Софийски университет „Св. Климент Охридски“
Факултет по математика и информатика

Петър Русланов Армянов

Файлов формат за съхраняване на мултимедийна информация - поточен пренос на векторна анимация

АВТОРЕФЕРАТ

на дисертационен труд

за присъждане на образователната и научна степен „Доктор“
в професионално направление
4.6 „Информатика и компютърни науки“,
научна специалност: 01.01.12 „Информатика“

Научен ръководител:
доц. д-р Павел Бойчев, ФМИ, СУ

София 2016 г.

Дисертационният труд е обсъден и насочен за защита на разширено заседание на катедра „Компютърна Информатика“ към Факултета по математика и информатика на Софийски университет „Св. Климент Охридски“, състояло се на 22.06.2016 г.

Авторът е асистент и докторант в свободна форма на обучение във Факултета по математика и информатика на Софийски университет „Св. Климент Охридски“.

Дисертационният труд е изложен на 152 страници, в които се съдържат 23 фигури, 18 таблици, 6 страници литература, включващи 99 заглавия, от които едно на български език, 73 на английски език и 25 интернет източника. Има 2 приложения, представени в 3 страници. Списъкът от публикации на автора по същността на дисертацията включва 4 заглавия.

Публичната защита на дисертационния труд ще се състои на открито заседание на …… г. от …… часа в …… Материалите по защитата са на разположение в …….. на ФМИ към СУ (София, бул. Джеймс Баучър №5).

Съдържание

Индекс на фигурите

Индекс на таблиците

Обща характеристика на дисертационния труд

Актуалност на проблема и мотивация

През последните години векторната графика набира все по-голяма популярност. Тя е в основата на повечето инженерни системи, на географските информационни системи и системите за триизмерно моделиране. Голямо нейно предимство е практически неограничената резолюция и възможност за увеличение и фокусиране на произволен детайл от изображението, както и възможността лесно да бъде възпроизвеждана с променливо ниво на детайлност.

Най-широко приложение векторната графика намира при изобразяването на тримерни обекти – при инженерното проектиране; визуализацията на математически или бизнес модели; моделирането на физични или природни явления; в сферата на развлекателния бизнес – при разработване на игри и филмови ефекти или дори на цели филми. Натрупващите популярност системи за виртуална и добавена реалност също използват различни технологии от векторната графика. В последно време тримерната визуализация навлиза и в интерактивното и дистанционното обучение, тъй като предлага възможност за изграждане на реалистични, детайлни и интерактивни модели на различни обекти от реалния свят, трудни или невъзможни за прецизно заснемане [1, 2].

Едно от неудобствата на векторната графика е необходимостта изображенията да бъдат преобразувани в растерна форма за да се визуализират на стандартен компютърен или телевизионен екран. Този проблем се адресира в съвременните компютърни видео контролери, чиито драйвери поддържат възможност за директно бързо визуализиране на цели сцени, изградени от графични примитиви. В настоящия момент не само компютрите, но и голяма част от мобилните устройства, като таблети и смартфони, имат вградени графични процесори, специално проектирани за обработка и най-вече визуализиране на векторна графика [3]. Новост са системите за виртуална реалност, които често са построени върху интерактивна векторна графика, използваща стерео изобразяване с цел постигане на реалистичен тримерен ефект.

Векторната графика навлиза все по-широко и в уеб пространството посредством поддръжката на изображения във векторни формати като SVG [4], или директно в новия HTML5 canvas [5]. Използването на тези формати е възможно чрез различни технологии като WebGL или X3D [6].

Всички тези технологии позволяват лесно визуализиране от страна на клиентското устройство на тримерна графика, с възможност за интерактивна промяна на сцената или параметрите на визуализиране. Те обаче не поддържат директно интерактивно управление на изображението на страната на клиента, а още по-малко поточно предаване на анимация. Възможна е реализация на клиентски скриптове, които комуникират със

сървъра и съответно възпроизвеждат определени трансформации, но към момента няма общоприет стандарт за такива приложения.

Най-голямо потребление на контролирана от сървъра векторна анимация има в масовите мултиплейърни онлайн игри (Massively multiplayer online role-playing games – MMORPG), където най-често има двупосочна комуникация и специализиран формат за пренос на данните [7]. При този тип решения, обикновено векторното представяне на обектите е предварително заредено чрез инсталация на програма при клиента [8]. При игрите често има и трети лица – наблюдатели, които следят развитието на играта отстрани като анимация, без да могат да ѝ влияят. При тези решения изображенията се доставят до наблюдателите вече преобразувани в растерен клип, което не позволява промяна на гледната точка или фокусиране над избран детайл от играта.

Подобни проблеми съществуват и при използването на потоци от векторна анимация за целите на дистанционното обучение. В настоящия момент игровата индустрия е значително по-развита в технологично отношение от обучителната. По тази причина все по-често идеи от игрите се използват при разработване на учебни програми и материали [9]. Това неминуемо води до взаимстване на подходи и техники от различни популярни игри при изготвяне на по-достъпно и атрактивно за обучаващите се съдържание [10], включително тримерна анимация.

Обект и предмет на разработката

Обект на настоящото изследване е поточното пренасяне на мултимедия и по-конкретно на векторна анимация, съпроводена с други медии – звук, текст и други.

Предмет на изследването е създаване на модел, детайлно описание и реализация на формат, позволяващ пренос на векторна анимация в споделен поток, успоредно с други медии, както и на алгоритми, дефиниращи поведението на модулите, участващи в пренасянето.

Цели и задачи на дисертационния труд

Целите, поставени пред дисертационния труд, са дефиниране на изискванията към формат за поточен пренос на векторно представена анимация, който да позволява разширяване с други типове медии. Изисква се да се опише детайлно такъв формат и алгоритми за работата на сървър и клиент за него. Извън обхвата на работата остава разработката на цялостни клиент и сървър, поддържащи формата. Прототип на такива ще бъде разработен с цел доказване на приложимостта на разработеното решение и емпирично измерване на параметрите на получения поток данни.

Задачите, които произлизат от така поставените цели са:

1. Изследване на спецификите на растерното и векторното представяне на изображения и видео, както и начините за тяхното съхранение и поточно предаване;

2. Изследване на съществуващи решения за пренасяне на векторна анимация, техните силни и слаби страни;
3. Формулиране на изискванията към формат за поточен пренос на графика и по-конкретно на векторна анимация;
4. Дефиниране и реализация на формат за поточен пренос на анимирана векторна графика, позволяващ вграждане и на други типове медии;
5. Описание на алгоритмите на работа на сървър и клиент, поддържащи този формат;
6. Изготвяне на тестови сценарии и прототипна тестова система за валидиране на разработката.

Структура и съдържание на дисертационния труд

Дисертационният труд е с общ обем 152 страници. Състои се от увод, изложение в 5 глави, заключение, списък с цитираната литература и 2 приложения. Текстът е описан на 149 страници, в които се съдържат 23 фигури, 18 таблици и 6 страници литература, включващи 99 заглавия, от които едно на български език, 73 на английски език и 25 източника от Интернет. Приложенията са 2 и са представени в 3 страници. Списъкът от публикации на автора по същността на дисертацията включва 4 заглавия.

Глава 1. Обзор

В първа глава е направено въведение в същността на предметната област и са обяснени основните термини. Посочени са съществуващи към момента решения на поставените задачи и е направен анализ на силните и слабите им страни. Резултатите от анализа са обобщени в сравнителна таблица. Като извод от анализа са описани ключовите характеристики на разработваното от автора решение. Принос на автора, постигнат в тази глава, е задълбоченият преглед, обзор и анализ на съществуващи решения на поставените проблеми, както и оценката на техните предимства и недостатъци, въз основа на които са дефинирани изисквания към решението – цел на дисертационния труд.

Мултимедия и мултимедийни формати

Понятието *мултимедия* означава смесено използване на различни видове медии – текст, графика, звук, анимация, възпроизвеждани от компютър с предоставена възможност за интерактивно въздействие на възпроизвеждането. В настоящия момент мултимедията се използва практически във всички аспекти на приложение на компютрите. Освен традиционно свързваната с мултимедия сфера на развлеченията (филми, игри) голямата изразителност на мултимедийното съдържание се използва за нагледно представяне на информация – в обучението; в научната или техническа дейност; в бизнеса [1, 11].

Основен дял в мултимедията заемат визуалните компоненти – видео и анимация. Видеото най-често е заснет предварително с камера материал или излъчване на живо. От своя страна анимацията представлява последователност от нарисувани или генерирани изображения, всяко от които е с определена разлика спрямо предното.

Въпросът за записването, съхранението и пренасянето на мултимедийното съдържание е важна част от сферата на информационните и комуникационни технологии. Разработени са стотици формати, предназначени да съхраняват различен тип мултимедийно съдържание. Те основно се делят на два типа – *медийни формати* и *контейнерни формати*.

Медийните формати са предназначени за представяне и съхраняване на един тип медия, така че да може да бъде възпроизведена в някакъв по-късен момент. Контейнерните формати имат задачата да обединят множество медии, записани в някакви медийни формати, в единно свързано цяло – най-често файл. Така се реализира представяне на мултимедия.

Растерни и векторни изображения. Особености на векторната графика

Възпроизвеждането на различни компютърни изображения най-често представлява визуализирането им на екран или подобно устройство, обикновено представляващо правоъгълна матрица от точки. Тази матрица се нарича *растер.* Подходът за представяне на изображения чрез съхраняване на такава матрица от точки, се нарича растерна графика. Основно неудобство на растерните изображения е, че те са обвързани с размера на матрицата си от точки. Ако потребителят поиска да възпроизведе такова изображението на екран с матрица с различен размер или форма, то често се получава загуба на качество или изкривяване на изображението.

При векторната графика изображенията се изграждат от множество геометрични обекти, наричани *геометрични примитиви.* Най-използваните примитиви са точки, отсечки и гладки криви, както и определени от тях по-големи обекти – многоъгълници, части от равнини, букви. Всички примитиви се описват аналитично. Такова представяне прави изключително удобно прилагането на различни геометрични трансформации върху изображението, без това да наруши визуалните му качества. Такива трансформации могат да бъдат мащабиране, ротация, транслация, създаване на огледален образ и други [12]. Най-разпространеното в момента приложение на векторната графика е изграждане и визуализация на тримерни обекти. Чрез прилагане на различни трансформации върху тези обекти лесно може да се смени гледната точка и да се визуализира даден тримерен обект от различни страни, нещо което е невъзможно при растерното представяне.

Съществена част от работата с векторни изображения е процесът на *растеризация.* При него от векторно-аналитично представяне трябва да се премине в растерна матрица от точки, всяка характеризираща се с цвят. Процесът се осъществява с помощта на геометрични преобразувания –

проекции. При тях може да се задава различна позиция на проекционната равнина, видът на проекцията, точката или направлението на проектиране и други. При растеризация едновременно от две гледни точки се получава стерео изображение, посредством което в наблюдаващия го се създава усещане за дълбочина на тримерния образ.

При векторните изображения лесно може да се симулира движение на даден обект чрез прилагане на трансформация над дадени примитиви – тези които изграждат него или части от него. Чрез последователно прилагане на такива трансформации се получава векторна анимация [13].

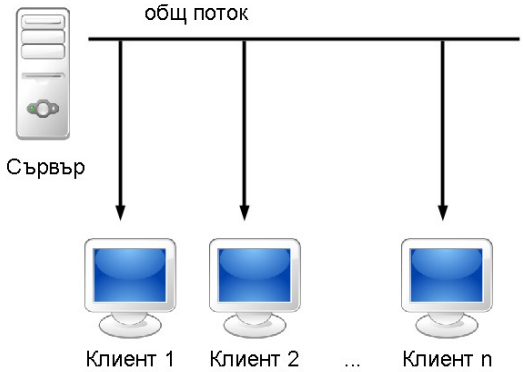
Компютърните видео клипове представляват последователност от изображения, възпроизвеждани последователно с опредена скорост. Тези изображения се наричат *кадри*. При векторната графика, за да се получи даден кадър трябва да се приложи растеризация на съответните му графични примитиви.

Файлово и поточно представяне на клипове

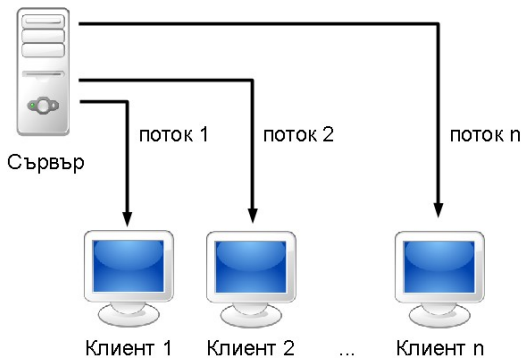
Има два основни сценария при достъп до мултимедийна информация – *файлов* и *поточен*. При първия сценарий, потребителят има произволен достъп до цялата мултимедийна информация във всеки момент от възпроизвеждането. При втория, потребителят получава дадена информация в точно определен момент и няма възможност да прочете данни, които са били доставени в предходен момент или данни, които не са доставени все още. Първият сценарий е свързан със съхранение на мултимедийната информация във файл, достъпен за клиента, докато вторият – с излъчване на мултимедия.

Файловото представяне изисква в началото на възпроизвеждането да са известни позицията, размерът и останалите специфични характеристики на всеки един кадър от клипа. Това прави този начин на съхранение неприложим за живи или непрекъснати предавания като цифрови телевизионни програми или излъчвания от камери на живо. В тези ситуации се използва *поточно* представяне на данните [14].

Има два основни подхода за реализация на инфраструктура за поточно разпространение на видео: *общо* (фигура 1) и *индивидуално* (фигура 2).



Фигура 1. Общо поточно разпространение на видео



Фигура 2. Индивидуално поточно разпространение на видео

При общото поточно разпространение сървърът генерира един поток данни и всеки клиент може да се закачи за него в произволен момент. След прочитане на определено количество данни от потока клиентът може да започне възпроизвеждане. При индивидуалното разпространение всеки клиент получава отделно копие на поточните данни. За всеки клиент на такова предаване е необходимо да се осъществи независимо възпроизвеждане от страна на сървъра, което изисква значителни ресурси. Поради тази причина за индивидуалното предаване натовареността силно зависи от броя на клиентите.

Съществуващи решения – силни и слаби страни

В дисертационния труд са разгледани три категории решения основани на:

1. предварителна растеризация;
2. предаване на векторни сцени;
3. други подходи.

Решенията, основани на предварителна растеризация използват подход, при който за всеки кадър векторната сцена предварително се трансформира в растерно изображение. След това за полученото изображение се прилагат стандартни подходи, използвани за пренос на растерни клипове – най-често основани на MPEG формати и потоци [14]. Най-голямото предимство на този подход е абсолютната прозрачност при възпроизвеждането на клипа за всякакви клиенти. Основни проблеми са невъзможността или ограничеността на клиента да контролира възпроизвеждането от своя страна, както и сравнително голямото натоварване на сървъра, който трябва да извършва три операции – растеризация на векторната графика, последваща компресия и предаване към клиентите.

В текста на дисертацията са разгледани следните решения:

1. *прихващане на видеото* (*capture*) – подход при който изходът, получен във видео контролера, се прихваща и записва кадър по кадър;
2. *архитектура на сървър с предварителна растеризация* – Hybrid ThinClient protocol, разработена от Гентския университет в Белгия [15];
3. *пренасяне на графика в реално време* – Real Time MPEG-4 stream, решение разработено от университета в Калифорния [16];

Решенията за поточно предаване на векторна анимация, основани на предварителна растеризация, изискват повече ресурс от страна на сървъра. Ако се изисква индивидуално предаване, за всеки клиент трябва да се генерира отделен поток, което ще доведе до изискване на изчислителни ресурси, пропорционални на броя на клиентите. Друг общ недостатък е, че клиентите могат да правят трансформации само върху растерната матрица.

Всички описани решения, използвайки за основа компресии и преносни формати, основани на MPEG, позволяват пренос не само на

видео, но и на допълнителни медии – звук, текстова информация и други. Още една съществена положителна страна на този подход е прозрачността на възпроизвеждане и минималните изисквания за ресурси от страна на клиента. Предвид характеристиките на съвременните клиентски компютри и преносими устройства, може да се счита, че клиентите ще имат значителни изчислителни възможности. Това позволява да се използват решения, които прехвърлят повече изчисления към тях.

При **решенията, основани на предаване на векторни сцени,** всяка клиентската машина може да визуализира предаваната графика или анимация с качество, съобразено с възможностите на нейния хардуер – така че той да се използва максимално, без да се претоварва. Друг аспект на клиентския контрол е възможността да се променят геометричните параметри на самата растеризация. Чрез промяна на гледната точка се позволява да се получи изображение на тази част от сцената, която е от интерес за наблюдаващия. Може също да се променят размерите и резолюцията на кадъра, в който се помества полученото растерно изображение, практически неограничено спрямо желанието на клиента [17]. Съобразно тези преимущества на предаването на векторни сцени, вместо предварително растеризирано изображение, съществуват редица решения, които реализират съхранение или предаване на векторна графика и по-конкретно на анимация. По-широко разпространените сред тях са:

1. *Прехвърляне на OpenGL инструкции* - OpenGL Stream Codec, изписван съкратено GLS [18]. Този подход силно напомня прихващането на екрана, но при него се прихващат и записват в двоичен вид всички команди към интерфейса на платформата OpenGL. Основен недостатък на това решение е огромният обем информация, която се генерира, както и обвързаността му с OpenGL.
2. *Програми на WebGL.* Чрез тази библиотека се позволява отдалечено подаване на графичните команди през поток към браузъра, в който кодът се изпълнява. Основен недостатък на тази технология е липсата на възможност за поточно предаване, както и необходимостта тя да работи в уеб браузър чрез специално програмиране от страна на клиента за възпроизвеждане и най-вече за управление на процеса на визуализация.
3. *Adobe Flash.* Тази технология поддържа възпроизвеждане на растерни изображения, видео и аудио медии и анимации, включително и при поточно предаване [19]. Недостатъци са липсата на поддръжка на тримерна графика, както и неефективното използване на системните ресурси.
4. *Пренасяне на векторна графика и анимация, във формат X3D.* Това е формат за представяне на графични сцени и трансформации над тях, описани в текстова или двоична форма [20]. Основно предимство на това решение е богатата гама възможности за управление при клиента. Недостатък е липсата на възможност за пренос на други медии, както и на поточен пренос.

Друг подход за отдалечено поточно предаване на мултимедия е протоколът RDP (Remote Desktop Protocol), разработен от Microsoft за осъществяване на отдалечен достъп до компютър [21]. Подходът има функционалности за пренос на специфична векторна графика. Позволява се използване на Microsoft DirectX API, което дава възможност за пренос на интерактивна тримерна графика. Поддържа се и пренос на звук и други различни медии. Недостатък е обвързаността на протокола със специфична за Microsoft функционалност, както и патентни ограничения.

В таблица 1 е направено сравнение на разгледаните решения, според важните от гледна точка на разработката критерии.

Таблица 1. Сравнение на различните технологии за пренос и представяне на векторна графика

Технология	Векторна графика	Различни медии	3D	Контрол на изобраз.	Поточно предаване	Натов. на сървъра	Натов. на клиента	Натов. на мрежата
Screen Capture	НЕ	НЕ	НЕ	НЕ	ДА	Средно	Малко	Средно
Hybrid ThinClient protocol	ДА, на сървъра	НЕ	ДА	НЕ	ДА	Средно	Малко	Средно
Real Time MPEG-4 stream	ДА, на сървъра	ДА	ДА	ДА	ДА	Голямо	Малко	Средно
GLS	ДА	НЕ	ДА	ДА	ДА	Малко	Малко	Голямо
WebGL	ДА	НЕ	ДА	ДА	НЕ	Малко	Малко	Малко
Flash	ДА	ДА	НЕ	НЕ	Частично	Малко	Средно	Средно
X3D/VRML	ДА	ДА	ДА	ДА	НЕ	Малко	Малко	Малко
RDP	ДА	ДА	ДА	НЕ	НЕ	Малко	Средно	Средно

Анализирайки данните в таблицата се вижда, че нито едно решение не покрива напълно всички поставени критерии. Решенията, които са предназначени за поточно предаване, с изключение на специално разработен за целта протокол, не поддържат възможност за контрол на визуализацията. Вариантите за пренос, които дават пълен контрол на визуализацията при клиента, имат обща слабост – липсата на поточно предаване. Основавайки се на опита, извлечен от различни съществуващи решения, техните силни и слаби страни, може да се направи изводът, че за пренос на векторна графика, даващ възможност да се контролира процесът на възпроизвеждане, е най-удачно растеризацията да се осъществява при клиента. Също така трябва да се предвиди възможност за поточно предаване, за която може да се извлече ценен опит от MPEG базираните формати, доказали се с времето и имащи най-широко приложение в съвременните технологии.

Глава 2. Описание на избрания подход за решение

Дефиниране на проблема и поставените цели

Целта на дисертацията е да се опише подход, чрез който да може да се пренася векторна анимация от сървър, на който тя се генерира, до клиент, където тя се визуализира. Това трябва да става в реално време (с минимално забавяне) и върху нормална съвременна мрежова инфраструктура.

Решението трябва да позволява както общо, така и индивидуално поточно разпространение. При индивидуалното разпространение трябва да се позволява на клиента да контролира работата на сървъра. Освен векторната анимация потокът трябва да позволява пренасяне на други видове медии, най-вече звук и текстова информация – субтитри, коментари или други специални данни като скриптови команди. Възможност за пренос на растерна графика или видео клипове, успоредно на векторната, също не трябва да се изключва.

При предаване на мултимедийната информация в индивидуален поток, клиентите трябва да могат да влияят интерактивно върху възпроизвеждането, а също така на самото предаване на данните – детайлността на обектите от сцената или на качеството на текстурите.

При предаване на мултимедийната информация в общ поток, клиентите трябва да имат възможност да започват възпроизвеждане на пренасяната медия в кратки срокове след започване на приемането на данните. При този тип поточен пренос, интерактивен контрол над сървъра от страна на клиентите не се очаква, но сървърът трябва да адаптира адекватно параметрите на предаваната информация според характеристиките на преносната среда или натовареността си, с цел да се избегне претоварване на сървъра или на мрежата, през която се пренасят данните.

Решението, което е представено в дисертацията, се стреми да покрие тези изисквания като използва стандартни алгоритми, подходи, формати и протоколи. По този начин може да се използват силните страни на доказали се решения в областта и след като се отчетат техните слабости, последните да се преодолеят. Конкретните параметри, които описаният в настоящата дисертация формат се стреми да покрие, са:

1. възможност за предаване на аналитично описана векторна графика;
2. възможност за реализиране на анимация;
3. възможност за предаване на различни типове медии, включително дефинирани от потребителя;
4. възможност за поточно предаване на данните, както в индивидуален, така и в общ режим;
5. стремеж за минимално натоварване на преносния канал.

Описание на елементите, изграждащи векторна сцена и анимация.

Всяка векторна сцена се състои от елементарни геометрични примитиви. В практиката най-честото геометрията на дадено тяло се представя чрез аналитично описание на изграждащата го полигонална мрежа, заедно с описание на начина на оцветяване на отделните многоъгълници, които го изграждат – с цветове или текстури. Един от най-разпространените формати за съхранение на подобна информация е OBJ. Всички геометрични обекти от една сцена могат да бъдат описани в такъв формат, независимо един от друг, чрез координати в собствено – *обектно*

геометрично пространство. Такъв начин на описване позволява прилагане на някаква трансформация над сцената, например преместване на даден обект, без да се променя описанието на самите обекти. Също така могат да се правят геометрични трансформации над обекта, които се отнасят до съпоставянето на обектното му пространство към реалното геометрично пространство на сцената.

В реално използваните сцени, обектите не се визуализират като „голи“ полигонални мрежи, а са покрити или с повърхнини, оцветени с плътни цветове, или с дадени растерни изображения, наречени *текстури*. Процесът на изобразяване на такова растерно изображение върху тримерна полигонална мрежа се нарича *текстуриране*. Текстурите се съхраняват извън описанието на обекта. Това позволява да се визуализират два еднакви обекта, текстурирани по различен начин, както и да се правят промени в оцветяването на даден обект, без да се променя описанието му.

За удобство, при работа с обекти във векторни сцени, често се извършва групиране. Групата представлява определено множество обекти, които се третират като неделимо цяло, и при прилагане на някакви трансформации върху групата те се прилагат еднакво върху всички, съдържащи се в нея, обекти. Самата група също има собствено координатно пространство – *групово пространство.* Позволява се добавяне на група като част от друга група и така се получава йерархично изграждане на обектите в сцената.

При растеризация на тримерно изображение съществено значение има осветяването на сцената. Използват се различни видове светлинни източници [22]: глобално осветяване, точков светлинен източник, насочена светлина, конична светлина и други. Важна е също позицията и характеристиките на камерата, през която се визуализира сцената.

Една векторна сцена може да се опише чрез:

1. файлове, дефиниращи всички обекти, участващи в сцената;
2. текстури, използвани за оцветяването на обектите;
3. описание на групираните обекти – списък на обектите, изграждащи дадена група, всеки съпроводен от матрица, задаваща съпоставянето на обектното му координатно пространство към това на групата;
4. матрица на съпоставяне на обектното координатно пространство към пространството на сцената за всеки обект или група;
5. описание на светлините в сцената – тип на светлинния източник, цвят и интензитет на светлината и допълнителна информация, специфична за типа на източника.

За да се получи **векторна анимация** е необходимо да се дефинира промяна на сцената между два кадъра. Прост начин за постигане на този ефект е предаване на цялата сцена за всеки пореден кадър. Този подход би генерирал прекалено много данни. При векторната анимация разликите между два кадъра, се получават чрез някаква трансформация на обекти от

сцената. Така, по-естествен подход за описване на векторна анимация е предаване на формалното описание на трансформациите, чрез които текущият кадър е получен от предния.

Възможните трансформации касаят обект, група обекти в сцената или светлинен източник. За трансформациите на обекти предоставяме три възможности: добавяне на обект, премахване на обект или промяна на характеристиките на обект. Такава може да е промяната на съответствието на геометричното пространство на обект спрямо това на сцената. Друг тип промени са модификации в рамките на самия обект или обектна група – промяна на оцветяването или трансформация, приложена над даден компонент в рамките на група. Така може да се постигне промяна на самите обекти в сцената. Възможна е и промяна на структурата на група – разпадане на даден обект на съставните му части и последващо тяхно движение в независими посоки. Промените, касаещи светлинните източници, освен добавяне или премахване на източник от сцената, включват и модифициране на характеристиките на светлинен източник – промяна на интензитета и цвета на светлината или позицията и ъгъла за насочените или коничните източници.

Чрез комбинираното прилагане на такива трансформации могат да се опишат голяма част от използваните в съвременните системи за тримерно моделиране анимации [23].

Описание на пренасяния поток данни

Предаването на векторна сцена изисква пренасяне на информация за геометричните обекти, тяхното оцветяване, групиране и позициониране, както и информация за осветлението. За получаване на анимация трябва да се предава информация за трансформациите, приложени над сцената между два кадъра. От тази информация най-съществен по обем дял заема описанието на тримерните обекти – полигоналните мрежи и текстурите. В практиката те се променят сравнително рядко, така че не е нужно предаването им да се случва при всеки кадър. Удобно е информацията за всички обекти, използвани в сцената, да бъде организирана в речник като на всеки обект бъде присвоено уникално име – обектен идентификатор. Така, при изграждането на сцената, обектите могат да бъдат реферирани посредством този идентификатор и извлечени от речника. Когато няколко обекта участват в сцената винаги в едно и също положение един спрямо друг е удобно те да бъдат обособени в по-голям обект – група. При голяма част от модификациите на сцената тази група може да се реферира като един обект и да се намали обемът на предаваните данни. При реализация на анимация е възможно да се наложи промяна на групата – отделяне на обект от нея или добавяне на нов обект. За това трябва да се предвиди механизъм за промяна на описанието на обектите от тип група.

При генериране на анимация, всеки следващ кадър се получава от предния, чрез някакви трансформации. С цел намаляване на обема на пренасяните данни е удачно да се описват само тези трансформации. Те

може да са геометрични трансформации на обект, на група обекти или светлинни източници, както и добавяне или премахване на обект или светлинен източник от сцената. Възможна е също така промяна на текстурирането на даден обект. При наличие на информация за сцената от предния кадър и данните за тези трансформации, клиентът лесно може да построи следващия кадър. Така пренасянето на анимация се свежда до пренасяне на трансформациите между два последователни кадъра.

Проблем при този подход е необходимостта клиентът да разполага с построена сцена за даден кадър, за да може да генерира всички следващи го посредством предаваните трансформации. При поточно разпространение не може да се осигури това условие. Затова се налага периодично да се пренася пълната информация за сцената. Така всеки клиент може да започне да приема потока данни и след известно време да прочете описание на цялата сцена. След това може да започне възпроизвеждане на анимацията, прилагайки последователно получените трансформации. Предаване на пълна сцена е за предпочитане и в ситуациите, когато много обекти се появят или изчезват от сцената – например при преминаване от една стая в друга в дадена игра. В такъв случай е възможно пълното описание да е по-малко като обем от описанието на всички трансформации.

Елементите, които задължително трябва да се предават, за да може да се получи поточен пренос на векторна анимация са:

1. речници на обектите и групирането на обекти;
2. речници на текстурите;
3. пълно описание на сцена;
4. описание на трансформациите между две сцени в два поредни кадъра;
5. промени в речниците.

Пренос на медии, различни от анимация.

Изискванията, наложени към разработвания формат, налагат възможност за пренос и на други медии. В практиката това се постига като всяка медия е организирана в отделен поток данни. След това, при предаване на тези потоци в общ преносен канал, те се разделят на отделни сегменти и се смесват. Този процес се нарича *мултиплексиране* [24].

Описваният в дисертацията формат е проектиран с идеята да може да смесва векторна анимация със звук, текст и други медии, дефинирани от потребителя, чрез мултиплексирането на различни потоци данни. За постигане на това са дефинирани две нива на описване на данните:

1. медийно ниво, което отговаря за описване на всяка една медия в собствен поток.
2. транспортно ниво, което отговаря за описването на начина, по който се смесват медийните данни, пренасят се през общ поток и се възстановяват от клиента за правилно възпроизвеждане.

Глава 3. Спецификация на формата

В тази глава е разгледан подробно разработеният формат. Това включва както синтаксиса и семантиката на предаваните данни в двете нива на формата, така и обосновка за избора на отделните параметри.

Обработката на данните е разделена на две нива – *медийно* и *транспортно*. От страна на сървъра на медийно ниво работят модули, които генерират съдържанието на потоци с различни медийни данни. Тези модули са наречени *медийни генератори*. На по-ниското транспортно ниво работи модул, който получава тези данни и ги разделя на удобни за пренос по-малки пакети. Този модул се грижи също за смесването на различните медии в общ поток. Това е процесът на мултиплексиране и съответният модул се нарича *мултиплексор*.

При клиента на транспортното ниво общият поток се разделя на различни медии и съдържанието на получените пакети се обединява за да се получи първоначалното описание на потоците медийни данни. Модулът, който изпълнява тази задача се нарича *демултиплексор*. Получените от неговата работа данни се подават на по-високото медийно ниво, където специфични модули ги консумират и възпроизвеждат. Тези модули се наричат *медийни консуматори*.

Всяко ниво дефинира структурата на своите пакети, която е независима от работата на другото ниво. В трета глава на дисертацията е описано подробно съдържанието и структурата на данните, обработвани на двете нива на системата.

Дефиниция на формата за мултиплексиране

Всички данни, предавани на транспортно ниво в потока, са организирани в преносни пакети. Тези пакети имат еднаква структура, независимо какъв тип данни пренасят - анимация, звук или други. Това улеснява реализацията на компонентите от транспортното ниво, както и дава възможност за лесно добавяне на нови типове медии. Тъй като пакетите се пренасят смесено в общ поток, те трябва да бъдат така структурирани, че да са ясно **различими** по тип в общия поток, както и **разграничими** помежду си. *Разграничимост* на пакетите в потока означава при последователно сканиране на данните, клиентът да може да разпознае началото и края на пакет. *Различимостта* между пакетите означава еднозначно да е възможно да се определи към кой тип данни принадлежи даден пакет. След оформяне на пакетите и разделянето им по типове, отделните медийни консуматори поемат задачата да обработят специфичните за тях данни.

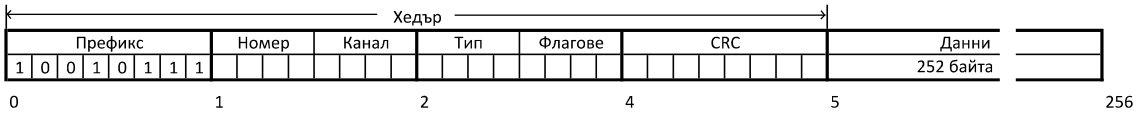
Основните предимства на използването на мултиплексиране, за едновременно предаване на данни от различен тип, е получаването на еднороден поток. Цялата логика, свързана с опаковането, пренасянето и разопаковането на данните, е изолирана в транспортния слой. Такова йерархично разделение на системата позволява също лесна промяна на

структурата на медийното съдържание, включително добавяне на нови типове медии, без това да влияе на начина на пренос на данните между сървъра и клиентите.

Този подход има и недостатък, появяващ се при пренос на свързани по между си медии. Необходимо е да се синхронизира възпроизвеждането им, тъй като данните за медии, които трябва да се възпроизведат едновременно, се пренасят последователно в канала. За целта най-често се използва добавяне на маркер за време във всеки медиен пакет преди мултиплексирането. Той позволява, след възстановяване на пакетите от демултиплексора, те да бъдат възпроизведени едновременно в правилното относително време.

В разработения формат всички преносни пакети имат еднаква структура и фиксиран размер. Използването на фиксиран размер на пакетите има предимството, че улеснява процеса по разграничаване на транспортните пакети.

Всички пакети се състоят от заглавна част, наречена *хедър* (header) и част с данни. Както пакетът, така и заглавната му част са с фиксирани размери, съответно 256 и 4 байта. На фигура 3 е показана структурата на транспортния пакет.



Фигура 3. Структура на транспортен пакет

Типовете медийни канали, които разработеният от автора формат поддържа, са посочени в таблица 2.

Таблица 2. Типове медийни канали

Тип		**Име**	**Описание**
двоичен	**десетичен**		
0000	0	Каталог	описателна информация за всички пренасяни канали
0001	1	Команди	специфични команди, контролиращи работата на клиента
0010	2	Анимация	векторна анимация
0011	3	Звук	звук
0100	4	Текст	свободен, описателен текст или позициониран във времето текст (като субтитри)
0101	5	Речник по поръчка	пълна информация за речници – отговор на заявка от клиента
0110	6	Запазен	запазен за бъдещо развитие на формата
0111	7	Запазен	запазен за бъдещо развитие на формата
1xxx	8-15	Дефиниран от потребителя	свободни стойности на типа, оставени за дефиниране от потребителите на формата

Дефиниция на медийните пакети

За всеки един тип медия, пренасяните данни са организирани в пакети, като всеки пакет се самоописва синтактично в контекста на дадената медия и е неделим. Това означава, че за да бъде извлечена и разбрана синтактично информацията от даден пакет е необходимо той да е наличен изцяло, без да е нужно специално допълнително знание. Ако част от данните в пакета са загубени или повредени е напълно възможно информацията, която той носи, да не може да бъде обработена коректно. За възпроизвеждане на тази информация може да е необходима допълнителна информация, пренасяна в друг пакет от същата медия. Самите медии трябва да са независими помежду си, т.е. трябва да е възможно да бъдат възпроизведени независимо една от друга. Информацията, кой пакет към коя медия се отнася, не се носи в самия пакет, а на транспортно ниво (в транспортните пакети).

Всеки пакет, описващ която и да е медия, се състои от заглавна част, която е с фиксиран размер от 8 байта и тяло с данни с произволен размер. Всеки пакет може да има различен размер. Различните типове медии, изброени в таблица 2, имат различна структура на пакетите си.

Описание на поток с данни за каталог

Целта на данните за каталог е да се опишат всички пренасяни в момента медийни канали. Така клиентът може да прецени какви модули за обработка на данните да стартира. При наличие на няколко канала от определен тип медия, може да се избере кой или кои да бъдат възпроизведени. Например, графика с различни нива на детайлност или текст/звук на различни езици.

Всеки запис в речника може да е с различен размер и да описва различна медия:

1. За анимация се описват препоръчителни размери на екрана за изобразяване, нивото на детайлност и текстово описание;
2. За звук се описва на какъв език е каналът, каква е компресията, която е използвана, броят звукови канали в потока и текстово описание;
3. За текст се описва какъв език е използван в конкретния канал, както и текстово описание на съдържанието.

Описание на поток с командни данни

Този поток служи за пренасяне на информация за команди за управление на поведението на клиента – кога и как да започне възпроизвеждане на съдържанието, каква да е позицията на камерата или нивото на звука. Могат да се предават и специфични команди, свързани с приложение на потока в обучението, добавената реалност и други.

В текущо описаната версия е дефинирана команда, описваща препоръчителните начални параметри за визуализация – позиция на камерата и параметри на проекцията. Други команди не са описани, но възможност за такива е оставена за конкретните приложения, използващи описания преносен механизъм.

Описание на поток с данни за анимация

Фокусът на разработения в дисертацията формат е насочен към векторната анимация. Пренасянето на такъв тип медия изисква няколко различни компонента, представени в следните типове пакети:

1. цялостно описание на сцената;
2. трансформации между два кадъра;
3. цялостно описание на речника с обекти;
4. информация за промяна на речник с обекти;
5. цялостно описание на речника с текстури;
6. информация за промяна на речник с текстури.

Цялостното описание на сцената представлява последователност от описания на елементи от сцената, заедно с подходящо задаване на разполагането им – чрез *матрица на трансформация*. Тези обекти могат да са референции към речника с описание на обекти, описание на група или на светлинен източник. За всеки един от тези типове елементи в текста на дисертацията е дадено детайлно описание на структурата и семантиката на описващите го данни, заедно с мотивация на избора на тези данни.

Трансформациите между два кадъра представляват пакети, чиято задача е да определят как от текущата сцена да се получи следващата и съответно чрез последователното им изобразяване в моменти от време, определени чрез информация, записана в медийните пакети, да се получи ефект на движение – анимация. Предполага се във всеки един момент клиентът да съхранява описание на сцената, логически еквивалентно на описаната йерархична организация. Върху това описание се прилагат последователно описаните в тези пакети трансформации, чрез които се получава сцената на следващия кадър. Поддържаните трансформации са:

0. матрична трансформация – геометрична трансформация, зададена аналитично чрез матрица 4x4;
1. ротация – завъртане на обект на определен ъгъл;
2. транслация – преместване на обект на даден вектор;
3. мащабиране – промяна на размерите на обект;
4. промяна на оцветяване – промяна на цвета или текстурата на елемент от сцената;
5. промяна на прозрачност – промяна на параметъра за прозрачност на елемент от сцената;
6. добавяне на нов елемент от произволен тип в група или в сцената;
7. премахване на елемент от дадена група или от сцената;
8. скриване/показване на елемент – според параметрите деактивира временно или отново активира елемент. Деактивираният елемент присъства в описанието на сцената, но не участва в изобразяването.

Запазени са и множество стойности за бъдещо развитие на формата или свободни за използване от клиентите според техни специфични потребноси. В текста на дисертацията е описана подробно структурата и семантиката на всяка трансформация.

Пакетите от тип **цялостно описание на речник с обекти** пренасят описанието на обектите, цитирани при изграждане на сцената. Речниците се състоят от два типа елементи – геометрично зададени обекти и обекти, зададени като групи от елементи. В описаната в дисертацията версия на формата за геометрично задаване на обект се използва представянето, разработено от компанията WaveFront, а именно obj файловете. Този формат е избран по няколко причини:

1. отворен и безплатен;
2. широко разпространен и поддържан от всички големи разработчици на 3D системи;
3. съдържа цялата необходима информация за геометрично представяне на обекти и разполагане на текстури върху тях.

Недостатък на този формат е текстовото представяне на данните. Това води до сравнително голям обем данни, която трябва да се съхрани или пренесе за описанието на даден обект. За решаването на този проблем е използван компресиращ алгоритъм, чрез който се постига ефектът тези текстови данни значително да намалят обема си. При избора на компресиращ алгоритъм са взети предвид следните критерии:

1. трябва да е отворен и популярен;
2. трябва да компресира добре текст, съставен от статистически малък набор различни символи – предимно числа;
3. трябва да е бърз и да поддържа поточно декомпресиране.

При така наложените изисквания най-удачен е алгоритъмът DEFLATE, който е отворен и популярен. Основа е на формата zip. Използва статистически компресиращ алгоритъм, базиран на код на Хъфман, който е алгоритмично лек и дава много добри резултати за данни с малък брой различни символи.

Типовете елементи на речника с обекти са:

1. *съответствие на идентификатор с псевдоним*. Този запис се използва само за да зададе съответствието на глобалния идентификатор на обекта с неговия псевдоним в текущата сесия. Такъв тип записи е удобен, когато клиентът притежава предварително пълен набор на обектите, които се използват и единствено е необходимо да получи съответствие към кратките им идентификатори.
2. *компресиран обект в obj формат*. Това е основният използван тип елементи в речника. Данните представляват един обект, описан във формата obj, като това описание е компресирано, посредством deflate алгоритъма, например с използването на признатата от стандарта библиотека zlib.
3. *обект в obj формат, без използване на компресия*. Данните в този тип елементи представляват описание на обект в obj формат, като това описание е използвано в суровата му форма. Този подход има

сериозния недостатък, че използва значително по-голям обем данни, в сравнение с компресирания вид (тип 2). Използването му е приложимо, когато пренасянето на речниците не се случва в реално време – например при записване на данните във файл и възпроизвеждане на по-късен етап.

4. *обект-група*. Обектите-групи позволяват да се дефинират сложни модели, съставени от множество вече дефинирани обекти. При тях данните са цитирания на други обекти, вече описани в същия речник, върху които е приложена някаква трансформация. Така лесно могат да бъдат построени сложни тримерни модели, чрез използване на относително малък обем данни.

Пакетите от тип **промяна на речник с обекти** имат за цел да се позволи промяна на речниците по време на предаване, без да се налага препредаване на цялото им съдържание. Особено удобно е използването на този тип пакети, когато клиентът използва предварително съхранена версия на речника с обекти. Чрез тях може да се предават кратки добавки към речниците, вместо цялостно описание, когато по време на предаване в сцената се появи нов обект. Поддържаните промени са добавяне на нов обект, временно изключване на обект от речника и възстановяване на изключен елемент. Оставени са и стойности за бъдещи разширения на форматната спецификация и за свободно използване от потребителите му.

Пакетите от тип **цялостно описание на речник с текстури** пренасят информацията за текстурите, с които обектите могат да бъдат облечени. Това се прави в случаите, когато оцветяване със солиден цвят не е достатъчно. Речниците от текстури се състоят от изображения, описани в някакъв графичен формат. Този формат може да бъде подбран така, че да е удобен за използване при визуализация на построените сцени от страна на клиентите. Поддържат се както стандартни формати за изображения, така и възможност за наслагване на множество текстури една върху друга, подобно на групирането на обекти.

Пакетите за **промяна на речник с текстури** позволяват модификация на речника посредством добавяне на нови обекти в него. За разлика от промяната на речника с обекти, при текстурите няма смисъл от временна забрана на определена текстура, както и от абсолютно премахване на текстура от речника. Ако в дадена конкретна програмна реализация потребителят се нуждае от подобна функционалност, той може да използва оставените за разширение стойности.

Описание на поток с данни за звук

Този поток служи за пренасяне на аудио медия, успоредно с анимацията. Позволява се пренасяне на звук, кодиран в стандартни формати, както и на некомпресиран цифров звук във формат PCM. В текста на дисертацията подробно са изложени типовете поддържани формати, както и данните, използвани за описване на некомпресирания звук.

Описание на поток с данни за текст

Този поток служи за пренасяне на текст, успоредно с анимация или звук. Целта е да се позволи добавяне на субтитри или друго текстово описание, съпровождащо възпроизвеждането на анимацията. При използването на описвания формат в обучението, този канал може да пренася както записки (лекционен материал), съпровождащи визуализацията, така и въпроси за самопроверка на студентите.

Поддържат се няколко типа текст:

0. *Общ текст от системата.* Такъв тип данни се използва за пренасяне на системни и информационни съобщения от сървъра. Например, за пренасяне на съобщения за грешки или журнални (logging) данни. Предполага се обемът на тази информация да не е голям и с цел по-лесното му проследяване текстът се пренася в чист вид (не се използва компресия).
1. *Общ текст към медията.* Такъв тип данни се използва за пренасяне на кратки съобщения, възникващи от страна на приложението, генериращо медията. Например, за пренасяне на чат съобщения, общи пояснения или коментари.
2. *Общ текст към медията, компресиран с deflate*. Този тип данни се използва за пренасяне на обемни съобщения, възникващи от страна на приложението, генериращо медията. Например описателна информация, текст на учебни материали и други.
3. *Общи субтитри – формат SubRip.* За този тип пакети данните представляват текст за субтитри към анимацията, описан във формата SubRip. Този формат е избран заради начина, по който се задават моментите на показване и скриване на текста по време на възпроизвеждане. Друго преимущество е възможността при представяне на текста да се опише елементарно форматиране.
4. *Скрити субтитри – формат EIA-608.* За този тип пакети данните представляват текст за скрити субтитри към анимацията. Този тип представяне на субтитри позволява изключително компактно предаване на символи към всеки кадър, които чрез натрупване изграждат текст. Използването на такъв тип субтитри е полезно при интегриране със софтуерни или хардуерни системи за обработка и разпространение на телевизионен сигнал. За генерирана векторна анимация би имал ограничено приложение. Въпреки това е отделена секция за него, поради широкото му стандартно приложение в сферата на медийната телекомуникация. В САЩ предаването на такъв тип данни към публичните медийни потоци е регламентирано със закон, защитаващ правата на хората с увреден слух.

Описание на поток с речници по поръчка

Този поток служи за пренасяне единствено на речници на обекти или текстури. При нормален режим на предаване, медийни пакети от този тип не присъстват в мултиплексирания поток. Когато генераторът на медия започне ново предаване, такива пакети се изпращат за да могат клиентите веднага да получат нужните им речници. При индивидуално предаване, клиентът може да поиска изпращане на един или всички речници от сървъра с цел да ги съхрани за по-късна употреба или максимално бързо да започне възпроизвеждането. Подобна опция е възможна и при общо поточно предаване, ако сървърът поддържа канал за обратна комуникация от клиентите, например при система от тип „виртуална класна стая“.

Описание на поток с дефинирани от потребителя данни

Този поток служи за пренасяне на всякаква информация, специфична за конкретна реализация на медиен генератор. Съответно детайлна спецификация в дисертацията не е изложена. Наложено е единствено изискването всеки медиен пакет да започва с описаната заглавна част.

Чрез използване на този поток, системите, потребители на формата, могат да реализират предаване на всякакви специфични за целта им данни, например инструкции за поведение на клиента; изпитни въпроси или лекционни материали; описателна информация в специално структуриран вид за реализиране на добавена реалност и много други.

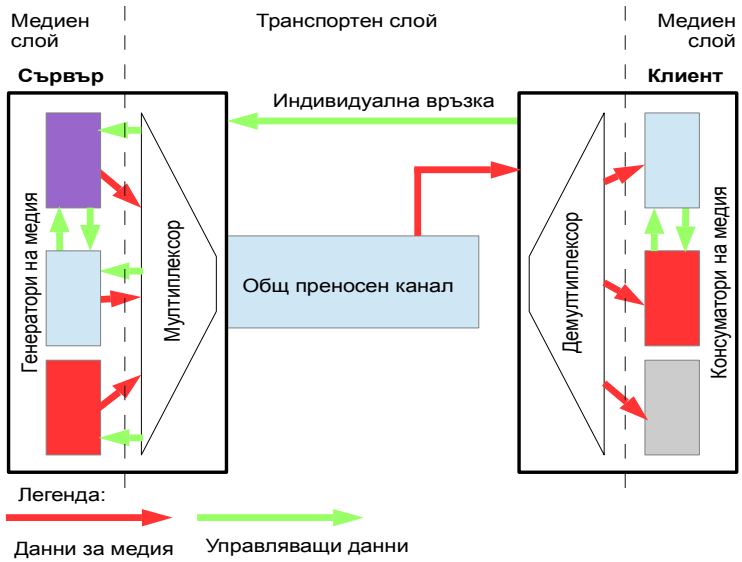
Глава 4. Алгоритми за работа с данните

В тази глава е изложен подход за предаване на векторна анимация в общ поток, съвместно с други медии. Показана е принципна архитектура на софтуерна система, решаваща поставените към дисертацията задачи. Описан е алгоритъм за мултиплексиране, ефективно предаване и демултиплексиране на различни видове медии и синхронизация на възпроизвеждането им. В главата се описва и алгоритъм за изграждане на поток, съдържащ анимация от векторни сцени на сървър, пренасянето му в споделен канал и възпроизвеждането му от клиент. Разгледани са различни схеми на връзка между сървъра и клиента.

Основни приноси на автора, изложени в тази глава, са дефинирането на алгоритъм за пренасяне в общ канал и синхронно възпроизвеждане на множество различни типове медии, както и алгоритъм за балансиране на забавянето на възпроизвеждане спрямо мрежовото натоварване. В процес на подготовка е статия, която ги описва.

За осъществяване на правилна комуникация между сървър, предаващ поточни данни според описания в дисертационния труд формат, и клиент, който да възпроизведе тези данни, е необходимо да бъдат зададени правила за поведение на клиента и на сървъра и интерпретация на данните. Тези правила са описани в глава четвърта на дисертационния труд.

Пренасяната според описания формат информация е разделена на два логически слоя – транспортен и медиен. Първият има задача да осъществи еднотипно пренасяне на различните медии в общ комуникационен канал. Вторият трябва да осъществи описването на различни по вид или съдържание медии. Клиентите извличат данни за медията от общ или индивидуален преносен канал като могат да осъществяват обратна връзка със сървъра посредством персонален канал, когато това е необходимо и се поддържа от системата. Както медийните генератори, от страна на сървъра, така и медийните консуматори, от страната на клиента, могат да разменят помежду си съобщения с цел правилната им съвместна работа. Принципна схема на системата е показана на фигура 4.



Фигура 4. Схема на системата – слоеве, комуникация, преносен канал

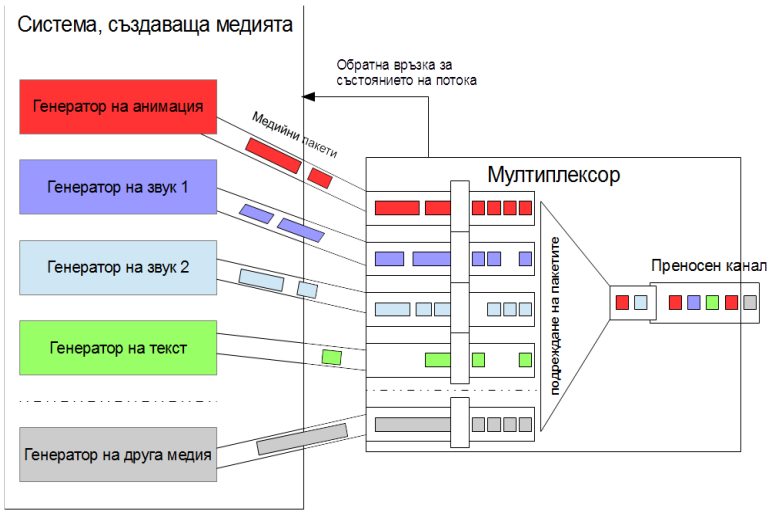
Общи правила на поведение на модулите

Тази част от дисертацията описва общите правила за поведение на модулите на системата в ситуации на непознати или повредени данни. Описани са също правилата за задаване на версия от сървъра при различни разширения на описания формат и разпознаването ѝ от клиента.

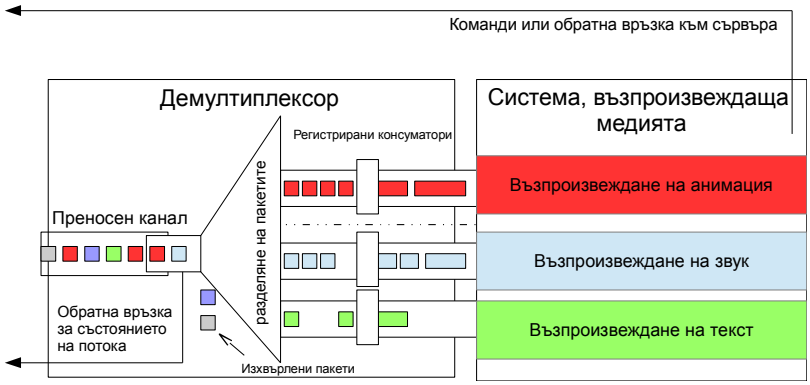
Изграждане, пренасяне и обработване на мултиплексния поток

Тази секция описва подробно поведението на двата модула, работещи на транспортния слой на системата както в индивидуален, така и в общ режим на разпространение. Описана е подробно работата на мултиплексора, включваща регистрирането на медийни генератори, пакетирането на данните в транспортни пакети и алгоритъм за смесването им. Мултиплексор е показан схематично на фигура 5.

Задачата на демултиплексора е по-проста, тъй като не се настройва с определени параметри, а единствено чете данните, получавани в потока и ги предава на съответните медийни консуматори. Схематично работата на демултиплексора е показана на фигура 6. В текста на дисертацията е изложен подход за разпознаване на транспортни пакети и коректното им сглобяване до медийни пакети, подавани на съответните консуматори.



Фигура 5. Схема на мултиплексор



Фигура 6. Схема на демултиплексор

Алгоритъм за изграждане на медийния поток с анимация и пренасянето му

В дисертационния труд е обърнато внимание на медията, описваща векторна анимация. Въпреки това, при построяване на потоците, пренасящи различните типове медия, се използват еднакви общи правила, които са разгледани в контекста на векторната анимация.

В съответната секция на дисертационния труд са описани процедурите по подготовката за предаване на векторна анимация, началните данни, с които това предаване започва и начинът на генериране, предаване и препредаване на различните пакети от описанието на векторната анимация. Подробно е разписана алгоритмична схема за препоръчителна реакция на медийния генератор на различните събития, докладвани от мултиплексора – най-вече събитията за ниско или високо натоварване на преносния канал.

В секцията за **възпроизвеждане на поток векторна анимация** е обърнато внимание на обработката на този поток данни от съответния консуматор. Описано е поведението на модула при получаване на различните видове пакети – построяване и обновяване на сцената по време на анимация.

Въз основа на данните за относителното време на показване, пренасяни с всеки пакет, е описан алгоритъм за **синхронизация на отделните канали**. При този алгоритъм един от каналите се избира за водещ, а останалите се синхронизират към неговото време на показване. Описаната алгоритмична схема има за цел синхронизирано възпроизвеждане на медии, но може да се използва и за управление на поведението на системата. Ако даден нов речник се маркира с някакво време на показване, той не се използва преди това време да настъпи. Същият подход може да се използва и за команди или каталози, за да се определи по-късен момент, в който те да станат валидни.

В края на глава четвърта е описана възможността за реализация на обратна връзка от клиента към сървъра. Такава трябва да има при индивидуален режим на предаване на медията. При общ режим е възможно сървърната страна да позволява на клиентите да се свързват индивидуално с нея, използвайки отделен канал – команден канал.

Чрез такъв механизъм клиент може да изпраща команди към сървъра – да поиска речници, по-високо или по-ниско качество на сцената или да зададе други команди, влияещи на работата на някой компонент на сървъра. Обратната връзка може да е еднократна или постоянна.

Глава 5. Тестова реализация и емпирични измервания

В тази глава е описана прототипна система, чрез която са извършени серия тестове с цел да се потвърди коректността на разработеното решение и да се изследва поведението му в различни ситуации. Описани са някои детайли от реализацията на системата, начинът на построяване на тестовите сценарии и са приложени резултати от изпълнението им, заедно с кратък анализ. Научно-приложни приноси, постигнати в тази глава са подборът на различните тестови сценарии и анализът на резултатите от тяхното изпълнение. Приложни приноси са реализация на описания формат и разработването на тестовата система. Подготвя се публикация на тези резултати.

При подбор на тестовите сценарии е поставена цел да се проверят следните възможности на решението:

1. пренасяне на векторна анимация;
2. пренасяне едновременно на различни типове медии;
3. предаване в непрекъснат споделен поток;
4. възможност за започване на възпроизвеждане от произволен момент на предаването;
5. приемливо натоварване на преносния канал и процесорите на клиента и сървъра при пренос на реални като параметри сцени.

За проверка на тези критерии са проектирани три тестови сценария, всеки изпълнен в режим само на анимация и в режим на смесване на анимация и звук.

Първият представя пренасяне на проста анимация – сцена от един обект, който е подложен на една и съща трансформация във всеки кадър. Целта на този сценарий е проверка на системата при най-проста ситуация. Чрез използването на споделен преносен канал и свързване на клиент в произволен момент след започване на предаването с този сценарий се проверяват първите четири критерия.

Вторият сценарий изгражда сцена, съставена от няколко сложни обекта. За всеки кадър се генерират на произволен принцип множество трансформации. Част от трансформациите не са валидни, което е умишлено направено – проверява се поведението на клиента при обработка на некоректни данни. В този тест се подават и промени по речниците. Основната цел на сценария е да се провери натоварването, което се генерира при медия, близка до очакванията на автора за реално приложение.

Сцената в третия сценарий е построена от външно приложение, използващо класовете на тестовата система за отдалечена визуализация на физична симулация. Приложението не е написано от автора. Целта на този тестов сценарий е по-задълбочена проверка на системата, от една страна, и валидация на използваемостта ѝ от реални външни системи, от друга.

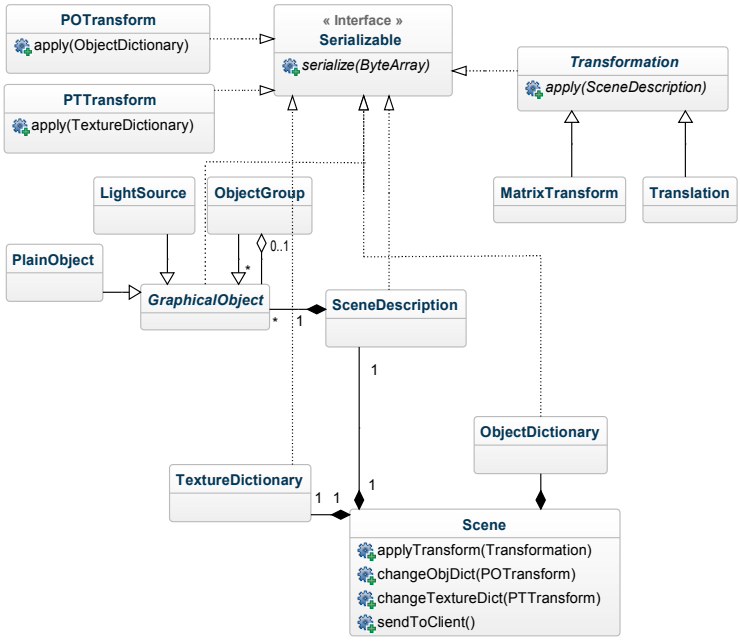
В текста на глава пета е описана хардуерната среда, върху която са изпълнени тестовете – система от два компютъра. На единия работи генериращ общ поток сървър, а на другия клиент, възпроизвеждащ получените от потока медии.

Описана е и софтуерната страна на тестовата система. Изброени са модулите, които изграждат сървъра и клиента и е описана алгоритмичната схема на работа им. Посочени са конкретни детайли от софтуерната архитектура и реализация на системата. Част от обектния модел на генератора на анимация е показана на фигура 7.

За всеки от тестовите сценарии са описани изпълнението му и получените резултати. Емпиричните данни от тестовете са приложени в табличен вид. Направен е анализ и са посочени изводи. Таблици 3, 4 и 5 показват резултатите от първия, втория и третия сценарий.

Най-важният резултат от проведените тестове е потвърждението, че разработеният в дисертацията формат успешно пренася векторна анимация, заедно с други медии. Натоварването на преносния канал, дори при тежки тестове, остава под ширината на съвременните мрежи. Добавянето към общия поток на канал, пренасящ звук, увеличава натоварването на мрежовия трафик константно за всички тестови сценарии. Натоварването на процесорите, както на клиента, така и на сървъра, слабо се влияе от този параметър. При клиента натоварването очаквано расте почти линейно спрямо честотата на кадрите. При сървъра натоварването зависи от сложността на генерираните трансформации.

При изпълнение на тестовете са отчетени серия ограничения и непълноти на описания формат. Част от тях са реализирани веднага и са описани във формата, изложен в дисертационния труд. Други са оставени като възможност за бъдещи разширения.



Фигура 7. Обектен модел на генератор на анимация.

Таблица 3. Резултати от тестовете по първия сценарий

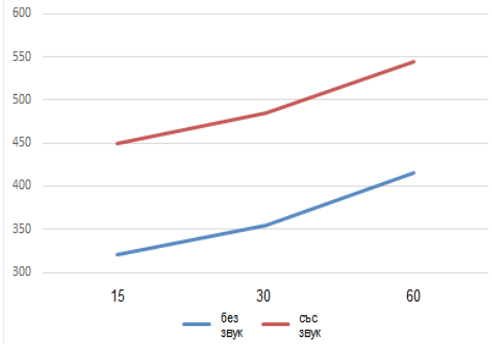
Тест		Натоварване на мрежата	CPU на сървъра	CPU на клиента	Диаграма на натоварването на мрежата
Звук	fps				
НЕ	15	320 Kbs	5%	3%	
НЕ	30	355 Kbs	7%	7%	
НЕ	60	415 Kbs	8%	15%	
ДА	15	450 Kbs	5%	4%	
ДА	30	485 Kbs	7%	7%	
ДА	60	545 Kbs	8%	15%	

Таблица 4. Резултати от тестовете по втория сценарий

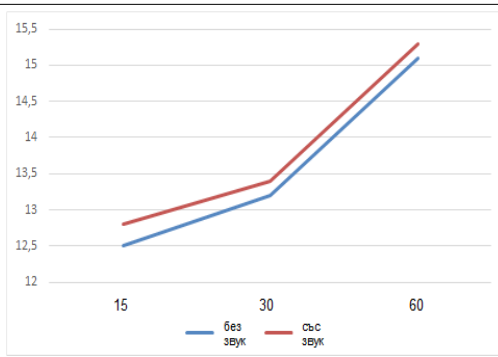
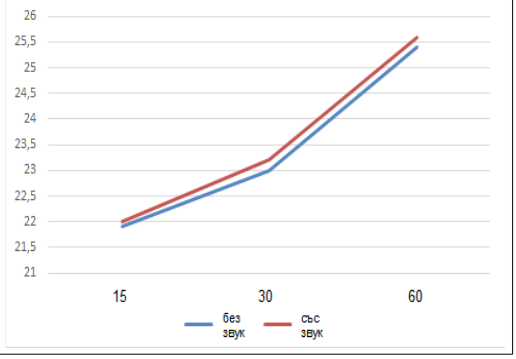
Тест		Натоварване на мрежата	CPU на сървъра	CPU на клиента	Диаграма на натоварването на мрежата
Звук	fps				
НЕ	15	12,5 Mbs	10%	7%	
НЕ	30	13,2 Mbs	12%	11%	
НЕ	60	15,1 Mbs	18%	23%	
ДА	15	12,8 Mbs	11%	7%	
ДА	30	13,4 Mbs	12%	11%	
ДА	60	15,3 Mbs	18%	23%	

Таблица 5. Резултати от тестовете по третия сценарий

Тест		Натоварване на мрежата	CPU на сървъра	CPU на клиента	Диаграма на натоварването на мрежата
Звук	fps				
НЕ	15	21,9 Mbs	22%	12%	
НЕ	30	23,0 Mbs	40%	19%	
НЕ	60	25,4 Mbs	65%	45%	
ДА	15	22,0 Mbs	22%	12%	
ДА	30	23,2 Mbs	40%	20%	
ДА	60	25,6 Mbs	65%	45%	



Заключение

Обобщение и получени резултати

Целта на дисертационния труд е създаване на формат, който описва мултимедийна информация и по-конкретно векторна анимация. Към формата трябва да се опише и протокол, който позволява поточното пренасяне на различни мултимедийни данни. Такова пренасяне трябва да позволява множество клиенти да слушат едновременно даден поток като могат да се свързват към него в произволен момент.

Описаното решение постига поставените цели – позволява предаване на векторна анимация, синхронно с други медии. Поддържат се двата типа предаване на медията – индивидуално и общо. Структурата на формата и правилата от преносния протокол решават проблема с поточното излъчване и възможността за стартиране на възпроизвеждане непосредствено след свързване към канала в произволен момент от излъчването. При дефинирането на формата са оставени възможности както за бъдещо развитие, така и за добавяне на специфична функционалност от системите, които го използват. По този начин по-нататъшното развитие на формата и адаптирането му към конкретни специфични цели е лесно. Направените емпирични измервания показват, че форматът може да пренася векторна анимация, синхронно със звук, при ниско натоварване на преносния канал. Това доказва приложимостта на разработеното решение.

Полезно приложение на разработката е възможността клиентът да дефинира сам параметрите при растеризация. Така при желание може да генерира едновременно изображения за една и съща сцена от две различни гледни точки и да получи стерео-3D изображение с параметри, които може да се настройват без никаква намеса от страна на сървъра.

Доказателство за реалната приложимост на разработката е също интересът за използването ѝ за отдалечена визуализация на резултат от флуидна симулация. Той е проявен от голяма софтуерна компания, разработваща система за фотореалистична визуализация на графика – стандарт в съвременната архитектурна и кино индустрия.

Според параметрите за оценка на решенията, посочени в таблица 1, разработеното в дисертацията решение покрива всички критерии, а именно:

1. възможност за пренасяне на векторна графика;
2. възможност за пренасяне на различни типове медии;
3. възможност за описване и пренасяне на тримерни сцени;
4. възможност за контрол на изобразяването при клиента;
5. поточно предаване на медиите и възможност за възпроизвеждане след свързване в произволна позиция на потока;
6. слабо натоварване на сървъра и клиента извън съответно генерирането и възпроизвеждане на графиката и останалите медии;
7. средно натоварване на мрежата, съизмеримо с WebGL и X3D.

Предвид покриването на тези критерии и съпоставка с описаните в таблица 1 решения, **разработеният в дисертацията оригинален формат за описание и поточен пренос на векторна анимация, покрива всички заложени изисквания и е по-добър от наличните към момента алтернативни решения.**

Идеи за бъдещо развитие

Въпреки, че постига поставените цели, разработеният формат далеч не изчерпва областта.

Една посока за подобрение е допълнително намаляване на обема на пренасяните данни. За целта може да се използва по-добра компресия на обектите или да се подбере по-ефективен, от гледна точка на обема, формат за представяне на обектите. Друг начин за намаляване на обема данни е използване на 16 битови числа с плаваща точка за описание на трансформациите. Този подход ще затрудни както генерирането, така и възпроизвеждането на медията на стандартни процесори, но ще намали обема на пренасяните данни и ще е удобен за директна работа с графичните процесори, които поддържат такива данни.

Друга посока за развитие е разширяване на функционалността на формата. Един начин за това е въвеждане на макро-език за описание на трансформациите, чрез който с минимален обем да се предават сложни трансформации – процедурни, механични или циклично повтарящи се. В настоящата версия на формата липсва поддръжка на шейдъри. Кодът на шейдърните програми може да се пренася в отделен тип пакет, подобно на текстурите и да се цитира по подобен начин. Полезно би било дефиниране на канал, пренасящ медия, описана общо по тип, например чрез MIME декларация, широко използвана в много приложни мрежови протоколи.

Трето направление за развитие е разширяване на възможностите на протокола. Възможно е реализиране на повече команди за управление както на поведението на клиентите, така и на сървъра. Такива команди ще са много полезни при реализация на учебни системи.

Форматът, резултат от изследванията, описан в дисертационния труд, позволява реализация на тези модификации и разширения без промяна на описанието и нарушаване на функционалността му – само чрез заложените възможности за разширение. Това също съответства на поставените цели.

Цитирана литература в автореферата

[1] L.A. Siiman, M. Pedaste, Towards a pedagogy for using digital 3-d content in science education, ICERI 2013

[2] Jean-Eric Pelet, E-Learning 2.0 Technologies and Web Applications in Higher Education, IGI Global, 2013

[3] Trak Lord, Visual Computing Meets Computer Vision: Augmented Reality & The GPU, Siggraph 2013

[4] Adam Alexander, SVG. Scalable Vector Graphics, Franzis Verlag, 2002

[5] Ian Hickson, Extending HTML, W3C HTML-5 pre-release discussion board, 2004

[6] Festa Paul, Borland John, Is a 3D web more than just empty promises?, CNET news, May 2005

[7] Mike Donald, Understanding WoW Communication Protocol, World of Warcraft unofficial blog, June 2011

[8] Justin Olivetti, RIFT Streaming Client, Massively at Joystiq, May 2012

[9] Robert Torres, Video Games for Learning and Assessment: Potential for a Sea-change in the Educational Landscape, ICERI 2013

[10] Kimberely Nettleton; Lesia Lennex, Cases on 3D Technology Application and Integration in Education, IGI Global, 2013

[11] Ramesh Jain, SIGMM Annual Report, July 2004-June 2005

[12] Vince John, Vector Analysis for Computer Graphics, Springer, 2007

[13] Mark de Berg, Cheong O., Kreveld M., Overmars M., Computational

Geometry Algorithms and Applications Third Edition, 3-rd ed., Springer, 2008

[14] Keith Jack, Video Demystified. A handbook for Digital Engineer, 4th ed. Elsevier, 2005

[15] D. De Winter, Simoens P., Deboosere L., A Hybrid ThinClient protocol for Multimedia Streaming and Interactive Gaming Applications, Network and OS Support for Digital Audio and Video Conference, USA, May 2006

[16] Liang Cheng, Bhushan A., Pajarola R., Zarki M., Real-Time 3D Graphics Streaming Using Mpeg-4, University of California, July 2004

[17] Shirley Peter, Fundamentals of Computer Graphics, A K Peters, 2005

[18] Silicon Graphics inc., “he OpenGL Stream Codec: A Specification, 1996

[19] E. James Shuman, Adobe Flash CS6 Revealed, Delmar, 2012

[20] ISO/IEC 19776 - Extensible 3D (X3D) encoding

[21] Understanding the Remote Desktop Protocol, Microsoft, 2011

[22] S. Robertson, Bertling T. How to Render: the fundamentals of light,

shadow and reflectivity, Design Studio Press, 2014

[23] C. Webster, Animation: The Mechanics of Motion, Focal Press, 2005

[24] R. Bates, M. Bates Principles of Voice & Data Communications, Career

Education, 2006

Авторска справка

По мнение на автора основните приноси в настоящата дисертация са следните:

Научни:

- Направен е преглед, обзор и анализ на съществуващи решения на поставените проблеми. Оценени са техните предимства и недостатъци (Глава 1).
- Дефинирани са изисквания към формат за представяне на векторна анимацията и протокол за пренос на мултимедия, включваща такава анимация (Глава 2).
- Дефиниран е формат за векторни сцени и трансформации, който покрива всички заложени изисквания (Глава 3).
- Дефиниран е преносен формат за векторна анимация, подлежащ на поточен пренос, който покрива всички заложени изисквания (Глава 3).
- Построена е схема за мултиплексиране и съвместно предаване на векторна анимация с други медии, чрез адаптиране на идеята, залегнала в наложилия се формат MPEG-1.1 (Глава 3 и Глава 4).
- Дефиниран е мултиплексен формат, който позволява синхронен пренос на множество медии в един канал, покриващ всички заложени изисквания (Глава 3 и Глава 4).
- Описан е адаптивен модел на протокол за пренос на поточни мултимедийни данни между клиент и сървър, осигуряващ нормално натоварване на преносната среда (Глава 4).

Научно-приложни приноси:

- Дефинирани са подходящи тестови сценарии за проверка на поведението на описания формат (Глава 5).
- Проведени са емпирични изследвания и е направен анализ на резултатите за натоварване при използване на предложените формати и протоколи в тестово софтуерно решение (Глава 5).

Приложни приноси:

- Реализиран е описаният формат и алгоритмите за използването му (Глава 5).
- Реализирана е прототипна система, чрез която са проведени тестове и е потвърдена валидността на описания в дисертационния труд формат (Глава 5).

Публикации и доклади, свързани с темата на дисертацията

Публикации:

П1. P. Armyanov, Minimal metadata set required for virtualisation of multimedia files – Proceedings of Third international conference Information systems and grid technologies, 28 - 29 May 2009, Sofia, Bulgaria (pp. 46-53)

П2. П. Армянов, Поточни формати за векторна графика, XVIII Международна научна конференция за млади учени, 3 - 5 юли 2009 г. Юндола, България (pp. 158-165)

П3. P. Armyanov, Open MP parallel adaptation of Loop's mesh subdivision algorithm, Proceedings of 4th International Conference on Information Systems and Grid Technologies, 28-29 May 2010 Sofia, Bulgaria. (pp. 184-188)

П4. P. Armyanov, File Format for Storage of Multimedia Information, Mathematica Balkanica New Series, Vol. 24, Fasc 3-4, 2010, (pp. 293-302)

Доклади:

Д1. Доклад на тема „Файлов формат за съхраняване на метаданни за мултимедия“, Научна сесия на ФМИ, 2009, София.

Д2. Доклад на тема „File Format for Storage of Multimedia Information“, Proceedings of SEE Young Researchers Workshop, Tempus Doctoral Studies In Mathematical Sciences, 16 – 20 September 2009, Ohrid, Macedonia.

Д3. Доклад на тема „Последователно пренасяне на графични примитиви за 3D визуализация“, Пролетна научна сесия на ФМИ, 2011, София.

Д4. Доклад на тема „Файлов формат за съхраняване на мултимедийна информация – поточен пренос на векторна анимация“, Пролетна научна сесия на ФМИ, 2014, София.

Д5. Доклад на тема „Файлов формат за съхраняване на мултимедийна информация – поточен пренос на векторна анимация“, Пролетна научна сесия на ФМИ, 2016, София.

Декларация за оригиналност

Настоящият дисертационен труд е изцяло авторски продукт и в неговото разработване не са използвани чужди публикации и разработки в нарушение на авторските им права.

Биографични данни за автора на дисертационния труд

Петър Армянов е роден през 1982 г. в гр. Добрич. През 2001 г. завършва Природо-математическа гимназия – Добрич. Същата година продължава образованието си във Факултета по математика и информатика (ФМИ) на Софийски университет (СУ) „Св. Климент Охридски“, специалност „информатика“. Дипломира се като бакалавър през 2005 г. През 2008 г. получава магистърска степен по информатика, специалност „софтуерни технологии“, отново във ФМИ. Магистърската му теза е в областта на описание на файлови формати за растерни видео клипове. С дипломирането си, Петър е зачислен като докторант на самостоятелна подготовка към катедра „компютърна информатика“ на ФМИ. Темата на докторантурата му е „Файлов формат за съхраняване на мултимедийна информация – поточен пренос на векторна анимация“.

От 2003 г., като студент, Петър Армянов работи като хоноруван преподавател във ФМИ и води упражнения към курсове по „увод в програмирането“, „обектно ориентирано програмиране“, „структури от данни и програмиране“, „семантика на езиците за програмиране“. От 2006 г. и понастоящем е асистент към катедра „компютърна информатика“ на ФМИ. През следващите години води упражнения и в други курсове – „логическо програмиране“, „функционално програмиране“, „изкуствен интелект“, „езици за системно програмиране“, „език за програмиране C#“, „анализ и проектиране на информационни системи“. Лектор е в серия изборни дисциплини: „проектиране и анализ на компютърни алгоритми“, „сложни структури от данни“, „управление на паметта“, „разширен семинар по програмиране“.

През 2010 и 2011 година е помощник треньор на студентските отбори по програмиране на ФМИ.

Научните интереси на Петър Армянов са разнообразни, но най-вече са в областта на езиците за програмиране, алгоритмите, мултимедията и високо производителните системи.

Благодарности

Поднасям най-сърдечните си благодарности на доц. д-р Павел Бойчев за невероятното търпение, което прояви към мен.

Огромно благодаря на проф. д-р Магдалина Тодорова, която ме подкрепя и мотивира през всичките години в катедрата.

Благодаря на доц. д-р Стоян Бъчваров, който пръв ме насочи по пътя на преподаването и повярва в мен.

Благодаря на доц. д-р Атанас Семерджиев за това, че ми показа красивото във ФМИ, когато бях спрял да го виждам.

Благодаря на колегите от ФМИ, и по-специално на всички от катедра „компютърна информатика“, за подкрепата и доверието.

Благодаря на семейството си, на близките и приятелите, за подкрепата в тежките моменти.