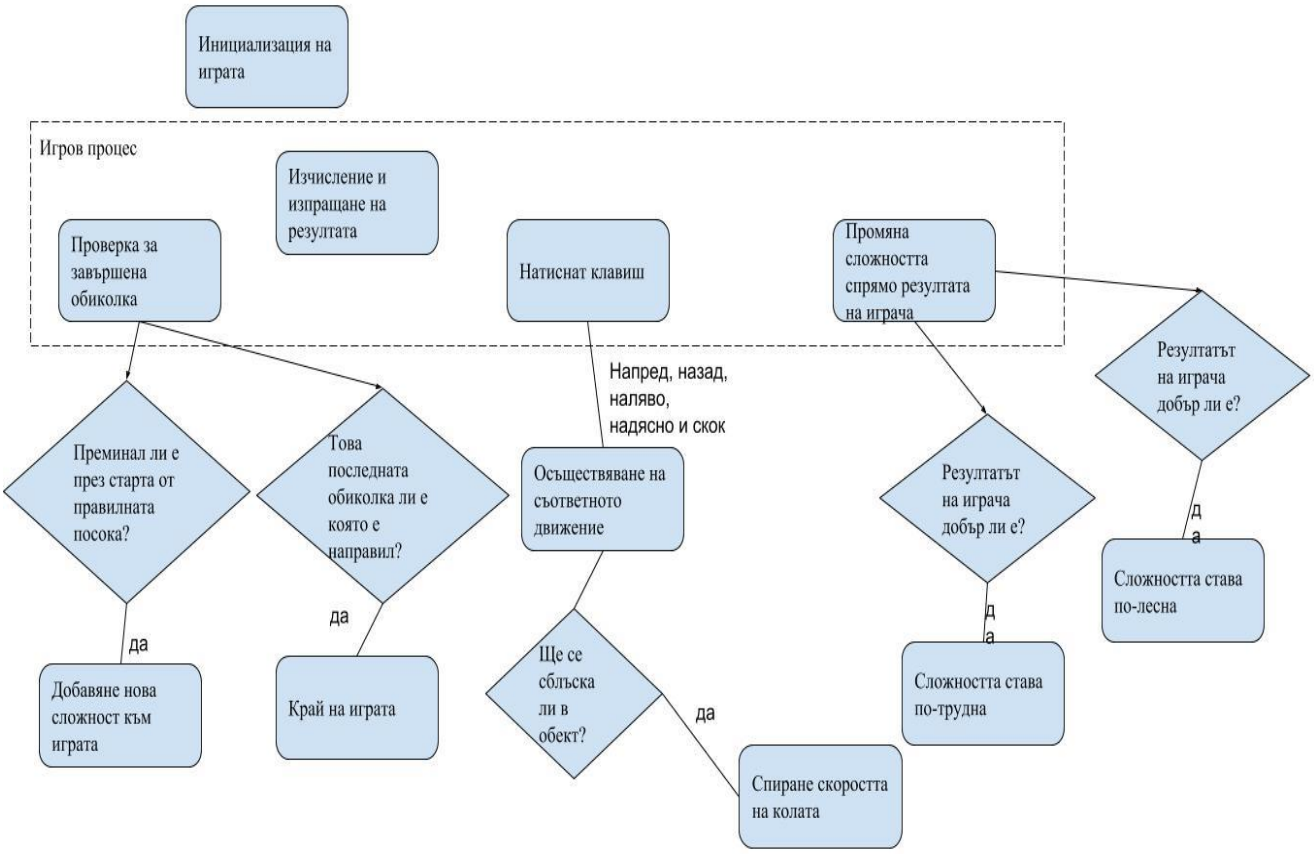
При неадаптивната версия игровият процес протича по следният начин:

- Първата обиколка играчът кара колата сред идеални условия.
- По време на втората обиколка се свечерява.
- При третата обиколка се появява и мъгла.
- По време на последната обиколка, започва да вали дъжд.

При адаптивните версии спрямо зададени нива и спрямо криви на учене игровият процес е следния:

- Първата обиколка играчът кара колата сред идеални условия.
- На втората обиколка се свечерява и спрямо резултатите на играча става по-тъмно или по-светло.
- На третата обиколка се появява и мъгла и спрямо резултатите на играча става по-гъста или по-рядка.
- На последната обиколка, започва да вали и дъжд и спрямо резултатите на играча трасето става по-хлъзгаво или се връща в нормалното си състояние.

В реално време, докато играча управлява автомобила, се изпращат текущите резултати, направени от играча, към уеб услуга (сървис), който ги записва. След завършването на последната обиколка играта приключва и се появява финалният екран от където играча може да излезе от приложението.



Фиг. 12 – Game Flow диаграма изобразяваща по-подробно игровият процес на играта

4.4.6. Прогресия в играта

Играчът трябва да направи 4 обиколки на трасето с предоставеният му автомобил, за да завърши успешно играта. Всяка обиколка предоставя на играча различни трудности с които ще трябва да се справи.

4.4.7. Механики

Тук ще обсъдим правилата на играта, модела на вселената около играта, как отделните части сработват и т.н.

Всяка една направена обиколка в играта предоставя нова сложност с която играча трябва да се справи. Трасето, по което е позволено да се движи автомобила, управляван от играча, е ограничено с най-различни обекти, които не му позволяват да кара колата безцелно в различни посоки. Трасето има завои, наклони по които да се изкачва или слиза с колата. Обектите, ограничаващи движението на автомобила могат да бъдат мантинели, поставени прегради, мостове, тунели. При сблъсък в тези обекти колата претърпява катастрофа която може да доведе до заседаване на колата. Поради тази причина е дадена възможност на играча да скача с колата, за да може да продължи своето преминаване по трасето. Повечето обекти като дървета, планини, водата, улично осветление, сгради и др. представляват част от декора на картата с които играчът не може да взаимодейства. Единственият обект с който може да взаимодейства играча това е автомобила който управлява.

4.4.8. Физика в играта

Автомобила не може да минава през други обекти, като мантинела, друг три-измерен обект. Също така колата не може да лети, тя се движи единствено по трасето. Източникът на светлина е слънцето, което може да бъде скрито за индикация на свечеряване. Появата на мъгла намалява видимостта на играча. Започването на дъжд прави пътя по хлъзгав и кара колата да се хлъзга по него ако не намали скоростта си. Камерата чрез която играча вижда самата игра е от перспективата на птичи поглед или още наречена Third person view. Самата камера се движи заедно с колата. При преминаване в тунела звуците от околната среда са заглушени, дъждът спира, а при излизането му всички визуални и звукови ефекти се нормализират.

4.5. Изводи

След прегледа на функционалните и нефункционалните изисквания, и подробното разглеждане на механиките, игровият процес, потокът на играта, физиката, контролите и дизайна на играта може да се сложи първоначалната рамка на предвижданите функционалности. Анализът показва, че основните функционалности са свързани с управлението на автомобила в различни условия, промяна на трудността, спрямо резултатите на играча, при каране в тези условия и изчисление и изпращане на резултата на играча.

5. Проектиране на симулатор за управление на автомобил

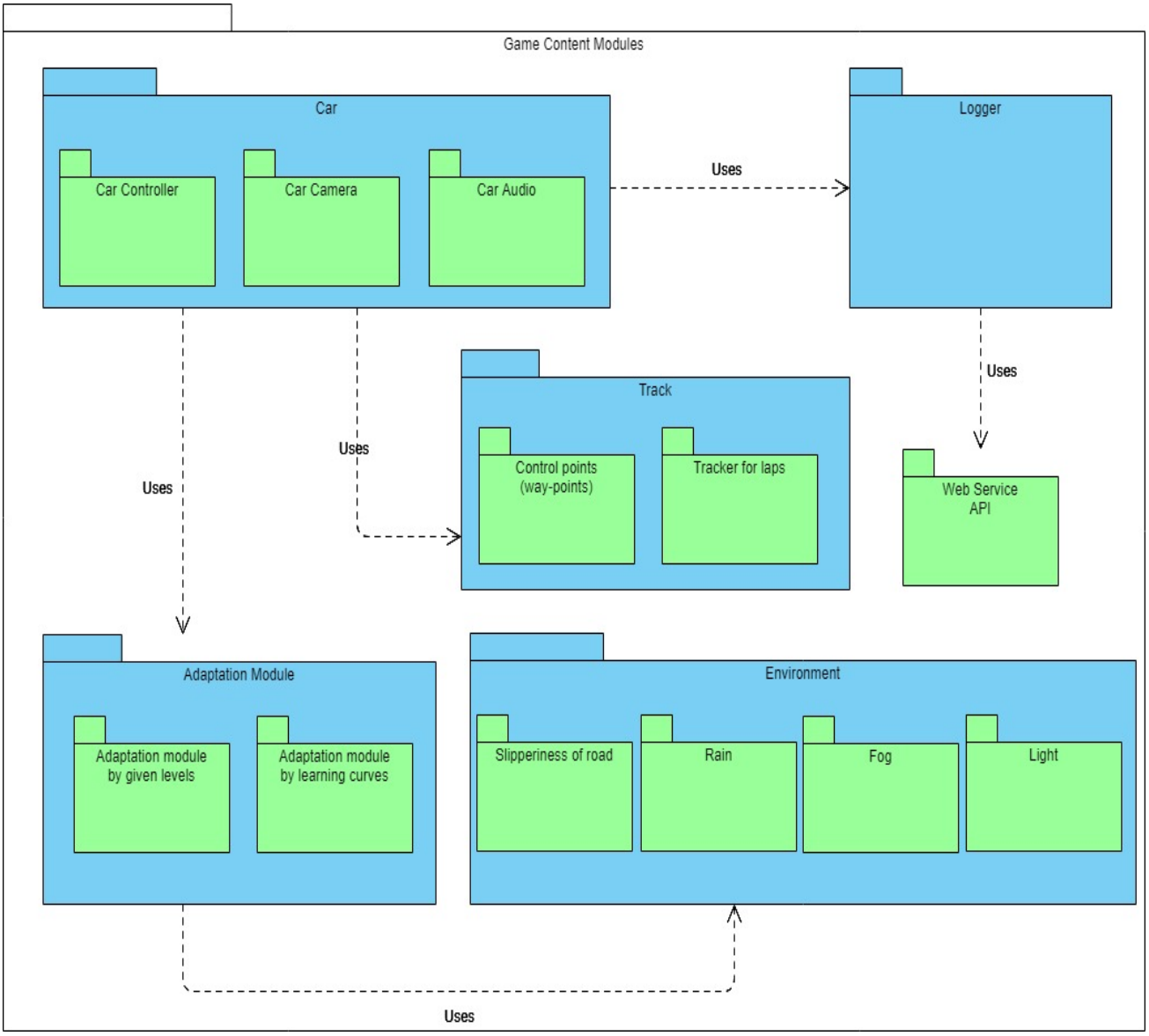
Проектирането на системата е онази част от реализацията на крайния софтуерен продукт, която цели да обособи нейната архитектура. Основната идея на създаването на тази архитектура е разделянето на бъдещата реализация на отделни модули според функционалните изисквания и същевременно да отговори на нефункционалните такива към системата. Тъй като платформата Unity ни предоставя модули и под модули за game engine на играта, ще бъдат разгледани различните подмодули в game content модула на играта. Наред с това е необходимо да се скицира примерен потребителски интерфейс, за да се изпълни едно от най-важните изисквания към играта – тя да бъде с адаптивен дизайн, подходящ за устройства с различи размери на екрана, както и използваемостта.

5.1. Архитектура

Тук ще бъдат разгледани основните структури и компоненти, които изграждат нужните елементи на играта, покриващи изискванията.

5.1.1. Декомпозиция на модулите, под модулите и употреба

Модулната декомпозиция е в основата на бъдещата разработка на всяка софтуерна система. Чрез нея се създава логическото разделение на кода на отделни единици за реализация. Декомпозицията на модулите определя в голяма степен възможността за промяна и ключовите изисквания към системата като обособява логически свързаните функционалности на едно място. Много често, декомпозицията на модулите служи за предвиждане и по-лесно проследяване на промените, разпределението на работата по екипите, заменяемост на логически независимите модули от системата. Това са и факторите, които определят избора за документиране на тази структура предвид нейните мащаби, бъдеща поддръжка и последвала разширяемост. [27]



Фиг. 13 – модулна декомпозиция

На Фиг. 13 са изобразени следните модули, които са част от модулната декомпозиция на системата.

Car – Това е модулът който, предоставя компоненти за управление на кола, камерата чрез която играча вижда играта, и звуците на колата.

- **Car Controller –** това е под модулът занимаващ се с управлението на колата, отчитащ текущите резултати и ги предава на Logger модула.
- **Car Camera –** това е под модулът който предоставя възможността да вижда виртуалния свят. Прикрепен е към колата, за да може играчът да управлява колата подобаващо.
- **Car Audio –** това е под модул занимаващ се с различните звуци които различните компоненти на колата издават.

Track – Това е модулът представляващ трасето по което ще се движи колата.

- **Control Points –** това е под модул на трасето занимаващ се с отчитане на време, скорост, брой колизии и други характеристики. Те се отчитат, когато колата премине през такава точка от трасето.
- **Tracker for laps –** това е под модул проследяващ колко обиколки е направил играчът. Спрямо това на коя обиколка се намира играча се променя и околната среда предоставяща сложностите с които трябва да се справи играча.

Adaptation module – това е модулът който получава резултатите и спрямо версията на адаптация, променя сложността на играта по различни начини. Промяната на сложността се прави като се променят характеристиките на околната среда, като промяна силата на светлината, гъстотата на мъглата, хлъзгавостта на трасето.

- **Adaptation module by given levels** – симулатора на управление на автомобил с такъв модул на адаптация, променя сложността на играта, като се следи дали резултатът на играча е достигнал определени зададени нива. Спрямо това какво ниво е достигнал резултата се променят съответните елементи на сложност на обиколката която прави в момента играча.
- **Adaptation module by learning curve** – симулатора на управление на автомобил с такъв модул на адаптация, променя сложността на играта, като сравнява текущия резултат на играча по зададени криви на учене. Ако намери такава крива на учене, спрямо нея прави промени в сложността на играта.

Environment – този модул се занимава с физиката на всички обекти във виртуалният свят. Също така се грижи за това, да променя околната среда (променя сложността на играта), спрямо получените инструкции от модула за адаптация, които оказват дали промените да улеснят играча или да го затруднят.

- **Light –** този под модул се грижи за светлината, от къде ще идва, как ще осветява различните обекти, и колко силно свети, и в какъв цвят ще свети.
- **Fog –** този под модул се грижи за графичните ефекти на мъглата, като може да променя нейната гъстота. По този начин видимостта на играча може да се влоши или подобри.
- **Rain –** този под модул се грижи за дъжда във виртуалния свят на играта. Той определя с каква сила да вали и колко да е гъст дъжда.
- **Slipperiness of road –** този под модул има грижата при силен дъжд да прави пътя по който върви колата по-хлъзгав, което затруднява управлението на колата.

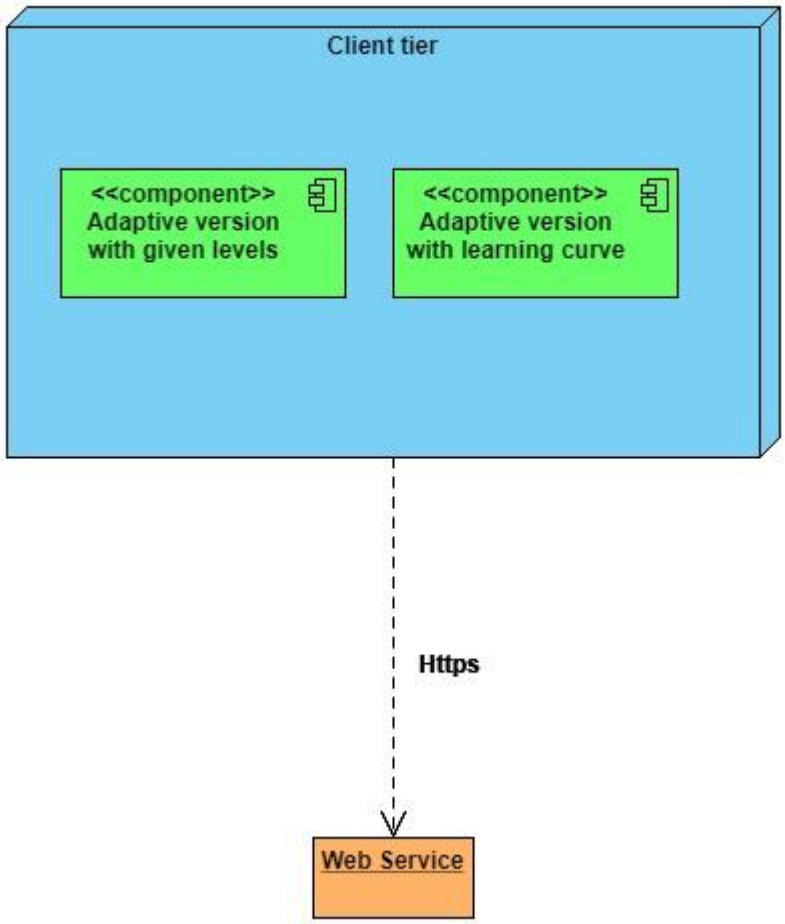
Logger – този модул се грижи за обработката на получените данни от Car controller модула, след което да ги изпрати на външен за системата модул, където да бъдат записани.

Web Service API – външен за системата модул, който получава изпратените му данни от играта и ги записва на сървъра, на който се намира, във CSV файлов формат.

5.1.2. Структура на компонентите и конекторите

Структурата на компонентите и конекторите спомага за изобразяването на отделните процеси в системата и тяхната комуникацията помежду им. Освен това с нейна помощ

се предоставя възможността да се проследят разположението на отделните компоненти върху софтуер и хардуер, които се явяват външни за системата.



Фиг. 14 – изглед на компонентите и конекторите на системата

На Фиг. 14 се разглежда комбинирания изглед на компонентите и конекторите. Елементите, които структурата разглежда са:

- **Client tier –** това е клиентският хардуер, върху който е инсталирана една от версиите на играта.
- **Adaptive version with given levels –** това е адаптивната версията на играта със зададени нива, по които да се променя сложността на играта.
- **Adaptive version with learning curve –** това е адаптивната версията на играта с криви на учене, по които да се променя сложността на играта.
- **Web Service –** това е уеб улуга (сървис) на който се изпращат данните от различните версии на играта, и се записват на сървъра.

Описанието на връзките между отделните компоненти:

Client – Web Service – Комуникацията се предава посредством HTTPS протокола, който е необходим, за да се гарантира сигурността на данните.

5.2. Модел на данните

Общо прието определение за модел на данните е съвкупност от абстрактни понятия, чрез които се описват свойствата на разработеното приложение, на което са присъщи следните свойства:

- Статични свойства – задават се чрез обекти, техните атрибути и взаимовръзки помежду им
- Динамични свойства – определят се от операциите върху обектите

Определението за моделиране на данните се отнася до процеса на прехвърляне на нещата от реалния свят в света на компютрите. До началото на 90-те години на XX век компютрите се наричат Електронно Изчислителни Машини, като обработването на данни става посредством линейни структури от вида матрици, масиви и т.н. В днешно време релационните бази от данни безпроблемно осигуряват съхранение на този тип данни в хранилища, състоящи се от таблици, т.е. в бази данни, където съхраняват данни от компютърната система чрез специални структури и с помощта на таблици (един или няколко файла в зависимост от системата за управление на данни). Базите от данни е специфичен начин за оптимално съхранение на много голямо количество информация, която трябва да се използва многократно за решаване на много задачи от много потребители. Базата от данни прилича на файлова система на компютъра за съхранението на данни в компютъра чрез файлове, т.е. на физическо ниво тези два елемента са подобни. Съществената разлика се състои в начина на достъп и на обработване на съхранените данни.

При файловата система достъпът до данните и тяхната обработка се осъществява чрез команди на операционната система. Пример за такъв достъп е приложение работещо под операционна система Windows е налице меню File, съдържащо команди за опериране с файлове.

При базите данни достъпа и тяхната обработка се осъществява чрез команда от специален език за програмиране, насочен към приложения в базите данни – SQL (език за структурни заявки). [26]

File Name: <GameType>_CarSim_<Email>_<DateTime>.csv
Local Time
Average Velocity
Number Of Collisions
Impact Of Collisions
Average Impact
Performance

Фиг. 15 – структурата на файла, съхраняващ резултатите на играча

Играта симулатор на управление на автомобил не използва релационни бази от данни за съхранение на данните (резултатите) от всеки играч, а използва файловата система на сървъра на който се помещава уеб услугата.

Както се вижда от Фиг. 15, структурата на файла се състои от четири колони в които се записват данни за:

- Локалното време – то винаги е уникално
- Средна скорост – средната скорост на играча за даденият момент
- Брой колизии – броят на ударите на колата в други обекти
- Въздействие на сблъсъците – оценка на колизията (сблъсъка)
- Средна стойност на колизиите – въздействието на колизиите делено на броя на колизиите зялото разделено на 50

- Резултат – средната скорост разделена на въздействието на сблъсъка разделен на 50 събрано с 1 или avrVel/(impact of collision/50 + 1)

За всеки frame играта изпраща HTTPS POST заявка със специфична структура от тип String: ***String.Format("{0}; {1}; {2}; {3}; {4}; {5}", avgVelocity, numberOfCollisions, impactOfCollisions, avgImpact, performance, TrackOfRace.GetNumOfTurn())***

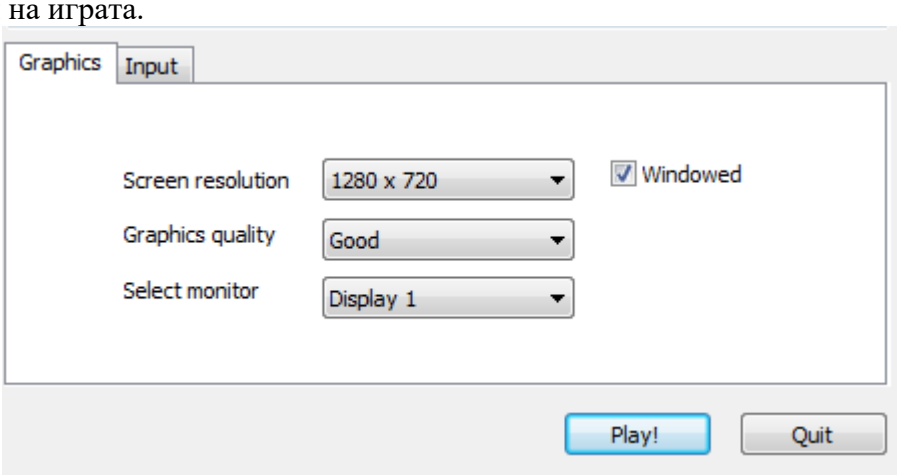
Уеб сървъра от своя страна обработва заявката и записва данните във файловата си система във файл с име типа на играта (адаптивна или не адаптивна), долна черта, CarSim(текст), долна черта, емайла на играч, долна черта, датата когато е генериран файла в и накрая разширение „CSV“.

5.3. Потребителски интерфейс

Един от основните елементи на проектирането е създаването на потребителския интерфейс. Това се налага от изискванията за поддръжка от различни по размер устройства, както и различни видове и версии на уеб браузъри. Основен фактор отново се явява избора и използваните технологии за разработка.

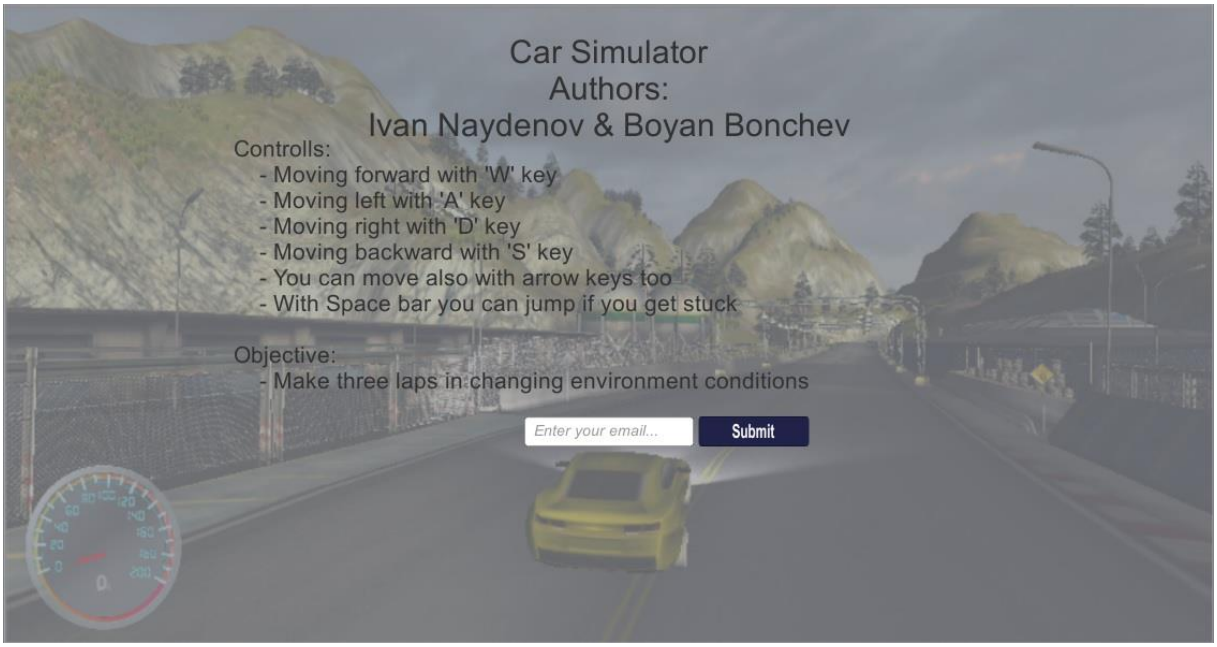
Основната задача пред проектирането му е създаването на адаптивен дизайн. Идеята му е да се създаде един и същ изходен код, който да взаимодейства с потребителя и да променя динамично подредбата и оразмеряване на елементите според размера на екрана. Това неизменно води до едно по-добро потребителско изживяване, както и по-лесна поддръжка и последвала преизползваемост. Технологиите които предлага платформата Unity, дават възможност разработените версии на симулатора да се играят, както на устройства с различни резолюции така и на различни видове и версии на уеб браузъри.

Симулатора има три версии, но всяка една от тези версии ще има един и същ интерфейс. Използват се едни и същи елементи и компоненти за интерфейсите на трите версии на играта.



Фиг. 16 – настройка на графиката

При стартирането на играта, потребителя ще може да избере настройките с които иска да играе на симулатора. Както се вижда от снимка 2 играчът може да избере резолюцията на екрана на която иска да играе играта, също така може и да избере с какво качество ще е графика на играта. При натискане на бутона ‘Play’ играчът стартира играта, а при натискане на бутона ‘Quit’ играчът излиза от приложението.



Фиг. 17 – изглед на началния интерфейс на играта

На Фиг. 17 е изобразено как изглежда началният екран на играта, при натискане на бутона ‚Play‘ от предишния прозорец. В горната централна част на екрана се намира заглавието на играта и авторите ѝ. Централно в средата на екрана се намира обяснение на контролите които да използва в играта, а под тях и целите на самата игра. Най-отдолу на екрана се намира текстово поле за въвеждане на имейла от играча и до него бутон, които при натискането му стартира играта, ако въведеният имейл е валиден.

Фиг. 18 – изглед на интерфейс по време на игра в отлични условия

Фиг. 38 – изглед на интерфейс по време на игра в лоши условия

Фиг. 19 – изглед на интерфейс по време на игра в лоши условия

Потребителският интерфейс по време на играене на играта, изобразен на Фиг. 18, Фиг. 19 и Фиг 38 демонстрира следните елементи на дизайна:

В долният ляв ъгъл на екрана ще се намира скоростомера, показващ текущата скорост на колата в играта. А около него има зелен индикатор показващ прогреса на текущата обиколка и колко остава до края ѝ. Останалата част от екрана ще изобразява самото управление на колата по трасето със заобикалящата го околна среда.

5.4. Извод

Така описаната архитектура цели в максимална степен да отговори на специфицираните изисквания и да определи насоките, по които ще се реализира и тества играта в следващите ѝ фази.

Така разгледаната модулна декомпозиция и употреба е добра предпоставка за бъдещата разширяемост на играта. Чрез ясното им разделение и скриването на вътрешната им структура, както и специфицирането на евентуални промени по модулната декомпозиция, се създават добри предпоставки за обособяването на отделните функционалности и аспекти на симулатора и бъдещото му разширяване, без да се налагат промени или да се нарушава целостта и действието на вече съществуващите модули в играта.

6. Реализация, тестване и внедряване

След създаването на софтуерната архитектура на системата е ред и на нейната реализация и тестване. Позовавайки се на модулната декомпозиция, можем да отбележим какво ще бъде логическото разделение на кода и цялостната му реализация. Имайки предвид че ще използваме платформата Unity за разработка на симулатора, тя предлага готови физически обекти които да бъдат използвани за реализацията на отделните компоненти съставящи играта. Unity предлага готови компоненти за терен (обекти от околната среда като дървета, вода, кали, сгради, предмети и др.), кола, дъжд, мъгла, светлина, различни аудио и звукови компоненти. Използвайки Unity се спестява моделирането, измисляне на дизайна и разработването на тези компоненти нужни за играта. Така акцента пада върху разработката на тяхното поведение във виртуалният свят на играта. Това ще се реализира със скриптовата структура, която предлага Unity. Всеки скрипт може да бъде назначен на даден обект. Така всеки един модул и под модул описан в модулната декомпозиция ще има прикачен към него скрипт реализиращ неговата логика и това което трябва да прави.

На базата на така направеният анализ се създава следният ред на разработка:

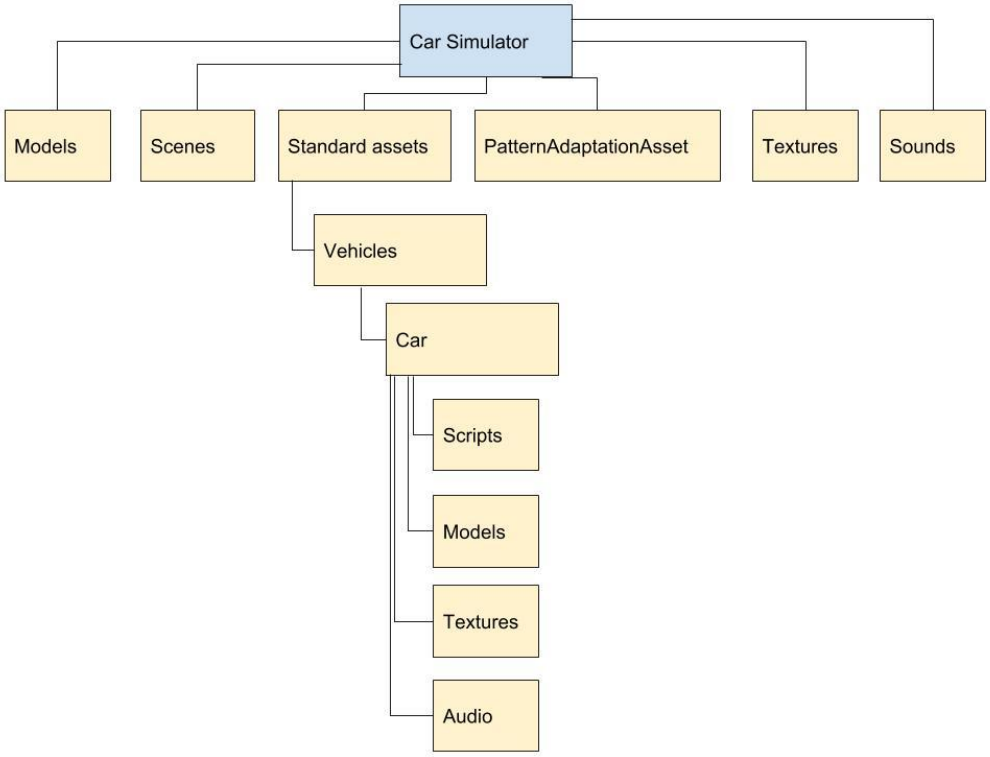
1. Имплементация на управлението на колата, инструментариум за следене характеристиките на колата (скоростомер, индикатор за прогреса на обиколката и др.).
2. Имплементация на логиката на камерата на колата.
3. Имплементация на аудио и звукови ефекти на колата.
4. Имплементация на контролните точки по трасето.
5. Имплементация на логика за следене на обиколките.
6. Имплементация на различните методи за адаптация на сложността.
7. Имплементация на логиката за различните под модули на околната среда (дъжд, мъгла, светлина, хлъзгавина на пътя и др.).
8. Имплементация на логиката за събиране на резултата.
9. Имплементация на модула за изпращане на данните от играта към уеб услугата която ще ги записва на сървъра на който се намира.
10. Имплементация на уеб услугата.
11. Модулно тестване.
12. Компонентно тестване.
13. Системно тестване.
14. Unit testing (юнит тестване).
15. Acceptance testing (тестване за приемливост на симулатора)

6.1. Структура на проекта

Представената по-долу Фиг. 20 има за цел да опише структурата на проекта, както и неговото физическо и логическо разделение на съставните части: модули, подмодули, външни компоненти и други. Структурата е направена с няколко главни директории, в които се съхраняват общи файлове за използване от всички модули и под модули.

- **Models –** В тази директория са общите модели на елементи и обекти, намиращи се във виртуалния свят на симулатора. Тези модели са суровият вид (как изглежда даден елемент или обект без приложени характеристики върху него като цвят, материал от който е направен, отражение, сянка и други) на обектите и елементите.
- **Scenes –** Тази директория съхранява различните сцени и нива на симулатора, като начален екран, сцената с колата и трасето, финален екран и др.

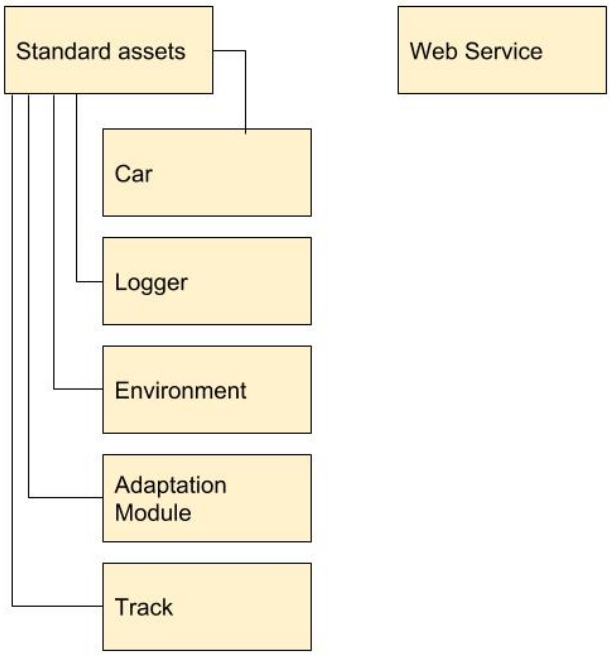
- **PatternAdaptationAsset –** В тази директория е инсталирана библиотеката и компонентите на кривата на учене (Player-centric rule-and-pattern-based adaptation asset). Тя ще се използва във версията на играта с адаптация чрез крива на учене.
- **Textures –** Обща директория с различните текстури които се слагат на дадени обекти и елементи.
- **Sounds –** Обща директория на всички аудио файлове използвани от дадени обекти в играта.
- **Standard Assets –** В тази папка се намират всички заложени обекти във реалния свят, като терен, кола и др. Всяка една от вътрешните директории имат еднаква структура която отделя различните компоненти на даден обект специфични само за него. Както е дадено за пример автомобила има папка скриптове които са специфични само за колата, папка модели специфични само за нея, както папка текстури и аудио специфични само за нея.



Фиг. 20 – Дървовидно представяне на структурата на играта

6.2. Реализация на модулите

Реализацията на модулите разглежда техните интерфейси за връзка и интересни аспекти по време на имплементация. Това включва ключови методи, алгоритми механизми, които покриват функционалните и нефункционални изисквания към всеки отделен модул и подмодул. На Фиг. 21 е показано физическото структуриране на модулите от декомпозицията.



Фиг. 21 – реализиране на модулите от системата

6.2.1. Реализация на модула за автомобила

В този модул са включени управлението на колата, камерата, и различните аудио и звукови елементи които една реална кола може да издава. Също така тук са имплементирани методите за взимане и изчисление на различни данни по време на управлението на колата, като скоростта ѝ, сцеплението, броят сблъсъци с други обекти и др.

Управлението на колата да се движи по трасето се изпълнява от следният фрагмент:

```
public void Move(float steering, float accel, float footbrake, float handbrake)
            {
                  for (int i = 0; i < 4; i++)
                  {
                        Quaternion quat;
                        Vector3 position;
                        m_WheelColliders[i].GetWorldPose(out position, out quat);
                        m_WheelMeshes[i].transform.position = position;
                        m_WheelMeshes[i].transform.rotation = quat;
                  }

                  //clamp input values
                  steering = Mathf.Clamp(steering, -1, 1);
                  AccelInput = accel = Mathf.Clamp(accel, 0, 1);
                  BrakeInput = footbrake = -1 * Mathf.Clamp(footbrake, -1, 0);
                  handbrake = Mathf.Clamp(handbrake, 0, 1);

                  //Set the steer on the front wheels.
                  //Assuming that wheels 0 and 1 are the front wheels.
                  m_SteerAngle = steering * m_MaximumSteerAngle;
                  m_WheelColliders[0].steerAngle = m_SteerAngle;
                  m_WheelColliders[1].steerAngle = m_SteerAngle;

                  SteerHelper();
                  ApplyDrive(accel, footbrake);
                  CarSpeed();

                  //Set the handbrake.
                  //Assuming that wheels 2 and 3 are the rear wheels.
                  if (handbrake > 0f)
                  {
                        var hbTorque = handbrake * m_MaxHandbrakeTorque;
                        m_WheelColliders[2].brakeTorque = hbTorque;
                        m_WheelColliders[3].brakeTorque = hbTorque;
                  }


                  CalculateRevs();
                  GearChanging();

                  AddDownForce();
                  CheckForWheelSpin();
                  TractionControl();
            }
```

Фрагмент 1 – функция за придвижване на колата в желаната посока.

Спрямо това дали колата върви напред или назад, с каква скорост и дали е завито, наляво или надясно се правят изчисления, които придвижват целият автомобил. От функцията CarSpeed() се изчислява скоростта в километри или в мили според предпочитанията на играча. Изпълнението на тази функция е показано на следният фрагмент:

```
private void CarSpeed()
		{
			float speed = m_Rigidbody.velocity.magnitude;
			switch (m_SpeedType)
			{
				case SpeedType.MPH:

					speed *= 2.23693629f;
					if (speed > m_Topspeed)
						m_Rigidbody.velocity = (m_Topspeed /
2.23693629f) * m_Rigidbody.velocity.normalized;
					break;

				case SpeedType.KPH:
					speed *= 3.6f;
					if (speed > m_Topspeed)
						m_Rigidbody.velocity = (m_Topspeed / 3.6f)
* m_Rigidbody.velocity.normalized;
					break;
			}

			Speedometer.ShowSpeed(speed, 0, m_Topspeed);
			m_CurrentSpeed.text = Math.Round(speed).ToString();
		}
```

Фрагмент 2 – функция за изчисляване скоростта в километри в час или мили в час.

CheckForWheelSpin() функцията определя дали гумите на колата превъртат ако да, се появява дим и следи от гумата по пътя. А ето и как изглежда тя:

```
private void CheckForWheelSpin()
		{
			for (int i = 0; i < 4; i++)
			{
				WheelHit wheelHit;
				m_WheelColliders[i].GetGroundHit(out wheelHit);

				// is the tire slipping above the given threshhold
				if (Mathf.Abs(wheelHit.forwardSlip) >= m_SlipLimit ||
Mathf.Abs(wheelHit.sidewaysSlip) >= m_SlipLimit)
				{
					m_WheelEffects[i].EmitTyreSmoke();

					// avoiding all four tires screeching at the same time
					// if they do it can lead to some strange audio
artefacts
					if (!AnySkidSoundPlaying())
					{
						m_WheelEffects[i].PlayAudio();
					}
					continue;
				}

				// if it wasnt slipping stop all the audio
				if (m_WheelEffects[i].PlayingAudio)
				{
					m_WheelEffects[i].StopAudio();
				}
				// end the trail generation
				m_WheelEffects[i].EndSkidTrail();
			}
		}
```

Фрагмент 3 – функция пускаща визуални и аудио ефекти в зависимост от въртенето на гумите на колата.

AddDownForce() тази функция се използва за добавяне на повече сцепление по отношение на скоростта.

Следващият фрагмент показва кода изпълняващ аудио и звуковите ефекти.

```
private void StartSound()
		{
			// get the carcontroller ( this will not be null as we have
require component)
			m_CarController = GetComponent<CarController>();

			// setup the simple audio source
			m_HighAccel = SetUpEngineAudioSource(highAccelClip);

			// if we have four channel audio setup the four audio sources
			if (engineSoundStyle == EngineAudioOptions.FourChannel)
			{
				m_LowAccel = SetUpEngineAudioSource(lowAccelClip);
				m_LowDecel = SetUpEngineAudioSource(lowDecelClip);
				m_HighDecel = SetUpEngineAudioSource(highDecelClip);
			}

			// flag that we have started the sounds playing
			m_StartedSound = true;
		}
```

Фрагмент 4 – функция пускаща звуците на колата.

А камерата е вграден компонент от Unity, който сме избрали да прикачим към колата за да може играчът да следи нейното движение.

6.2.2. Реализация на ‘Logger’ модула

Този модул се грижи да обработи изпратените му данни и да ги изпрати под формата на HTTPS POST заявка към уеб услугата която се занимава със записването им в отделни файлове за всеки играч.

```
private void LogMessageInCSV(string logMessage, string columnNames, string
fileName)
		{
			fileName = prefixFileName + "_" + "CarSim" + "_" +
PlayerPrefs.GetString("email") + "_" + fileName + currentDate + ".csv";

		String logFileName = "http://huibg.000webhostapp.com/" +
serviceFileName + ".php";
		StartCoroutine(logWebServiceCall(logFileName, fileName, columnNames,
DateTime.Now.ToString("hh:mm:ss:ffff") + "," + logMessage));
		}

IEnumerator logWebServiceCall(string logFileName, string fileName, string
columnNames, string logContent)
		{
			WWW w = new WWW(logFileName + "?filename=" + fileName +
"&&columnNames=" + columnNames + "&&logContent=" + logContent);

			yield return w;
		}
```

Фрагмент 5 – функции за обработване получената информация и за нейното изпращане.

Формата на данните изглежда по следният начин:

```
String.Format("{0}; {1}; {2}; {3}; {4}; {5}", avgVelocity, numberOfCollisions,
impactOfCollisions, avgImpact, performance, TrackOfRace.GetNumOfTurn())
```

Фрагмент 6 – формат на данните.

Първо се записва средната скорост, след нея броя на колизиите, после въздействието от колизиите, средна стойност на колизиите, резултат на текущото представяне на играча и текущата обиколка.

6.2.3. Реализация на модула за трасето

Този модул слага определени точки на трасето по които може да се определи дали върви в правилната посока или не. Ако играчът поради някаква причина е тръгнал в обратната посока на движение, тогава му се появява съобщение което го предупреждава че върви в обратната посока. Тази логика е заложена в следващият Фрагмент 7:

```
int nameNumberOfWayPoint = Convert.ToInt16(this.gameObject.name.Replace("WayPoint", ""));
                           currentNameNumberWayPoint = nameNumberOfWayPoint;
                           if (passedWayPoints.Length == 0 && nameNumberOfWayPoint ==
numberOfWayPoints)
                           {
                                  isReverseStart = true;
                           }
                           else if (passedWayPoints.Length == 0 && nameNumberOfWayPoint ==
1)
                           {
                                  isReverseStart = false;
                           }

                           //detect the direction of the circle
                           nameNumberOfWayPoint = isReverseStart ? (numberOfWayPoints -
nameNumberOfWayPoint + 1) : nameNumberOfWayPoint;
                           if (currentWayPointEnter == nameNumberOfWayPoint &&
(passedWayPoints.Contains(currentWayPointEnter.ToString() + "&") || currentWayPointExit ==
currentWayPointEnter))
                           {
                                  currentWayPointEnter--;
                           }
                           else if (currentWayPointEnter < nameNumberOfWayPoint)
                           {
                                  currentWayPointEnter++;
                           }
                           LoadingBar.GetComponent<Image>().fillAmount =
(float)currentWayPointEnter / numberOfWayPoints;

                           if (currentWayPointExit == nameNumberOfWayPoint ||
passedWayPoints.Contains(nameNumberOfWayPoint + "&"))
                           {
                                  passedWayPoints =
passedWayPoints.Replace(nameNumberOfWayPoint + "&", "");
                                  currentWayPointExit--;
                           }
                           else if (!passedWayPoints.Contains(nameNumberOfWayPoint + "&"))
                           {
                                  passedWayPoints += nameNumberOfWayPoint + "&";
                                  currentWayPointExit = nameNumberOfWayPoint;
                           }
```

Фрагмент 7 – логика за проверка дали играчът не се върти в кръг.

А функцията която следи за номера на текущата обиколка е следната:

```
public static int GetNumOfTurn()
            {
                  return numOfTurn;
            }
```

Фрагмент 8 – Функция за връщане номера на текущата обиколка.

6.2.4. Реализация на модула за околната среда

Този модул се грижи за компонентите на околната среда, като дъжд, мъгла, светлина, хлъзгавост на пътя и тяхното изменение спрямо резултата на играча. Всичките компоненти са предоставени от Unity и вградени в играта. За това при реализацията им ние единствено трябва да се грижим за тяхното изменение спрямо резултатите на играча.

```
private void UpdateRain()
        {
            // keep rain and mist above the player
            if (RainFallParticleSystem != null)
            {
                if (FollowCamera)
                {
                    var s = RainFallParticleSystem.shape;
                    s.shapeType = ParticleSystemShapeType.ConeVolume;
                                  s.angle = 0.0f;
                                  s.radius = 25f;
                                  s.length = 25f;
                                  RainFallParticleSystem.transform.position =
Camera.transform.position;
                    RainFallParticleSystem.transform.Translate(0.0f, RainHeight,
RainForwardOffset);
                    RainFallParticleSystem.transform.rotation =
Quaternion.Euler(0.0f, Camera.transform.rotation.eulerAngles.y, 0.0f);

                                  if (RainMistParticleSystem != null)
                    {
                        var s2 = RainMistParticleSystem.shape;
                                        s2.shapeType =
ParticleSystemShapeType.Box;// HemisphereShell;
                        Vector3 pos = Camera.transform.position;
                        pos.y += RainMistHeight;
                        RainMistParticleSystem.transform.position = pos;
                    }
                }
                else
                {
                    var s = RainFallParticleSystem.shape;
                    s.shapeType = ParticleSystemShapeType.Box;
                    if (RainMistParticleSystem != null)
                    {
                        var s2 = RainMistParticleSystem.shape;
                        s2.shapeType = ParticleSystemShapeType.Box;
                        Vector3 pos = RainFallParticleSystem.transform.position;
                        pos.y += RainMistHeight;
                        pos.y -= RainHeight;
                        RainMistParticleSystem.transform.position = pos;
                    }
                }
            }
        }
```

Фрагмент 9 – Функция за промяна на дъжда.

На Фрагмент 9 е показана функцията за промяна на дъжда. На Фрагменти 11, Фрагменти 12 и Фрагменти 13 са показани приложените логики за изменение на светлината, мъглата, и хлъзгавостта на пътя.

```
RenderSettings.sun.intensity += lightIntensityModifier;
```

Фрагмент 11 – Логика за изменение на светлината.

```
RenderSettings.fogDensity += fogModifier;
```

Фрагмент 12 – Логика за изменение на гъстотата на мъглата.

```
wheelFrictionCurve.stiffness += stiffnessModifier;
```

Фрагмент 13 – Логика за изменение на хлъзгавостта на пътя.

6.2.5. Реализация на модула за Адаптация

В този модул са заложени методите на адаптация на играта спрямо резултатите на играча. Първо ще разгледаме метода на адаптация по зададени нива как е имплементиран.

Първо се изчисляват резултатите на играча по следният начин:

```
float avgVelocity = (numberOfUpdates == 0) ? 0 : sumVelocity / numberOfUpdates;
                    double avgImpact = (numberOfCollisions > 0 ? impactOfCollisions
/ numberOfCollisions : 0) / 50;
                    double performance = Math.Round(avgVelocity /
(impactOfCollisions / 50 + 1), 2);
```

Фрагмент 14 – Логика за изчисление на резултатите.

След като вече знаем резултата на играча за даденият момент, спрямо него определяме стойностите на модификаторите които ще приложим върху компонентите от околната среда. Определянето на модификаторите става по следният начин:

```
if (performance < minimumPerformance){
                    fogModifier = -0.005f;
                lightIntensityModifier = 0.2f;
                    stiffnessModifier = 0.2f;
                }
                if (performance >= minimumPerformance && performance <=
mediumPerformance){
                    fogModifier = -0.005f;
                    lightIntensityModifier = 0.1f;
                    stiffnessModifier = 0.1f;
            }
                if (performance >= mediumPerformance && performance <= highPerformance)
{
                    fogModifier = 0.005f;
                    lightIntensityModifier = -0.1f;
                    stiffnessModifier = -0.1f;
            }
                if (performance > highPerformance){
                    fogModifier = 0.005f;
                lightIntensityModifier = -0.2f;
                    stiffnessModifier = -0.2f;
            }
```

Фрагмент 15 – Логика за определяне модификаторите.

След което се използва логиката за промяна на елементите от околната среда за да ги промени. Тази последователност се изпълнява на всеки фрейм.

Методът за адаптация на симулатора по крива на учене.

Първо се задават метрики които представляват самите криви на учене в нашият случаи те изглеждат така:

```
testMetricPattern = new PatternBasedAdaptationAsset();
                    FittingLine fittingLine1 = new FittingLine(0.8, 0.6);
                    testMetricPattern.RegisterPattern("Decrease AA 20% for 4s",
"Performance", PatternFeature.None, "t t+2000 t+4000", "x x*0.90 x*0.8",
fittingLine1);
                    testMetricPattern.RegisterPattern("Decrease AA 40% for 10s",
"Performance", PatternFeature.None, "t t+5000 t+10000", "x x*0.80 x*0.6",
fittingLine1);
                    testMetricPattern.RegisterPattern("Decrease AA 80% for 16s",
"Performance", PatternFeature.None, "t t+6000 t+12000", "x x*0.60 x*0.2",
fittingLine1);
                    testMetricPattern.RegisterPattern("Increase AA 20% for 10s",
"Performance", PatternFeature.None, "t t+5000 t+10000", "x x*1.10 x*1.2",
fittingLine1);
                    testMetricPattern.RegisterPattern("Increase AA 40% for 10s",
"Performance", PatternFeature.None, "t t+5000 t+10000", "x x*1.20 x*1.4",
fittingLine1);
                    testMetricPattern.RegisterPattern("Increase AA 80% for 10s",
"Performance", PatternFeature.None, "t t+5000 t+10000", "x x*1.40 x*1.8",
fittingLine1);
```

Фрагмент 16 – Инициализиране на Кривите на учене.

Резултатите на играча се взимат по абсолютно същият начин като и при метода на зададени нива. След като бъдат изчислени, биват сравнявани със дефинираните криви на учене за да предели модификаторите по които ще се променя околната среда.

```
List<string> patterns = testMetricPattern.SetMetricValue("Performance",
(float)performance, t);
```

Фрагмент 17 – Логика за изменение на хлъзгавостта на пътя.

Спрямо тези модификатори се извиква Event Handler на Player-centric rule-and-pattern-based adaptation asset компонента, който изпълнява логика за промяна на елементите от околната среда.

```
testMetricPattern.PatternEventHandler(patterns, gameObject);
```

Фрагмент 18 – Логика за извикване на Event Handler

6.2.6. Реализация на външния модула за уеб услугата

Този модул ще се занимава с получаването на данните и тяхното записване на сървъра на който се намира самата уеб услуга. Уеб услугата записва данните във файл с разширение CSV формат. На следният фрагмент може да се види как изглежда самата уеб услуга.

```
<?php
   $isFileExists = false;
   if(file_exists($_GET['filename']))
   {
      $isFileExists = true;
   }

   $logFile = fopen($_GET['filename'], "a") or die("Unable to open file!");
   if(!$isFileExists)
   {
      $txt = $_GET['columnNames'];
      fwrite($logFile, $txt);
      fwrite($logFile, "\n");
   }

   $txt = $_GET['logContent'];
   fwrite($logFile, $txt);
   fwrite($logFile, "\n");

   fclose($logFile);
   http_response_code(200);
?>
```

Фрагмент 19 – Имплементация на уеб услугата.

6.3. Планиране на тестването

Същинското тестване на системата преминава през няколко етапа с цел гарантиране на функционалните и нефункционалните изисквания към играта, които бяха описани и предварително зададени. Развитието на една игра преминава през много цикли от етапи. Софтуерът за игри е дефиниран, оценен, проектиран, построен, тестван, преминал през алфа версия, бета версия и накрая финалната версия. Цикълът на разработване от етапите често се описва последователно, но основните задачи често протичат паралелно, особено по време на развитие на играта. Тестването може да се изпълни на всеки етап от цикъла на проекта. Целта на цялостното тестване на играта е да се идентифицират и коригират грешките, възможно най-рано, така че продуктът накрая да е качествен и работещ. Компаниите, които разработват игри за масова употреба, разполагат с ресурси, подпомагащи тестването на техните продукти. Поради невъзможността да се преминат всички типове тестове и сценарии, са избрани тези, които пряко засягат симулаторът на управление на автомобил и в същото време е в рамките на възможното, като целта е да се покрият в максимална степен качествените изисквания на видеоиграта. Планират се следният тип тестове: Модулно тестване, Компонентно тестване, Системно тестване, Тестване за надеждност (Unit test) и накрая Тест за приемане (Aacceptance Test).

6.3.1. Реализация на външния модула за уеб услугата

Модулното тестване засяга взаимодействието на всеки един модул със останалите. Пример за това са тестове, които се отнасят до адаптивните модули на играта спрямо резултатите по време на игра. Модула отговарящ за изпращането на данните към уеб услуга по време на игровия процес. Модула на колата която върви по трасето. Модулът на звука който излиза при удара на колата в друг обект. Всяко едно такова тестване протича по време на разработката на играта, както и при цялостното и завършване.

6.3.2. Компонентно тестване

Този тест налага разбиването на играта на малки тестови компоненти. Тестването на играта означава проверка на всяка част от нея, като тези части включват:

- **Анимация –** успешното движение на автомобила по трасето, и падането на дъждовните капки, плискането на вода при минаването покрай локви (усещането за движение, реализъм, скорост на кадрите).
- **Аудио и звукови ефекти –** звуковите ефекти от околната среда, шума на двигателя на колата или при завиване.
- **Камерата**
- **Виртуалният свят и отделните сцени**
- **Инструментите на колата –** показване на скоростта и оставащото разстояние до края на текущата обиколка
- **Условията за успешно завършване на играта**
- **Опциите на играта** – Стартиране на играта, изход на играта, режимите на игра от гледна точка на това дали е на цял екран или в отделен прозорец, резолюцията с която играчът иска да играе играта, качеството на графиката, контролите за управление.
- **Специалните ефекти в играта**

6.3.3. Тест за надеждност (Unit Test)

Този тест ни гарантира, че най-малките елементи на играта работят според критериите на изискванията на играта. Това може да се отнася до определен компонент или конкретен механизъм на игратаq който може да бъде тестван и изолиран от останалите. Това обаче не означава че този механизъм или компонент е проектиран изолирано. Чрез този тест е проверен механизмите за адаптация на играта (механизъм за адаптиране чрез нива и механизъм за адаптиране чрез отриване кривата на учене). Освен този механизъм, чрез Unit test е проверен и механизма за изпращане на данните към уеб услугата.

6.3.4. Системно тестване

Системния тест се извършва когато играта се играе от самото начало до самия край. Първите системни тестове могат бъдат и симулирани, но в някакъв етап от разработването на играта ще е необходим да се създаде и прототип. Чрез прототипа ще трябва да преработим и да тестваме играта няколко пъти. Системния тест е шансът да отметнем колко се може повече критерии за качество, преди да поканим друг играч или играчи на за тест, за да се опитаме да симулираме възможно най-много различни ситуации и сценарии в които играта би могла да се провали. След като сме проверили всички критерии за качество на играта, сме готови за тестовете за приемане.

6.3.5. Тестване за приемане

Този тест отразява обикновена игрална сесия в която играчите играят играта. Този тест е тясно свързан с точка: Анализ на резултатите от тестването и начина на отразяването им.

6.4. Експериментално внедряване

Внедряването е процес при който активно си сътрудничат клиента и вече реализирания продукт. Един софтуерен продукт се счита за внедрен когато е инсталиран на необходимите места. Двете адаптивни версии и неадаптивната версия на играта

симулатор на управление на автомобил са разположени на сървър любезно предоставен от проф. д-р Боян Бончев за целите на дипломната работа от където всеки желаещ любител на игрите може да участва в експеримента.

3D видео игра за управление на автомобил (проект RAGE)

Добре дошли в страницата на експериментална 3D видео игра за управление на автомобил, създадена по проекта RAGE (Realising an Applied Gaming Ecosystem).

Целта на играта е управление на автомобил с минимален брой произшествия при максимално висока възможна скорост, съобразена с различни условия на шофиране. Играчът трябва да направи възможно най-бързо три обиколки на трасето:
- първата е при отлични условия на шофиране;
- втората е при мъгла и малко преди здрачаване;
- третата е при мъгла и малко преди здрачаване, при което вали дъжд и пътят е хлъзгав.

Скоростта се показва на велометъра, а прогресът на изминаване на текущата обиколка е представен от растящ зелен кръг около скоростомера. Няма наказателни точки за произшествия. Управлението на автомобила е с клавишите - стрелки. При евентуално буксуване и невъзможност за придвижване, използвайте бутон за табулация или ENTER.

Всеки участник в експеримента трябва да е изиграл играта поне веднъж, преди да премине към попълване на въпросник за идентифициране на качествата на видео играта. Участието е анонимно, като полето за e-mail служи единствено за евентуална връзка с участника в експеримента и получаване на резултата (времето за извършване на трите обиколки). Въпросникът е на адрес https://goo.gl/forms/9PBgFtlKzA3J0fiy1 и се попълва за не повече от 10 мин.

Предварително благодарим за отделеното време!

Играта може да се играе по два начина:
1) във Firefox браузър с инсталиран Unity Web Player Plugin (от адрес https://unity3d.com/webplayer, с цел играене в браузър) - отворете страницата CarGame.html и изиграйте една или повече игри до края;
2) локално с десктоп версията на играта - ZIP архив с десктоп версия на играта може да свалите оттук.

Фиг. 22 – Сайта с неадаптивната версия на играта

Адаптивна 3D видео игра за управление на автомобил

Добре дошли в страницата на експериментална адаптивна 3D видео игра за управление на автомобил!

Целта на играта е управление на автомобил с минимален брой произшествия при максимално висока възможна скорост, съобразена с различни условия на шофиране. Играчът трябва да направи възможно най-бързо три обиколки на трасето:
- първата е при отлични условия на шофиране;
- втората е при мъгла и малко преди здрачаване;
- третата е при мъгла и малко преди здрачаване, при което вали дъжд и пътят е хлъзгав.

Скоростта се показва на велометъра, а прогресът на изминаване на текущата обиколка е представен от растящ зелен кръг около скоростомера. Няма наказателни точки за произшествия. Управлението на автомобила е с клавишите - стрелки. При евентуално буксуване и невъзможност за придвижване, използвайте бутон за табулация или ENTER.

Всеки участник в експеримента трябва да е изиграл играта поне веднъж, преди да премине към попълване на въпросник за идентифициране на качествата на видео играта. Участието е анонимно, като полето за e-mail служи единствено за евентуална връзка с участника в експеримента и получаване на резултата (времето за извършване на трите обиколки). Въпросникът е на адрес https://goo.gl/forms/YzjnLI7GbDTSk2uw2 и се попълва за не повече от 10 мин.

Предварително благодарим за отделеното време!

Играта може да се играе локално с десктоп версията, като ZIP архив с тази десктоп версия може да свалите оттук.

Предварително благодарим за отделеното време!

Приятна игра и успех!

Фиг. 23 – Сайта с адаптивната версия на играта по метода на зададени нива

Фиг. 24 – Сайта с адаптивната версия на играта по метода на крива на учене

6.5. Анализ на резултатите

Анализът на резултатите от тестването, на различните методи за адаптация на сложността на симулатора на управление на автомобил, ще ни покажат кой от имплементираните методи е подходящ за имплементацията на сериозни игри от този тип чиято цел е да накара играчът да придобие нужните знания и опит чрез играене. За целите на тестване и изследване на методите за адаптация, бяха имплементирани няколко версии на една и съща игра с различни методи на адаптация. След различните проведени тестове (компонентно тестване, тест за надеждност, системно тестване, тестване за приемливост) за отстраняване на появилите се бъговете, игрите трябваше да бъдат тествани от различни хора (експериментално внедряване). Различните версии на играта се даваха на отделни група хора, като във всяка една от групите не трябваше да присъстват едни и същи хора. Причината за това е следната, ако един човек вече е играл една от версиите на играта, то той вече знае какво да очаква от играта и би се справил по добре при всеки следващ свои опит, и това ще доведе до невалидни резултати за изследване, което от своя страна ще направи изводите на това изследване невалидни. След като даден човек свърши с играта, той трябва да попълни въпросник (Game Experience Questionnaire (GEQ), (Nacke, 2009)) [28], който има за цел да постави оценка за умение, потапяне, поток, напрежение, предизвикателство и положително / отрицателно въздействие, почувствано от играча по време на играта. Въпросникът се

състои от 46 въпроса. За целта на експеримента всяка една от версиите на играта беше дадена на различни хора, а след играта всеки един от тях е попълнил въпросникът.

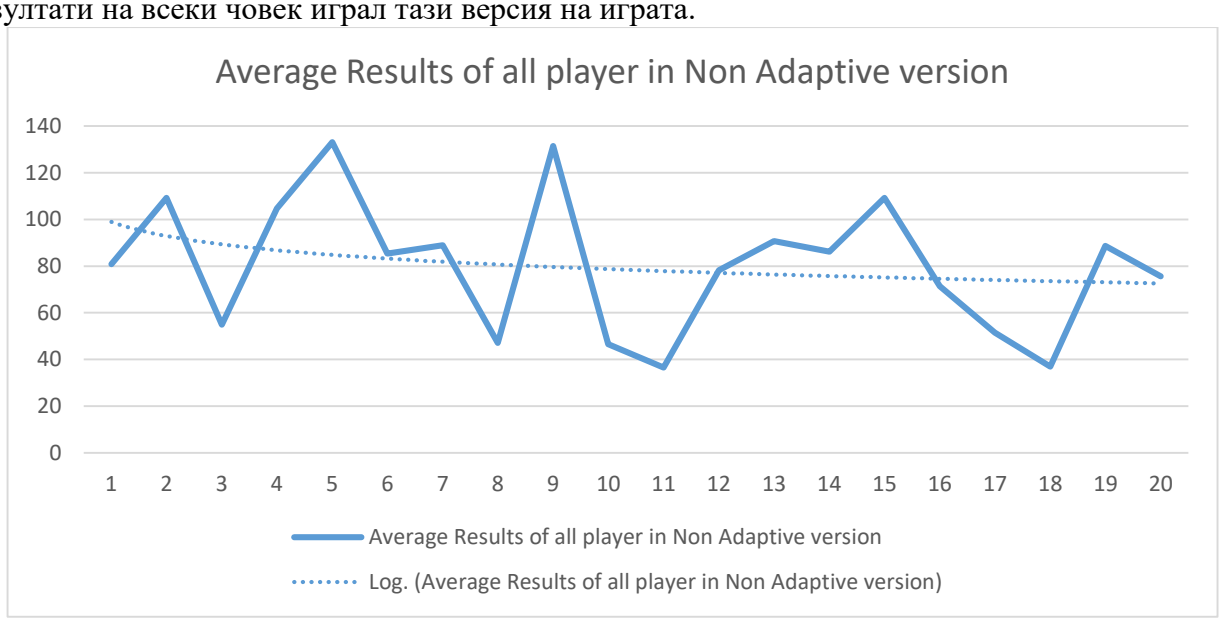
В тази точка ще разгледаме и анализираме по отделно получените резултати от всяка версия на играта и съответните резултати от въпросниците към всяка версия. Разглеждането и анализирането на резултатите възпроизведени от играта става по следният начин: От дадена версия на играта се взимат файловете със записаните резултати на играчите играли тази версия. От всеки файл се взима средната стойност на представянето на играча по време на цялата игра. Представянето на играча се изчислява по следната формула:

Performance = AvgVel/(ImpCol/50 +1), където AvgVel е средната стойност на ускорението на колата за даденият момент, а ImpCol е въздействието на колизията, ако съществува такава (ако не съществува се получава 0/50 за това събираме с едно), за даденият момент. Оценката на даден играч правим като вземем средната стойност на Performance на играча през цялото му време на игра. След като вземем средните стойности от всички играчи играли дадената версия, изчисляваме спрямо тях средната стойност, по която ще сравним всичките версии на играта. По големите стойности означават по добри постигнати резултати спрямо дадения метод на адаптация.

Резултатите възпроизведени от въпросниците за дадена версия на симулатора се анализират по следният начин: За всеки въпрос от въпросника на дадена версия от играта се гледа колко процента е бил отговорена дадена опция от дадения въпрос. Ако даден отговор има най-много пъти избиран от анкетираните, него ще считаме за абсолютен отговор, на този въпрос.

6.5.1. Резултати от неадаптивната версия на симулатора на управление на автомобил и въпросникът към нея

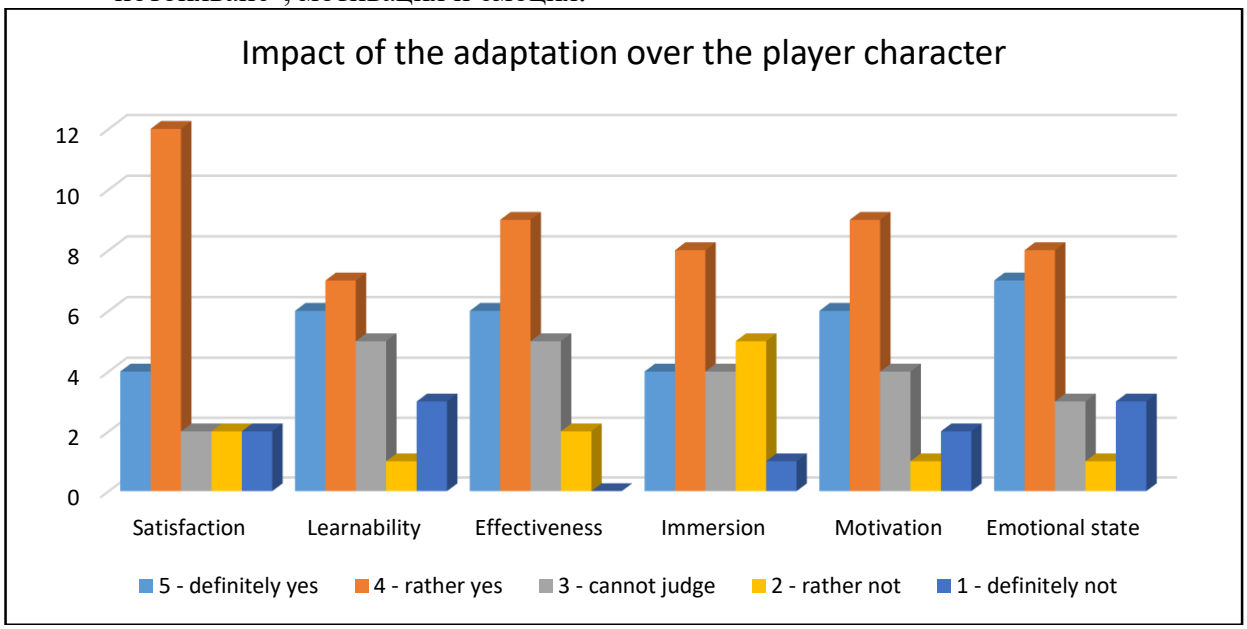
Средната стойност от всички средни стойности на представяне на играчите играли тази версия на симулатора е 80.35267479. Долу на фиг. 25 са показани средните резултати на всеки човек играл тази версия на играта.



Фиг. 25 – средните стойности от резултатите на играчите от неадаптивната версия на играта

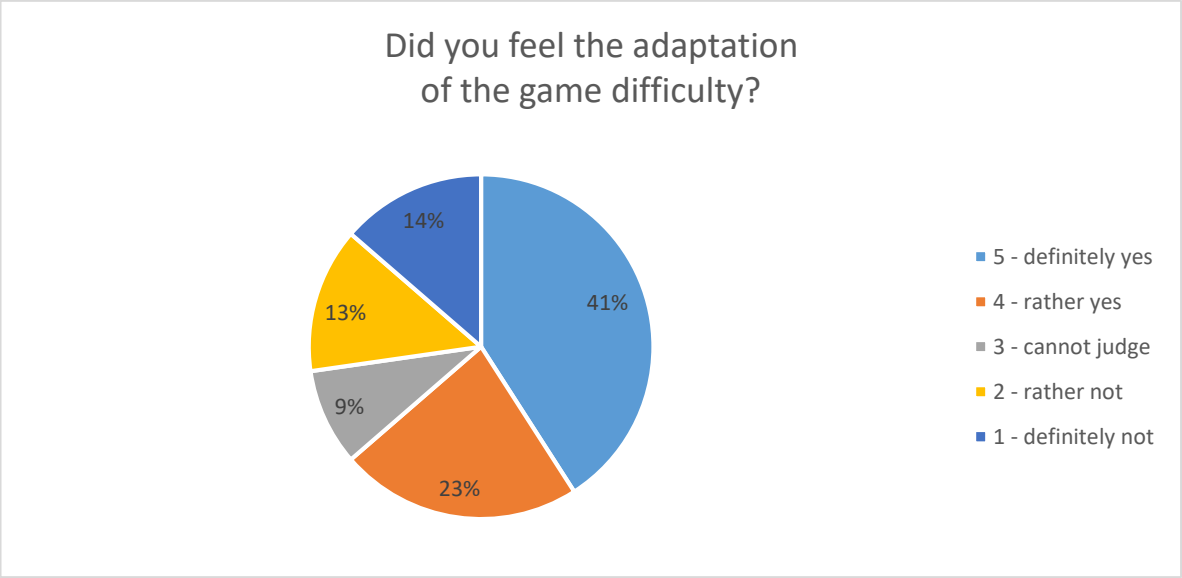
Резултатите от въпросникът на тази версия са следните:

- По време на играта повечето от играчите не са били доволни.
- Повечето играчи са се чувствали способни да преодолеят предизвикателствата.
- Повечето играчи са намирали сценарият на играта за скучен.
- Повечето от играчите са се забавлявали от своите грешки.
- Играчите не са се чувствали погълнати от играта.
- По време на играта повечето играчи са се чувствали безразлично.
- Напрежението на играчите по време на видеоиграта е било минимално за повечето от тях.
- Повечето смятат че не са научили нищо по време на играта.
- По-голяма част от играчите намират играта за скучна и са били отегчени.
- Докато са играели повечето хора не са били достатъчно погълнати от играта за да забравят за всичко ставащо около тях.
- Повечето от играчите не са изпитвали чувство на гняв по време на играта.
- Хората са нямали стимул докато играят играта.
- Много от хората не са усетили трудностите в играта и не са придобили почти никакъв опит.
- Всички играчи не са усетили адаптиране на трудността на играта.
- На фиг. 30 е отразено влиянието на адаптацията върху самият играч, спрямо това какво удоволствие е изпитал, възможности за учене, ефективност, "потопяване", мотивация и емоция.



Фиг. 30 – влиянието на адаптацията върху играчите от играта без адаптация

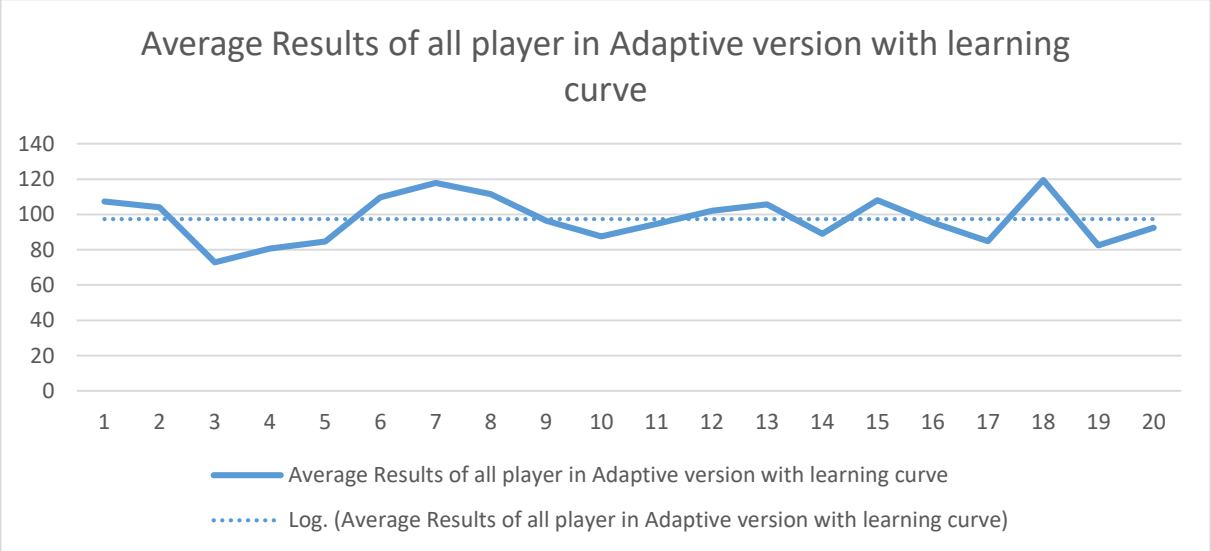
- Повечето хора са усетили даптацията в играта, което и показва фиг. 29



Фиг. 29 – отговорите на хората по отношение на адаптацията за играта без адаптация

6.5.2. Резултати от адаптивната версия на симулатора на управление на автомобил спрямо криви на учене

Средната стойност от всички средни стойности на представяне на играчите играли тази версия на симулатора е 97.28819871. Долу на фиг. 26 са показани средните резултати на всеки човек играл тази версия на играта.

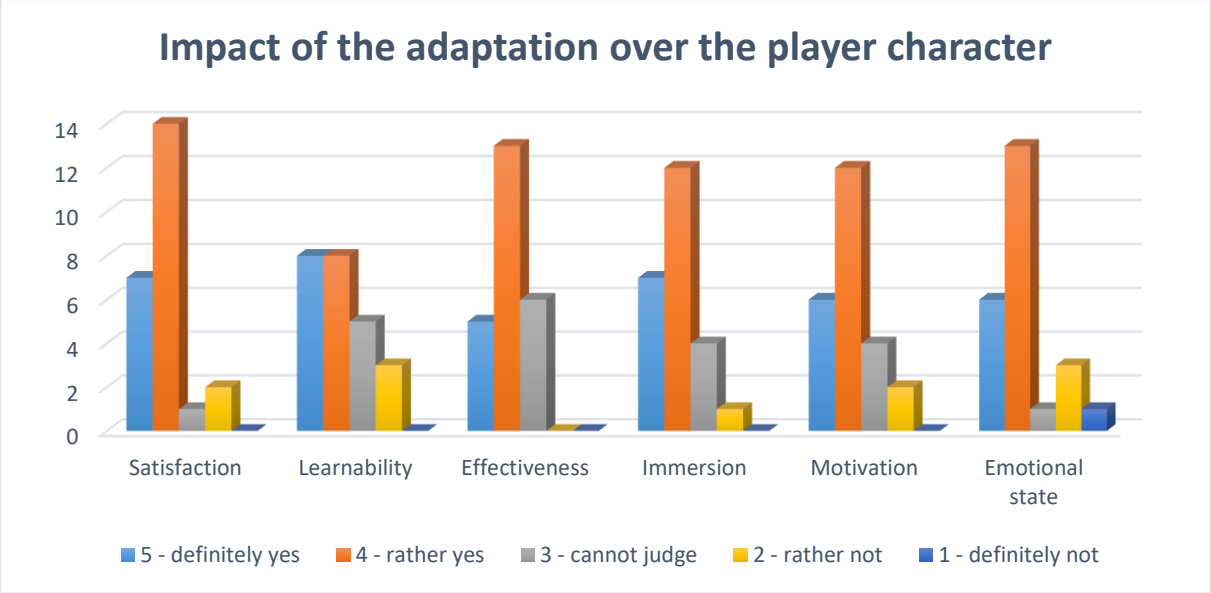


Фиг. 26 – средните стойности от резултатите на играчите от адаптивната версия на играта с криви на учене

Резултатите от въпросникът на тази версия са следните:

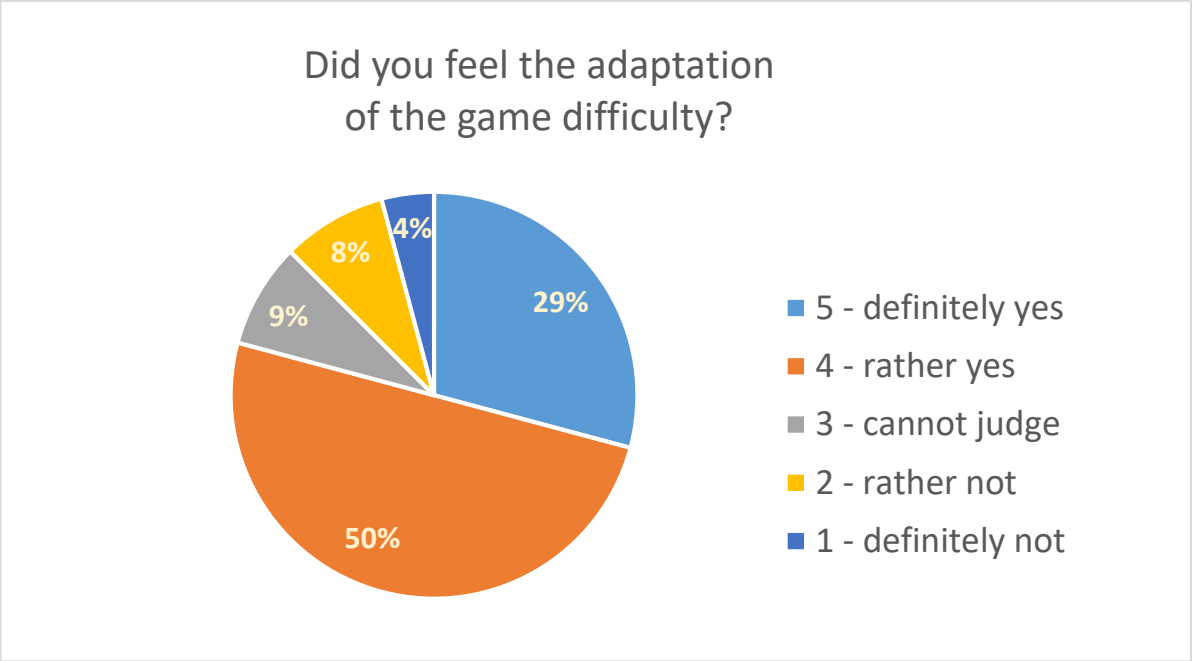
- По време на играта повечето от играчите са били напълно доволни.
- Повечето играчи са се чувствали истински способни да преодолеят предизвикателствата.
- Повечето играчи са намирали сценарият на играта за доста интересен.
- Повечето от играчите са се забавлявали от своите грешки.

- Играчите са били абсолютно погълнати от играта.
- По време на играта повечето играчи са се чувствали щастливи.
- Напрежението на играчите по време на видеоиграта е било нормално за повечето от тях.
- Повечето смятат че са научили нещо по време на играта.
- По-голяма част от играчите намират играта за интересна и не са били отегчени.
- Докато са играели повечето хора са били погълнати от играта, до толкова че да забравят за всичко ставащо около тях.
- Повечето от играчите не са изпитвали чувство на гняв по време на играта.
- Хората са били доста стимулирани по време на играта.
- Всички са усетили трудностите в играта .
- Всички играчи са усетили адаптиране на трудността на играта.
- На фиг. 32 е отразено влиянието на адаптацията върху самият играч, спрямо това какво удоволствие е изпитал, възможности за учене, ефективност, "потопяване", мотивация и емоция.



Фиг. 32 – влиянието на адаптацията върху играчите от играта с криви на учене

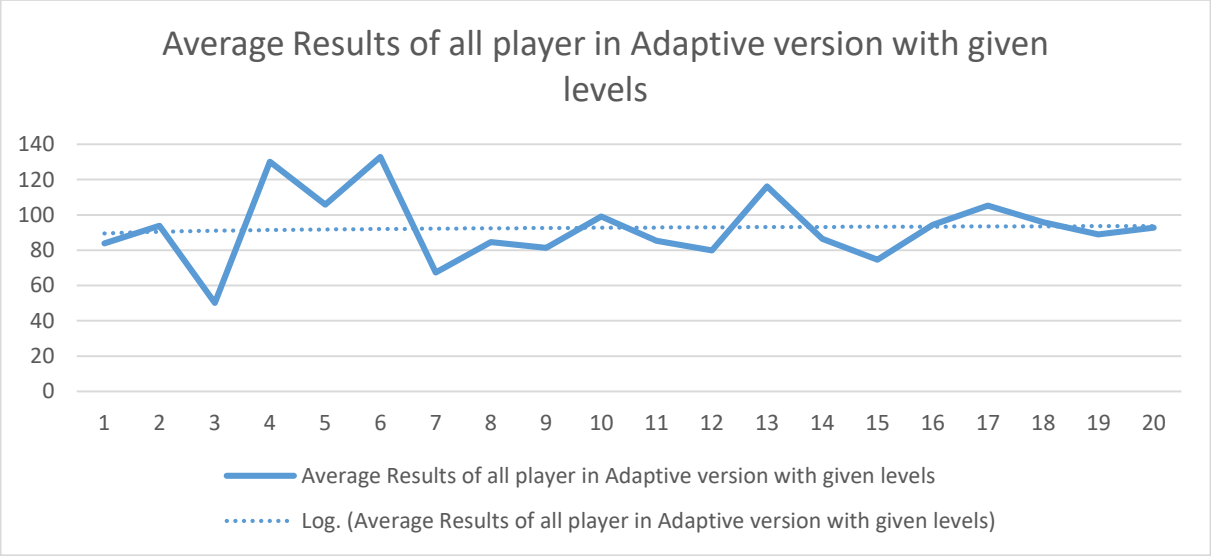
- По-голяма част от хората са усетили адаптацията, това и показва фиг. 31



Фиг. 31 – отговорите на хората по отношение на адаптацията за играта с адаптация спрямо кривата на учене

6.5.3. Резултати от адаптивната версия на симулатора на управление на автомобил спрямо зададени нива

Средната стойност от всички средни стойности на представяне на играчите играли тази версия на симулатора е 92.46765047. Долу на фиг. 27 са показани средните резултати на всеки човек играл тази версия на играта.

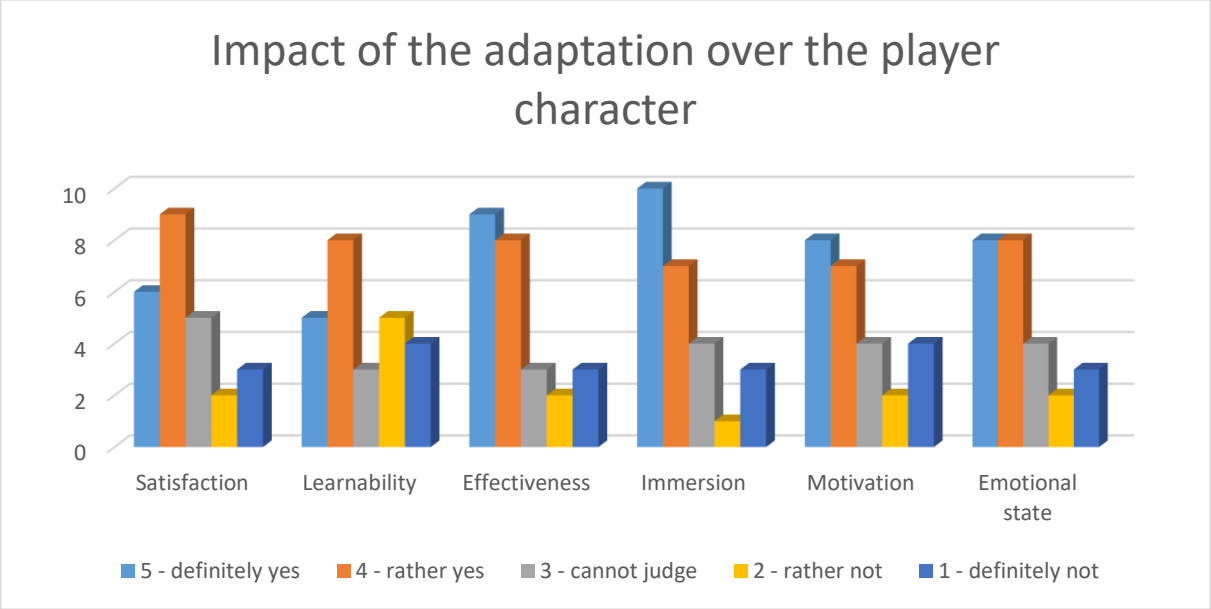


Фиг. 27 – средните стойности от резултатите на играчите от адаптивната версия на играта със зададени нива

Резултатите от въпросникът на тази версия са следните:

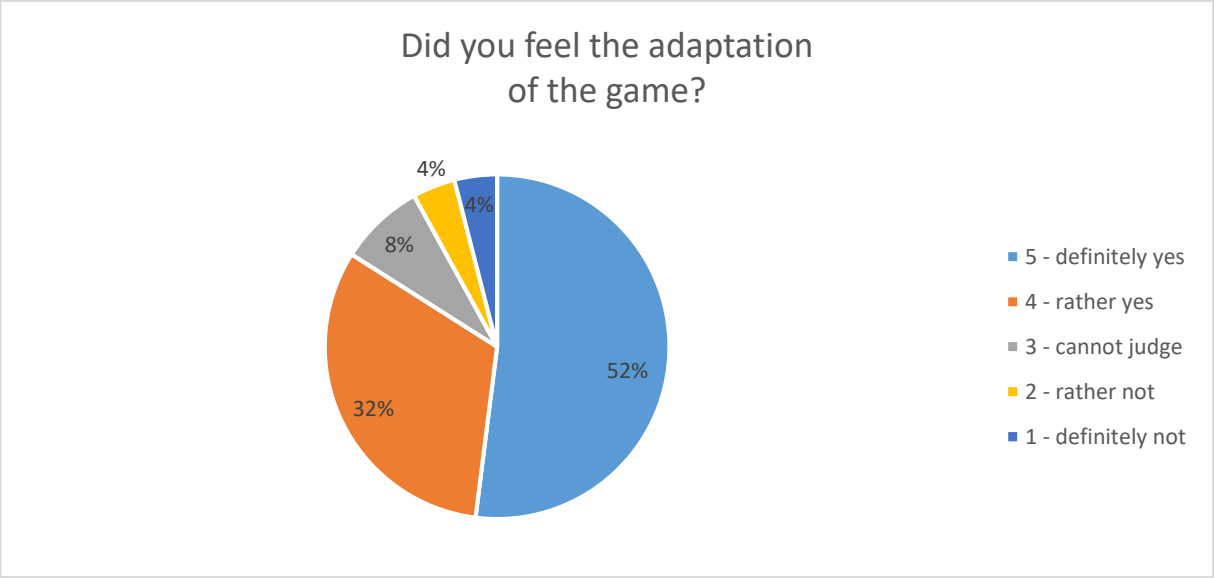
- По време на играта повечето от играчите са били доста доволни.
- Повечето играчи са се чувствали доста способни да преодолеят предизвикателствата.

- Повечето играчи са намирали сценарият на играта за интересен.
- Повечето от играчите са се забавлявали от своите грешки.
- Играчите са били напълно погълнати от играта.
- По време на играта повечето играчи са се чувствали щастливи.
- Напрежението на играчите по време на видеоиграта е било нормално за повечето от тях.
- Повечето не могат да преценят дали са научили нещо по време на играта.
- По-голяма част от играчите намират играта за интересна и не са били отегчени.
- Докато са играели повечето хора са били нормално погълнати от играта, но не чак толкова че да забравят за всичко ставащо около тях.
- Повечето от играчите са се ядосвали леко по време на играта.
- Хората са били стимулирани по време на играта.
- Всички са усетили трудностите в играта .
- Всички играчи са усетили адаптиране на трудността на играта.
- На фиг. 34 е отразено влиянието на адаптацията върху самият играч, спрямо това какво удоволствие е изпитал, възможности за учене, ефективност, "потопяване", мотивация и емоция.



Фиг. 34 – влиянието на адаптацията върху играчите от играта спрямо зададени нива

- Почти всички от хората са усетили адаптацията, това и показва фиг. 33



Фиг. 33 – отговорите на хората по отношение на адаптацията за играта с адаптация спрямо зададени нива

6.5.4. Оценка на методите за адаптация на видеоигри

Според резултатите от версиите на игрите и на въпросниците можем да направим следните изводи за нашето изследване на методите за адаптиране на симулатори спрямо кривата на учене:

Най-подходящият метод за адаптация на сложността на видеоигри е по метода динамичната адаптация спрямо кривите на учене. Докато метода за адаптация по зададени нива също показва добри резултати, не е толкова ефективен и ефикасен колкото метода за адаптация спрямо кривите на учене.

А за достоверността на данните получени от анкетите е използван тестът на Стюдънт [37]. Получените данни от трите версии на игрите от въпросниците бяха сравнени помежду си две по две (резултати от неадаптивна версия с резултати от адаптивна версия спрямо зададени нива, резултати от неадаптивна версия с адаптивна версия спрямо крива на учене и резултати от адаптивна версия спрямо зададени нива с адаптивна версия спрямо крива на учене). На следните фигури са показани и резултатите.

	Variable 1	*Variable 2*
Mean	3.290232558	2.956659619
Variance	0.530216611	0.185257132
Observations	43	43
Hypothesized Mean Difference	0	
df	68	
t Stat	2.585998023	
P(T<=t) one-tail	0.005928	
t Critical one-tail	1.667572281	
P(T<=t) two-tail	0.011856	
t Critical two-tail	1.995468931	

Фиг. 35 – резултати от теста Стюдънт спрямо неадаптивната версия и адаптивната версия със зададени нива

	Variable 1	*Variable 2*
Mean	2.956659619	3.130813953
Variance	0.185257132	0.517905208
Observations	43	43
Hypothesized Mean Difference	0	
df	69	
t Stat	-1.361885868	
P(T<=t) one-tail	0.088831917	
t Critical one-tail	1.667238549	
P(T<=t) two-tail	0.177663835	
t Critical two-tail	1.994945415	

Фиг. 36 – резултати от теста Стюдънт спрямо неадаптивната версия и адаптивната версия със крива на учене

	Variable 1	*Variable 2*
Mean	3.290232558	3.130813953
Variance	0.530216611	0.517905208
Observations	43	43
Hypothesized Mean Difference	0	
df	84	
t Stat	1.021097823	
P(T<=t) one-tail	0.155069963	
t Critical one-tail	1.663196679	
P(T<=t) two-tail	0.310139927	
t Critical two-tail	1.988609667	

Фиг. 37 – резултати от теста Стюдънт спрямо адаптивната версия със зададени нива и адаптивната версия със крива на учене

Спрямо теста на Стюдът [37]за да отхвърлим нулевата хипотеза е да докажем вярността ѝ следното условие трябва да е изпълнено: t Stat < -(t Critical two-tail) || t Stat > t Critical two-tail и P(T<=t) two-tail < 0.05 [36] [37]. По така определените критерии и резултати може да се заключи че резултатите, от неадаптивната версия и адаптивната версия със зададени нива, един спрямо друг са достоверни и може да им се вярва. Докато резултатите от неадаптивната версия и адаптивната версия със крива на учене и адаптивната версия със зададени нива и адаптивната версия със крива на учене са недостоверни.

7. Заключение

7.1. Обобщение на изпълнението на началните цели

В обобщение на поставените цели пред настоящата дипломна работа трябва да обърнем внимание върху ключовите функционалности, които биха превърнали предмета на настоящата дипломна работа в работещ продукт. В днешно време има изключително много игри от най-разнообразни жанрове но голяма част от тях не предлагат адаптивност на сложността на играта нито по зададени нива нито по криви на учене, при тях нивото на сложност се задава още в началото. Поради тази причина настоящата дипломна работа, цели да покаже че човек сядайки пред компютъра пред играта може да усвои нови знания и умения.

Дипловната работа е разработени в рамките на научно-изследователския проект APOGEE (http://apogee.online), който планира да създаде решение, което от една страна да запълни липсата на проста, но ефективна софтуерна платформа с отворен код за лесно изграждане на образователни видеоигри от ИТ специалисти, а от друга страна – да отговори на нуждите от огромно количество специализирани адаптивни видеоигри за обучение по различни учебни дисциплини.

По време на проучванията на технологичните решения при разработването на подобен тип игра, бяха разгледани и сравнени предимствата и недостатъците на най-популярните към момента платформи за разработка на игри. На базата на тези сравнения беше избрана платформата Unity за разработката на играта стрелец от първо лице.

На следващия етап от анализа на събраните изисквания бяха определени ключовите функционални и нефункционални изисквания. Също така беше подробно дефиниран потокът на играта при взаимодействието между играта и играчът.

Етапът проектиране на играта се вземат важни решения относно архитектурата на софтуера, както и нефункционалните изисквания към него. Тук се проектират връзките и комуникацията с различни модули и външни услуги, чрез декомпозиция на модули и под модули. Дефинират се логическото разделение на модули и под модули за разработка, като по този начин се обособяват основните и най-важни функционалности на играта.

Етапът за реализация е един от особено важните, тъй като без него няма реализиран продукт. Тук е моментът за осъществяване на направените технологични проучвания, анализираните изисквания и проектираната архитектура. На тази фаза се разискват и съответно имплементират алгоритми, заимстват се добри практики, интегрират се методите за адаптация в играта. Определят се стандартите за писане на код, които целят да направят кода на играта разбираем и лесен за разширение и поддръжка.

На етапа тестване която е неизменна част от разработката на играта се определят различните типове и нива на тестване, които да гарантират функционалните и нефункционалните изисквания към играта.

Получените резултати от тестването ще бъдат публикувани в статия за международна научна конференция.

7.1. Насоки за бъдещо развитие и усъвършенстване

Играта симулатор за управление на автомобил е игра, която отчасти влиза в кръга „Сериозни игри“. Голяма част от хората не са чували за подобен тип игри а тези които са чували са слабо запознати. Причината за това е липсата на информация и разбира се финансови средства за разработка на подобен тип игри. Запознавайки се понятието

„Сериозни игри“ почти всички участници заявяват, че според тях сериозните игри са подходящи за обучение и биха могли да се приложат в области на обучение където практическото обучение в реални условия е свързано с рискове, и че би им било интересно да играят такива игри и в бъдеще и че този тип игри биха дали по – добри резултати в сравнение с традиционните инструменти за обучение.

Използвана литература

[1] Best sim for realism. Besides iRacing.
https://www.reddit.com/r/simracing/comments/6hafk8/best_sim_for_realism_besides_iracing/
Свалено от *https://www.reddit.com.*

[2] проф. Д-р Боян Бончев. *Design of computer video games – User Interface and experience*

[3] проф. Д-р Боян Бончев. *Design of computer video games – Game Development*

[4] Десислава Василева. *Realizing and Applied Gaming Ecosystem*

[5] rFactor
Свалено от *http://www.rfactor.net/*

[6] Steampowered rFactor_2
Свалено от *https://store.steampowered.com/app/365960/rFactor_2/*

[7] Assetto Corsa
Свалено от *https://www.assettocorsa.net/competizione/*

[8] Steampowered Assetto Corsa Competizione
Свалено от *https://store.steampowered.com/app/805550/Assetto_Corsa_Competizione/*

[9] X-Plane
Свалено от *https://www.x-plane.com/kb/x-plane-11-system-requirements/*

[10] Steampowered XPlane 11
Свалено от *https://store.steampowered.com/app/269950/XPlane_11/*

[11] IRacing
Свалено от *https://www.iracing.com*

[12] Euro Truck Simulator 2
Свалено от *https://eurotrucksimulator2.com/*

[13] Euro Truck Simulator 2 *https://en.wikipedia.org/wiki/Euro_Truck_Simulator_2*
Свалено от *https://en.wikipedia.org/*

[14] Steampowered Euro Truck Simulator 2
Свалено от *https://store.steampowered.com/app/227300/Euro_Truck_Simulator_2/*

[15] Dynamic Player Modelling a Framework for Player-centred Digital Games
https://www.researchgate.net/publication/251860100_Dynamic_Player_Modelling_A_Framework_for_Player-centred_Digital_Games
Свалено от *https://www.researchgate.net/*

[16] Pros and cons of learning to drive in a simulator *https://medium.com/@rmgrunis/pros-and-cons-of-learning-to-drive-in-a-simulator-a73b7e90c4f5*
Свалено от *https://medium.com/*

[17] The pros and cons of web services *http://doveltech.com/innovation/the-pros-and-cons-of-web-services/*
Свалено от *http://doveltech.com/*

[18] The game engine modules *https://www.researchgate.net/figure/The-game-engine-modules-9_fig1_228849435*
Свалено от *https://www.researchgate.net/*

[19] проф. Д-р Боян Бончев, Десислава Василева. *Detection of player learning curve in a car driving game*

[20] Constructor. *https://www.scirra.com/*
Свалено от *https://www.scirra.com/*

[21] Game Maker Studio. *https://www.yoyogames.com/gamemaker*
Свалено от *https://www.yoyogames.com*

[22] Unity. *https://unity3d.com/*
Свалено от *https://unity3d.com/*

[23] Unreal Engine. *https://www.unrealengine.com/en-US/what-is-unreal-engine-4*
Свалено от *https://www.unrealengine.com/*

[24] Named Pipes. *https://docs.microsoft.com/en-us/dotnet/standard/io/how-to-use-named-pipes-for-network-interprocess-communication*
Свалено от *https://docs.microsoft.com*

[25] Sockets. *https://docs.microsoft.com/en-us/dotnet/framework/network-programming/how-to-create-a-socket*
Свалено от *https://docs.microsoft.com*

[26] Web Services *http://doveltech.com/innovation/the-pros-and-cons-of-web-services/*
Свалено от *http://doveltech.com*

[27] Understanding Game Architecture. *https://www.studytonight.com/3d-game-engineering-with-unity/game-development-architecture*
Свалено от *https://www.studytonight.com*

[28] проф. Д-р Боян Бончев. *Adapt Times Player-Centric Adaptation in Video games for education*

[29] Cite Bontchev, B. Adaptation in Affective Video Games: a Literature Review, *Journal of Cybernetics and Information Technologies*, Vol. 16, No.3, 2016, pp.3-34, DOI: 10.1515/cait-2016-0032

[30] проф. Д-р Боян Бончев. *Design of computer video games – Introduction to computer games*

[31] Bontchev, B., Vassileva, D. (2018) Dynamic game adaptation based on detection of behavioral patterns in the player learning curve, Proc. of the 10th Annual Int. Conf. on Education and New Learning Technologies, Palma de Mallorca (Spain), July 2018, ISBN: 978-84-09-02709-5.

[32] Bontchev, B., Vassileva, D. (2018) Detection of player learning curve in a car driving game, Proc. of 12thAnnual Int. Technology, Education and Development Conference (INTED'2018), March 2018, IATED, Valencia, Spain, ISBN: 978-84-697-9480-7, pp.9302-9311.

[33] Bontchev, B., Vassileva, D., Ivanov, D. (2018) Player-centric adaptation of a car driving video game, Proc. of e-Society'18 Int. Conf., April 2018, IADIS, Lisbon, Portugal, ISBN: 978-989-8533-75-3, pp.193-200.

[34] Bontchev, B. P., & Vassileva, D. (2017). Affect-based adaptation of an applied video game for educational purposes. *Interactive Technology and Smart Education*, Emerald, ISSN: 1741-5659, 14 (1), pp.31-49. DOI: 10.1108/ITSE-07-2016-0023

[35] Bontchev, B. Adaptation in Affective Video Games: a Literature Review, *Journal of Cybernetics and Information Technologies*, Vol. 16, No.3, 2016, pp.3-34, DOI: 10.1515/cait-2016-0032

[36] t-Test. *https://www.excel-easy.com/examples/t-test.html*
Свалено от *https://www.excel-easy.com/*

[37] Student's t-test. *https://en.wikipedia.org/wiki/Student%27s_t-test*
Свалено от *https://en.wikipedia.org/*