## *ОГЛАВЛЕНИЕ*

### Часть 1. Тематический выпуск "HOMO LUDENS", I

## Часть 2. Оригинальные статьи

## ТЕМАТИЧЕСКИЙ ВЫПУСК "HOMO LUDENS", ЧАСТЬ 1

## ПРЕДИСЛОВИЕ

### Формирование российской школы исследования видеоигр

Уважаемые коллеги, исследователи и энтузиасты игровой индустрии!

Представляю вашему вниманию первую часть тематического выпуска, организованного на базе научных трудов II Всероссийской конференции разработчиков видеоигр "Homo Ludens" (Человек Играющий), которая состоялась в стенах Казанского федерального университета 27 декабря 2024 года. Эта конференция стала значимой площадкой для обмена опытом, идеями и инновационными разработками в области создания видеоигр. Мы собрали под одной крышей талантливых исследователей, разработчиков и студентов, объединенных общей целью – развития отечественной игровой индустрии и продвижения научного подхода к разработке игр.

Работы, представленные ниже, охватывают широкий спектр актуальных направлений: от генеративных методов создания адаптивных персонажей и автоматизированного переноса игровых сцен между движками до анализа биометрических данных для разработки адаптивных сред в виртуальной реальности и типизации кооперативных механик многопользовательских игр. Отрадно видеть такие исследования, направленные на оптимизацию процессов разработки, как создание синтетических датасетов для скиннинга 3D-моделей и разработка документации игрового дизайна для VR-проектов.

Каждая работа, вошедшая в тематический выпуск, представляет собой уникальный вклад в развитие теоретической и практической баз игровой разработки. Междисциплинарный характер исследований наглядно демонстрирует, что современное игростроение находится на стыке программирования, дизайна, психологии, математики и искусства.

Выражаю искреннюю благодарность всем авторам за их вклад в формирование научного дискурса в области разработки видеоигр, а также организационному комитету конференции, рецензентам и всем, кто принимал участие в подготовке и проведении этого мероприятия.

Уверена, что материалы тематического выпуска будут полезны как опытным специалистам, так и студентам, только начинающим свой путь в захватывающем мире разработки видеоигр. Надеюсь, что конференция "Homo Ludens" продолжит свое развитие и в будущем станет еще более представительной и значимой площадкой для профессионального общения и обмена опытом.

Председатель конференции "Homo Ludens", редактор-составитель тематического выпуска,

В.В. Кугуракова

УДК 004.428.4+004.832.28+004.925.84

# ГЕНЕРАТИВНАЯ СИМУЛЯЦИЯ ИГРОВОГО ОКРУЖЕНИЯ В РЕАЛЬНОМ ВРЕМЕНИ

**Э. С. Большаков[1] [0000-0002-2208-9515], В. В. Кугуракова[2] [0000-0002-1552-4910]**

[1, 2]*Институт информационных технологий и интеллектуальных систем Казанского федерального университета*

[1]edward.bolshakov117@gmail.com, [2]vlada.kugurakova@gmail.com

***Аннотация***

Рассмотрены возможности генеративных нейросетевых симуляций с фокусом на применении методов обучения с подкреплением и нейросетевых мировых моделей для создания интерактивных миров. Описаны ключевые достижения в области обучения агентов с использованием обучения с подкреплением. Особое внимание уделено нейросетевым моделям мира, а также генеративным моделям, таким как *Oasis*, *DIAMOND*, *Genie* и *GameNGen*, использующим диффузионные сети для создания реалистичных и интерактивных игровых миров. Рассмотрены возможности и ограничения моделей генеративных симуляций, такие как проблемы с аккумуляцией ошибки и ограничениями памяти, а также их влияние на качество генерации. В заключении названы темы дальнейших исследований.

***Ключевые слова****: видеоигры, игровое окружение, генеративная симуляция, обучение с подкреплением, генеративные нейросети, симуляция игрового процесса, мировые модели.*

## ВВЕДЕНИЕ

Развитие технологий обучения с подкреплением и нейросетевых мировых моделей открыло новые горизонты в области генеративных нейросетей. Особенно интересен прогресс в области видеоигровой индустрии, так как видеоигры являются отличным примером виртуального окружения, которое может использоваться для обучения агентов определенному поведению.

На текущий момент для разработки видеоигр в основном используют готовые игровые движки[1], такие как *Unreal Engine*[2], *Unity*[3] или *Godot*[4], которые уже стали стандартом игровой индустрии. Они берут на себя основные задачи по симуляции физики, просчету столкновений и отрисовке игровой графики, а основная логика разрабатывается традиционным программированием в рамках этих готовых инструментов.

Модели генеративных симуляций способны произвести революцию в устоявшемся подходе, заменив собой основные функции игровых движков. Использование этих моделей позволит генерировать интерактивные игровые миры так же просто, как это сейчас делают модели генерации изображений по текстовым описаниям. За последние несколько лет было представлено несколько моделей, способных полноценно симулировать игровой процесс, и они уже показывают многообещающие результаты.

В настоящей работе проанализированы перспективы использования генеративных нейросетевых симуляций. В первом разделе представлены обзор методов обучения с подкреплением, их эволюция и связь с генеративными симуляциями. Во втором разделе описаны нейросетевые модели мира как продолжение исследований в области обучения с подкреплением, применяемые для моделирования виртуальных миров. В третьем разделе рассмотрены генеративные модели, которые применяют диффузионные сети для разработки реалистичных и интерактивных игровых пространств. В четвёртом разделе показаны ограничения генеративных симуляционных моделей, в частности, отмечены проблемы накопления ошибок и ограничения по памяти, а также их влияние на качество генерации. В заключении выделены направления будущих исследований.

[1] Игровой движок (англ. game engine) – программное обеспечение, предназначенное для разработки и создания видеоигр и представляющее собой комплекс программных компонентов, обеспечивающих основные функциональные возможности игры: визуализацию графики, физическое моделирование, управление памятью, воспроизведение звука, работу скриптов, анимацию, искусственный интеллект, сетевой код, управление игровым процессом и др. Современные игровые движки, такие как Unreal Engine, Unity, CryEngine, являются интегрированными средами разработки, предоставляющими набор инструментов для создания интерактивного контента.

[2] http://unrealengine.com/

[3] https://unity.com/

[4] https://godotengine.org/

**RL-АГЕНТЫ**

Создание генеративных симуляций в первую очередь обязано прогрессу в области обучения с подкреплением (Reinforcement Learning, далее в статье RL) [1]. Это область машинного обучения, в которой агент обучается принимать решения, взаимодействуя с окружающей средой и получая обратную связь в виде вознаграждений или наказаний. Цель RL-агента – максимизировать суммарное вознаграждение, выполняя оптимальные действия в различных ситуациях.

С момента своего возникновения RL претерпело значительные изменения. Ранние методы, такие как Q-обучение, позволяли агентам обучаться на основе таблиц Q-значений. Однако с развитием глубоких нейронных сетей появились более мощные алгоритмы, такие как Deep Q-Networks (DQN), которые используют глубокие нейросети для приближения Q-функций [2]. Это позволило значительно улучшить производительность агентов в сложных задачах.

Последующее развитие привело к созданию алгоритмов, способных обучаться без использования человеческих данных. Например, *AlphaGo Zero*, разработанный DeepMind [3], обучался, играя против самого себя, и достиг уровня игры, превосходящего предыдущие версии, включая *AlphaGo*, которая для обучения использовала выборку человеческих партий [4].

Яркими примерами RL-агентов, обученных на видеоиграх, являются также *AlphaStar* от DeepMind – агент, обучившийся играть в *StarCraft II*[5] на выборке из записей профессиональных игр [5], и *OpenAI Five* – агент, играющий в *Dota 2*, обученный компанией OpenAI на аналогичном датасете [6]. При этом вторая модель способна моделировать игру сразу 5 человек – именно столько человек находятся в одной из двух противостоящих команд в единичной игровой сессии *Dota 2*. Подобные обученные RL-агенты могут также использоваться в тестировании видеоигр, где они способны эффективно заменить обычных игроков в задачах тестирования игрового баланса [7, 8].

[5] Все упомянутые видеоигры приведены в разделе Лудография.

**Моделирование взаимодействия агента с окружающей средой**

В задачах обучения с подкреплением для моделирования процессов обучения и взаимодействия агента с окружающей средой используется модель марковского процесса принятия решений – МППР (Markov Decision Process – MDP) [9].

Агент наблюдает определенное состояние (S) окружающей среды, затем совершает определенные действия (A), взаимодействуя с этой средой, и получает вознаграждение (R). В зависимости от действий агента состояние среды изменяется с вероятностью (P). Целью оптимизации для МППР является поиск оптимальной стратегии ($\pi$), которая представляет собой функцию зависимости действий агента от текущего состояния окружающей среды [10].

Большинство моделей, рассматриваемых в этой статье, использует обобщенную модель МППР – частично наблюдаемый марковский процесс принятия решений. В этом обобщении агент наблюдает не полноценное состояние окружающей среды, а использует сенсорную модель – распределение вероятностей наблюдений (O) для определенного состояния. Стратегия такого процесса уже сопоставляет последовательность наблюдений с действием агента [11].

**Сбор данных с помощью RL-агентов**

Для создания правдоподобной симуляции, с которой будет взаимодействовать игрок, обучение должно происходить на выборке записей игры реальных людей. Поскольку получить достаточно большой датасет таких записей может быть довольно проблематичным, можно с этой целью обучить RL-агента, приближенно имитирующего действия реального человека. Такой подход позволяет автоматизировать процесс сбора данных – записей геймплея одновременно с записями действий RL-агента [12].

В отличие от типичного набора параметров во время обучения с подкреплением, который направлен на максимизацию игрового результата, здесь цель обучения – собрать набор обучающих данных, которые напоминают игру человека и содержат достаточно много разнообразных примеров из различных игровых сценариев, чтобы повысить полноту обучающей выборки. Из-за этого функция вознаграждения для обучения RL-агента будет фиксированной для конкретной игры.

С другой стороны, можно использовать самого агента как основу для создания генеративных симуляций. Речь идёт о самом процессе изучения окружающей среды во время обучения. Агент может сам представлять собой модель, обучающуюся моделировать правила окружения, чтобы на основе этих правил выбирать следующее своё действие. Модель мира, которую агенты изучают в процессе, становится основой для генеративной симуляции.

Изучение модели мира в процессе обучения – это один из подходов (model-based) для обучения агентов, сформировавшийся в области обучения с подкреплением [13].

## НЕЙРОСЕТЕВЫЕ МИРОВЫЕ МОДЕЛИ

В этом разделе будут рассмотрены ключевые работы, связанные с model-based подходом к обучению RL-агентов, в частности, в игровых средах. Основной фокус при рассмотрении работ сделан как раз на потенциальной возможности использовать изученные мировые модели для симуляции.

### World Models

В одноимённой работе World Models авторы предлагают использовать подобные модели для обучения агентов методом обучения с подкреплением [14]. Нейросетевая модель сама представляет собой агента, обучающегося при взаимодействии с моделируемым миром, и состоит из трёх компонентов. В процессе обучения нейросеть запоминает временные и пространственные параметры моделируемого мира, в результате чего один из компонентов обученной модели можно использовать для обучения агентов без непосредственного взаимодействия с моделируемым миром. Авторы ссылаются на схожесть их подхода с механизмом человеческих сновидений.

Архитектура нейросети состоит из трех подсетей. Одна из них – это вариационный автокодировщик [15] (VAE – Variational AutoEncoder), преобразующий наблюдения агента при взаимодействии с моделируемым миром в вектора скрытого пространства признаков (V). Вторая – рекуррентная нейросеть (M), обучающаяся делать вероятностные прогнозы векторов этого пространства, которые ожидает получить от автокодировщика с целью распознавания динамики в статичных изображениях мира. Третья сеть – простая линейная однослойная

нейросеть (C), использующая результаты предсказаний для выбора действий агента (см. рис. 1).

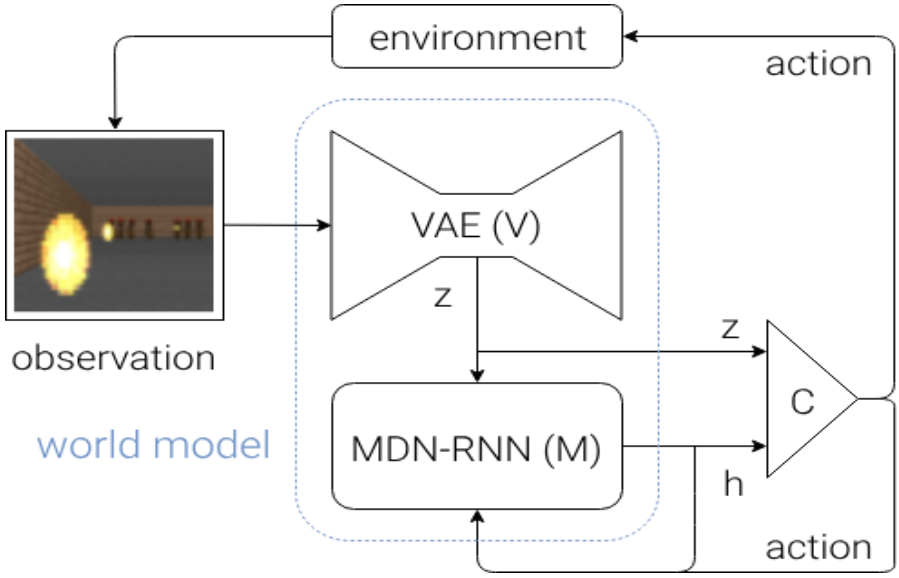



Рис. 1. Архитектура модели World Models [14]

Совокупность первых двух моделей и есть модель мира, сформированная в процессе обучения агента. Спрогнозированные моделью мира вектора скрытого пространства используются как входные данные для нового прогнозирования в той же модели. Таким образом нейросеть «воображает» сценарии из мира, в котором обучалась.

Авторы статьи предложили использовать этот подход для обучения других агентов методом обучения с подкреплением. Учитывая, что нейросетевая модель мира обучалась делать предсказания на своих собственных действиях, она способна заменить симулируемый мир при обучении других агентов. При этом результаты обучения на нейросетевой симуляции можно использовать уже для тестирования агента в обычной симуляции, а автокодировщик позволяет сжимать пространство признаков исходного мира в скрытое пространство меньшей размерности, упрощая процесс обучения, в результате чего этот подход стал применяться в робототехнике.

**PlaNet**

Исследователи Google AI совместно с DeepMind доработали систему обучения агентов с использованием нейросетевых симуляций. Изначальный подход предполагал использование изображений для прогнозирования дальнейших

действий агента. Использование нейросетевой симуляции подразумевало постоянное кодирование и декодирование этих изображений, которые затем формировали воображаемые нейросетью сценарии взаимодействия [16].

Архитектура *PlaNet* использует то же самое скрытое пространство, в котором в процессе обучения формируется пространственно-временная модель реального мира, полученная благодаря преобразованию наблюдений во время обучения агента. *PlaNet* прогнозирует значения скрытого пространства, а сценарии взаимодействия с миром моделируются нейросетью через эти значения, а не через изображения. Подобный подход позволяет агентам изучать более абстрактные представления, такие как положения и скорости объектов. Это упрощает прогнозирование, исключая необходимость генерации изображений в процессе обучения.

**IRIS**

Следующий шаг в области моделирования миров для генеративных симуляции – это работа 2022 года, в которой представлена модель *IRIS* – агент глубокого обучения, использующий трансформеры для эффективного моделирования мира [17]. В отличие от традиционных методов глубокого обучения, которые требуют значительных объемов данных для обучения, *IRIS* достигает высокой эффективности при ограниченном количестве взаимодействий с окружающей средой.

Для обучения и тестирования модели авторы выбрали бенчмарк *Atari 100k* [18], содержащий 26 различных игр. Архитектура модели состоит из тех же автокодировщика и трансформера [19] для обработки последовательности данных, то есть наблюдений и действий. Трансформер используется как авторегрессионная модель, делая новые предсказания на основе своих предыдущих предсказаний. Трансформеры позволяют модели работать с большими объёмами информации и учитывать контекст из разных частей последовательности. Механизм внимания позволяет модели фокусироваться на важных аспектах окружающей среды и игнорировать несущественные данные.

*IRIS* достаточно около двух часов игры для обучения. Модель смогла достичь результата 1.046 по метрике Mean Human-Normalized Score. Это не самый высокий результат среди других агентов, например, модель *EfficientZero* [20] пре-

восходит *IRIS* практически в два раза, однако здесь интересно именно само применение трансформеров для моделирования мира.

## ГЕНЕРАТИВНЫЕ СИМУЛЯЦИИ

Успешное развитие разнообразных подходов к моделированию миров для обучения в них агентов породило новую ветку исследований в области генеративных симуляций. Одни из основных вопросов исследователей в этой области – можно ли использовать генеративные нейросетевые модели для создания уникальных и полностью интерактивных миров? Могут ли подобные модели симулировать взаимодействие с этими мирами? Решение этих задач может значительно изменить игровую индустрию в целом, оставив в прошлом привычную разработку игр в специализированном ПО – игровых движках. Далее в статье приведены примеры уже существующих нейросетевых моделей, способных, с некоторыми ограничениями и допущениями, но всё же генерировать и симулировать интерактивные миры.

### DIAMOND

На основе результатов авторов *IRIS*, в 2024 году вышла работа, демонстрирующая результаты генерации уже другой модели – *DIAMOND* (DIffusion As a Model Of eNvironment Dreams) [21]. В ней авторы генерируют интерактивную симуляцию на базе диффузионных моделей для того же *Atari* бенчмарка.

Целью этой работы было показать, как сильно визуальные детали в мировых моделях влияют на последующее обучение агентов внутри них. По сравнению с *IRIS*, *DIAMOND* достигла оценки Mean Human-Normalized Score в 1.46, что уже значительно превышает результаты *IRIS*.

Чтобы модель могла работать в реальном времени, исследователи внесли изменения в архитектуру диффузионной модели. Распространённую DDPM [22] они заменили на модифицированную EDM [23], что позволило снизить количество итераций удаления шума до 3. Это также позволило избежать проблемы аккумуляции ошибки, которая присутствовала в DDPM-подходе, при малом количестве итераций удаления шума. Но наиболее интересным достижением авторов является эксперимент с симуляцией игрового процесса игры *CS:GO* (см. рис. 2).


Рис. 2. Возможности генеративной симуляции модели *DIAMOND* [24]

Цель эксперимента – изучить возможности генеративной симуляции модели уже в трёхмерном мире. В этот раз модель обучалась на специальном датасете, содержащем более 87 часов геймплея на одной из игровых локаций [24]. По результатам эксперимента модель оказалась способной достоверно воспроизводить окружение и реагировать на действия игрока на протяжении 100 шагов, но она испытывает трудности с последовательной генерацией, например, при приближении к стене, когда модель теряет контекст из-за ограничения памяти.

**Genie**

Чуть ранее в том же году Google DeepMind представила *Genie* – генеративное интерактивное окружение. Это модель, способная генерировать полноценные интерактивные игры жанра платформер только из пользовательского текстового описания или изображения [25].

Ключевое достижение исследователей из Google DeepMind – обучение модели только на визуальном наборе данных, без использования меток действий. Это позволило модели самостоятельно выявлять закономерности, что является

важным шагом к созданию универсальных агентов, способных адаптироваться к различным задачам.

Чтобы достичь подобного результата, была использована модель скрытых действий, кодирующая изменения между кадрами в определённое действие. Для этого использовался автокодировщик на основе сетей векторного квантования, что позволило сократить количество кодируемых действий до небольшого дискретного набора кодов (см. рис. 3). В основе модели всё так же лежит трансформер, но с альтернативной пространственно-временной архитектурой (ST-Transformer) [26]. Исследователи также упоминали об использовании другой модификации трансформера для эффективной обработки данных из изображений (Vision transformer) [27].

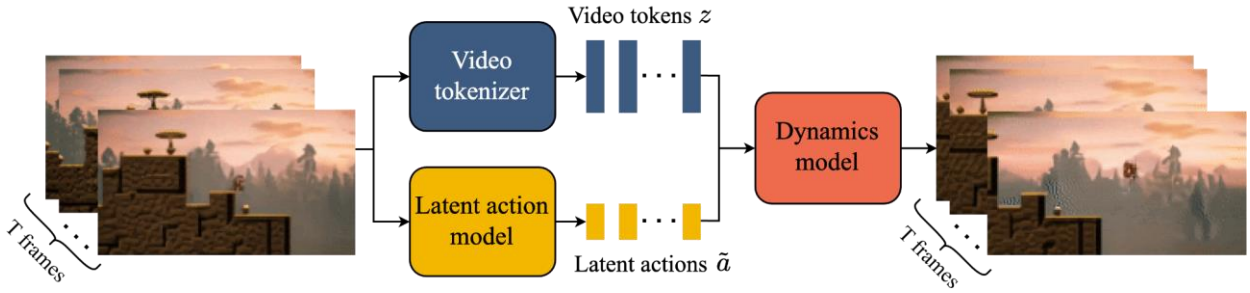



Рис. 3. Архитектура и процесс обучения модели *Genie* [25]

Модель обучалась на более чем 30 тысячах часов (около 6.8 млн. небольших клипов) видеозаписей геймплея из различных 2D-платформеров. Конечное количество параметров модели довольно внушительное – около 10 млрд., хотя и обучение производилось на кластере из 256 TPU[6].

Помимо возможности воспроизводить игровой процесс игр платформеров, исследователи сообщают о некоторых эмерджентных способностях модели. Например, она научилась воспроизводить одну из характерных особенностей игр этого жанра – параллакс игрового фона, что говорит о способности модели воспринимать и трёхмерные пространства.

[6] TPU (сокр. от англ. Tensor Processing Unit) – тензорный процессор, это особый тип процессоров, используемый для нейронных вычислений.

**GameNGen**

Параллельно с предыдущими работами исследователи из Google Research представили свою модель – *GameNGen* [28]. Это генеративная модель для симуляции игрового процесса игры *DOOM*, также созданная на базе диффузионной нейросетевой модели для генерации изображений *Stable Diffusion 1.4* [29]. Для обучения модели разработчики использовали выборку игр, предварительно записанных во время обучения отдельного нейросетевого агента в самой игре.

Диффузионные модели обучаются на тексте и текстовых встраиваниях [30]. Авторы *GameNGen* смогли модифицировать модель для обучения на последовательности предыдущих кадров, используя кодировщик. Он кодирует изображения в скрытом пространстве, сжимая их и уменьшая размерность исходных данных. Очевидно, одних лишь изображений недостаточно для интерактивной симуляции. В этом случае вместо текстовых встраиваний изучались встраивания действий. В игре – это нажатия определённых клавиш. Авторы также адаптировали механизм перекрёстного внимания для работы с закодированной последовательностью действий.

Симуляция игрового процесса происходит в реальном времени, с небольшой, но достаточной для стабильной игры частотой кадров – около 20 кадров в секунду. Предположительно, благодаря изначально низкому разрешению симулируемой игры и использованию предыдущих кадров для предсказания, авторы смогли сократить количество итераций удаления шума до 4, что позволило уменьшить количество вычислительных операций и получить практически такое же качественное изображение, как при 20 итерациях.

Хоть эта модель не является точной симуляцией игры, она способна симулировать сложные игровые механики, такие как подсчёт здоровья и боеприпасов, атаки врагов, повреждения объектов, открытие дверей и сохранение состояния игры на длительных траекториях (см. рис. 4). Интересно, что игровой интерфейс, в котором отображается основная информация о здоровье, боеприпасах и доступном вооружении, сохраняет свою согласованность с игровым процессом на протяжении всей симуляции. Учитывая этот факт, можно предположить, что подобная визуализация части информации о состоянии игрового мира в игровом интерфейсе помогает модели не только выявлять закономерности изменения

среды и интерфейса, но и использовать интерфейс как способ долгосрочного хранения информации.


Рис. 4. Фрагмент игрового процесса, сгенерированного *GameNGen* [30]

Для верификации, помимо отдельных метрик оценки качества генерации, таких как пиковое отношение сигнала к шуму (PSNR) или изученное воспринимаемое сходство участков изображения (LPIPS) [31], авторы также представили статистику человеческих оценок, собранную с помощью отдельного инструмента. Они дали 10-ти кандидатам возможность изучить около сотни небольших клипов длительностью от 1.5 до 3 секунд, сгенерированных моделью, каждый из которых был показан параллельно с таким же клипом из самой игры. Кандидаты должны были отличить, какой из клипов был симуляцией, а какой – фрагментом настоящего игрового процесса. По итогам тестирования оценщики смогли отличить симуляцию только в 58% случаях, но с увеличением длительности клипов результаты ухудшились до 60%.

**Oasis**

Модель *Oasis* представляет собой интерактивную нейросимуляцию с открытым миром, разработанную компанией Decart в сотрудничестве с Etched [32]. *Oasis* генерирует интерактивный мир в реальном времени. Симуляция основана на игре *Minecraft*, которая отличается высокой степенью агентивности игрока. Благодаря этому, каждое действие пользователя непосредственно влияет на происходящее в мире.

Симуляция способна воспроизводить почти все игровые механики исходной игры, от простого перемещения до взаимодействия с игровыми персонажами и изменения игрового окружения игроком. Модель также научилась симулировать взаимодействие с игровыми интерфейсами. Компания выложила демонстрационную версию модели в открытый доступ.

*Oasis* основана на диффузионных моделях, которые зарекомендовали себя как эффективные инструменты для генерации изображений и видео. В отличие от традиционных диффузионных моделей, *Oasis* использует трансформеры, что позволяет учитывать контекст предыдущих кадров. Архитектура включает дополнительные временные слои внимания, интегрированные между пространственными слоями внимания, что обеспечивает контекстуализацию на основе предыдущих кадров. Диффузия выполняется в скрытом пространстве, созданном с помощью вариационного автокодировщика Vision Transformer (ViT VAE) [33], что позволяет сосредоточиться на более высокоуровневых характеристиках изображения (см. рис. 5).

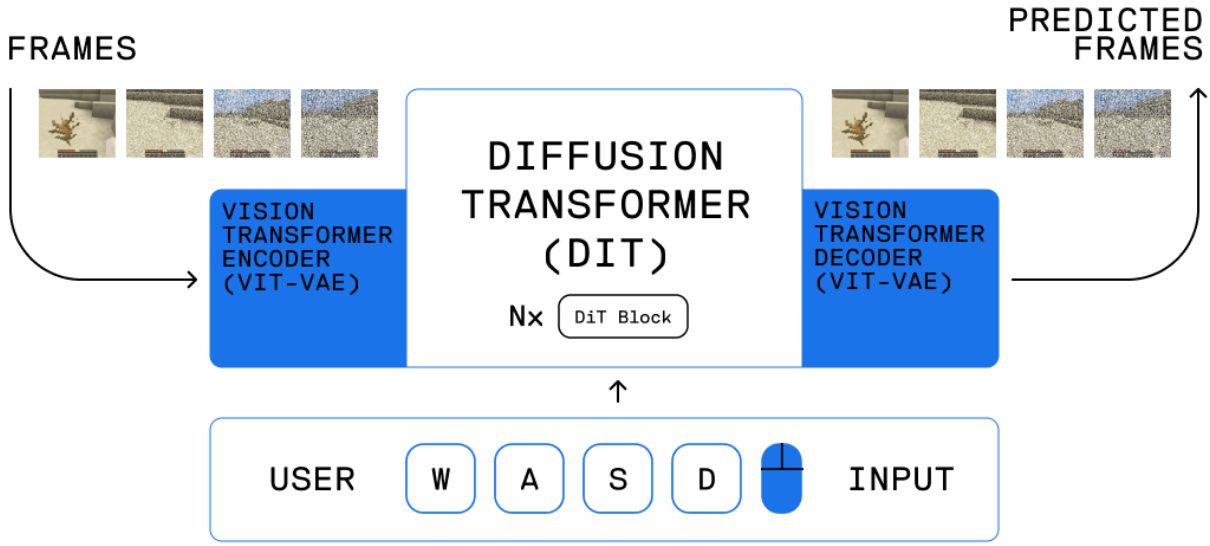



Рис. 5. Архитектура *Oasis* [33]

Подобная трансформерная архитектура выбрана не случайно. Компания хочет протестировать модель на своих интегральных схемах особого назначения (ASIC) Sohu[7], предназначенных для работы с трансформерами. Авторы надеются

[7] https://etched.com/announcing-etched

увеличить возможности генерации до 4К разрешения. На данный момент модель способна генерировать 20 кадров симуляции в секунду в разрешении 360 p и только на специальном тензорном процессоре NVIDIA H100[8].

## ОГРАНИЧЕНИЯ ГЕНЕРАТИВНЫХ СИМУЛЯЦИЙ

Несмотря на впечатляющие результаты, модели генеративных симуляций – это сравнительно новая область исследований. Эксперименты по выявлению эффективных архитектур моделей, подходов к обучению все еще продолжаются, а современные прототипы для симуляции игровых процессов очень далеки от популярных игровых движков, которые они и стремятся заменить. Как минимум, для обучения модели нужно симулировать игровой процесс, этот самый игровой процесс необходимо смоделировать или использовать уже готовую игру, как это было в большей части рассмотренных ранее работ.

Качество симуляции всех моделей страдает от одних и тех же проблем: аккумуляция ошибки и ограничение памяти. Большинство работ признает наличие этих проблем, а где-то исследователи уже демонстрируют свои решения, что мы и рассмотрим в этом разделе.

### Аккумуляция ошибки

Из-за своей авторегрессионной природы модели для генеративных симуляций страдают от проблемы аккумуляции ошибки. Например, авторы *GameNGen* отмечают, что статичное изображение начинает деградировать на 20–30 шаге симуляции, если игрок не перемещается (рис. 6). Авторы *DIAMOND* отмечали, что этот эффект происходит, начиная с 50-го шага генерации уже динамичного изображения в двухмерных играх. *Oasis* демонстрирует нестабильную работу при статичной симуляции мира с некоторыми структурами и не всегда справляется с игровыми интерфейсами.

[8] https://nvidia.com/en-us/data-center/h100/

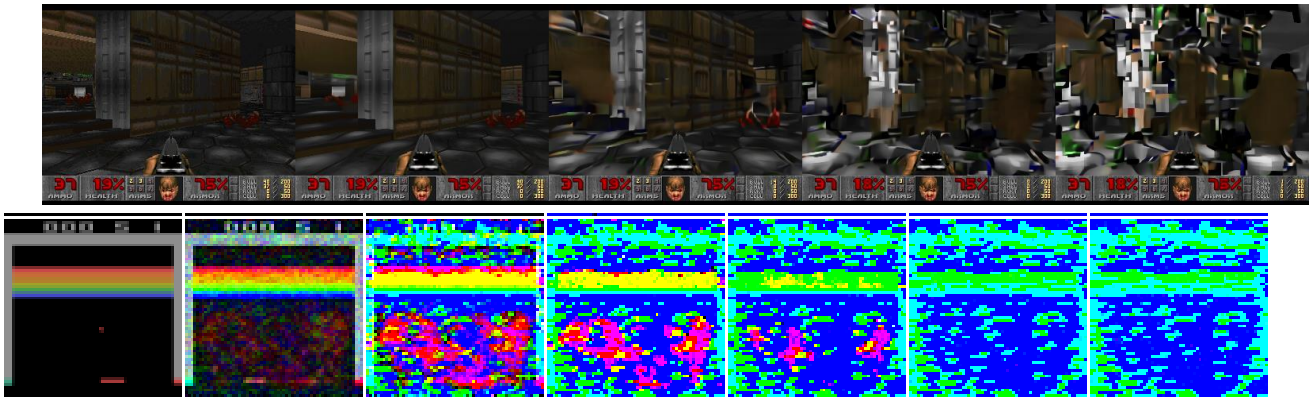


Рис. 6. Визуальные артефакты, появляющиеся в результате аккумуляции ошибки в моделях *GameNGen* (сверху) и *DIAMOND* (снизу)

Авторы каждой из упомянутых моделей по-своему решили проблему аккумуляции. *GameNGen* использует механизм увеличения шума. Определённое количество гауссовского шума добавляется к закодированным кадрам во время обучения, а уровень шума предоставляется модели в качестве входных данных [34]. Разработчики *GameNGen* сообщают, что подобный подход позволил им добиться стабильной продолжительной генерации без потерь качества изображения.

В *DIAMOND* с этой целью использована альтернативная диффузионная модель генерации изображений (EDM). Помимо стабилизации изображения на длительной дистанции, она также позволяет сократить число итераций удаления шума.

*Oasis* решили названную проблему, внедрив систему динамического шумоподавления, которая регулирует шум во время прогнозирования, добавляя шум при первых прямых проходах диффузии, чтобы уменьшить накопление ошибок, и постепенно удаляя шум при последующих проходах, чтобы модель могла находить и сохранять высокочастотные детали в предыдущих кадрах для повышения последовательной генерации.

**Ограничение памяти**

Вторая гораздо более крупная проблема подобных моделей – ограниченная память. К настоящему моменту результаты генераций всех подобных моделей визуально воспринимаются как сон. Детали на статичных изображениях по-

стоянно изменяются, динамичное изменение окружения сохраняет свою консистентность лишь в пределах поля зрения игрока. Ни одна из моделей не способна на последовательную генерацию, и ограничение памяти проявляется в симуляциях по-разному.

Модели, симулирующие трехмерное пространство (*GameNGen*, *DIAMOND*, *Oasis*), не способны долгосрочно запоминать сгенерированное окружение. В *Oasis*, как только часть окружения скрывается из поля зрения игрока, модель забывает его и генерирует новое, если игрок вернёт камеру в исходное положение. Чуть меньше эта проблема проявляется в *DIAMOND* и *GameNGen*. Там это происходит, когда игрок подходит вплотную к стене или смотрит в пол или в небо. Дело в том, что *GameNGen* и *DIAMOND* обучались на играх с фиксированным игровым окружением и обучились достоверно прогнозировать его по предыдущим кадрам, в то время как *Oasis* пытается симулировать открытый процедурно сгенерированный мир (см. рис. 7).

Рис. 7. Демонстрация ограничения памяти на примере *Oasis:* модель «забывает» сгенерированный мир, когда он покидает поле зрения камеры

В попытке решить проблему без кардинального изменения архитектуры Google DeepMind представила вторую версию модели *Genie*, обученную уже на трёхмерных играх различных жанров. Эта модель уже способна на последовательную симуляцию целой минуты игрового процесса [35]. По большей части это связано с увеличением размера самой модели, огромному объему обучающей выборки и высокими вычислительными мощностями, доступным компании. Несмотря на это, в некоторых примерах игрового процесса, сгенерированного *Genie 2*, способность модели запоминать окружение не превосходит уровень предыдущих моделей.

## ЗАКЛЮЧЕНИЕ

На основе рассмотренных работ можно отметить, что существуют следующие направления для дальнейших исследований:

- **Память.** Решение проблемы с ограничениями памяти модели может быть неплохой отправной точкой для стабильной генерации окружения. Вероятно, это потребует пересмотра самой архитектуры моделей и нейросетей, что уже создает большое поле для экспериментов.
- **Обобщение генеративной модели на другие игры.** Следующий шаг к полноценной генерации игрового окружения – это обучение обобщенных моделей, способных стимулировать игровые процессы не только конкретной видеоигры, но и целого отдельного класса видеоигр.
- **Тестирование.** Обучение с подкреплением можно использовать для автоматизации процесса тестировании видеоигр.

В ходе изучения возможностей существующих моделей генеративных симуляций мы выделили основную проблему существующих моделей – ограничение памяти. Для решения этой проблемы мы провели детальное сравнение описанных архитектур, отобрав наиболее важные компоненты каждой. Эти компоненты составят основу модели, на которой мы будем проводить дальнейшие эксперименты, с целью выявить архитектуру, позволяющую качественно улучшить память моделей генеративных симуляций.

Изучение возможностей генеративных нейросетевых симуляций продемонстрировало огромный потенциал для создания интерактивных и динамических виртуальных миров, что может радикально изменить не только игровую индустрию, но и подходы к моделированию реальных процессов. Модели на основе нейросетевых миров и генеративных сетей позволяют успешно имитировать сложные взаимодействия с окружающей средой.

Однако, несмотря на значительный прогресс, текущие технологии сталкиваются с рядом проблем, таких как аккумуляция ошибки и ограничения памяти, что ограничивает возможности их применения в долгосрочных и сложных симуляциях. Решение этих проблем в долгосрочной перспективе может сделать модели нейросетевых генеративных симуляций новым стандартом игровой разработки и даже вытеснить игровые движки.

**ЛУДОГРАФИЯ**[9]

Blizzard Entertainment. (2016) [2010.] *Starcraft II*. [видеоигра][загрузка][Windows, macOS]. Blizzard Entertainment.

Id Software. (1993) [1993.] *DOOM*. [видеоигра][загрузка, CD][DOS, Windows, macOS, Linux]. Id Software.

Mojang Studios. (2025) [2011.] *Minecraft* [видеоигра][загрузка][Windows, macOS, Linux, Android, iOS, iPadOS, Xbox 360, Raspberry Pi, Windows Phone, PlayStation 3, Fire OS, PlayStation 4, Xbox One, PlayStation Vita, Wii U, Apple TV, tvOS, Samsung Gear VR, Nintendo Switch, New Nintendo 3DS, PlayStation VR]. Mojang Studios, Xbox Game Studios, Sony Interactive Entertainment.

Valve Corporation. (2025) [2013.] *Dota 2* [видеоигра][загрузка][Windows, macOS, Linux] Valve Corporation.

Valve Corporation, Hidden Path Entertainment. (2023) [2012.] *Counter Strike: Global Offensive* [видеоигра][загрузка][Windows, macOS, Linux] Valve Corporation.

**СПИСОК ЛИТЕРАТУРЫ**

1. *Fayaz S.A. et al.* Machine learning: An introduction to reinforcement learning // Machine Learning and Data Science: Fundamentals and Applications. 2022. P. 1–22.

2. *Mnih V. et al.* Human-level control through deep reinforcement learning // Nature. 2015. Vol. 518. No. 7540. P. 529–533.

3. *Silver D. et al.* Mastering the game of go without human knowledge // Nature. 2017. Vol. 550. No. 7676. P. 354–359.

4. *Silver D. et al.* Mastering the game of Go with deep neural networks and tree search // Nature. 2016. Vol. 529. No. 7587. P. 484–489.

5. *Vinyals O. et al.* Grandmaster level in StarCraft II using multi-agent reinforcement learning // Nature. 2019. Vol. 575. No. 7782. P. 350–354.

6. *Berner C. et al.* Dota 2 with large scale deep reinforcement learning // arXiv

[9] Лудография (от лат. ludus – «игра»и греч. graphō (γράφω) – «пишу») – список игр, упоминаемых в тексте, который отделён от библиографии. Здесь описание каждой игры представлено в формате, предложенном в рекомендациях http://semionerd.blogspot.com/p/how-to-write-ludography.html.

preprint arXiv:1912.06680. 2019.

7. *Сахибгареева Г.Ф., Кугуракова В.В., Большаков Э.С.* Инструменты балансирования игр // Электронные библиотеки. 2023. Т. 26. No. 2. С. 225–251.

8. *Rani G. et al.* A deep reinforcement learning technique for bug detection in video games // International Journal of Information Technology. 2023. Т. 15. No.. 1. С. 355–367.

9. *Wiering M.A., Van Otterlo M.* Reinforcement learning // Adaptation, learning, and optimization. 2012. Vol. 12. No. 3. P. 729.

10. *Майн Х., Осаки С.* Марковские процессы принятия решений. 1977.

11. *Spaan M.T.J.* Partially observable Markov decision processes. //Reinforcement learning: State-of-the-art. Berlin, Heidelberg: Springer Berlin Heidelberg. 2012. P. 387–414.

12. *Cai Q. et al.* A survey on deep reinforcement learning for data processing and analytics // IEEE Transactions on Knowledge and Data Engineering. 2022. Vol. 35. No. 5. P. 4446–4465.

13. *Moerland T.M. et al.* Model-based reinforcement learning: A survey //Foundations and Trends in Machine Learning. 2023. Vol. 16. No. 1. P. 1–118.

14. *Ha D., Schmidhuber J.* World models // arXiv preprint arXiv:1803.10122. 2018.

15. *Pinheiro Cinelli L. et al.* Variational autoencoder // Variational methods for machine learning with applications to deep networks. Cham: Springer International Publishing. 2021. P. 111–149.

16. *Hafner D. et al.* Learning latent dynamics for planning from pixels // International conference on machine learning. PMLR. 2019. P. 2555–2565.

17. *Micheli V., Alonso E., Fleuret F.* Transformers are sample-efficient world models // arXiv preprint arXiv:2209.00588. 2022.

18. *Kaiser L. et al.* Model-based reinforcement learning for Atari // arXiv preprint arXiv:1903.00374. 2019.

19. *Vaswani A. et al.* Attention is all you need // Advances in neural information processing systems. 2017. Vol. 30.

20. *Ye W. et al.* Mastering Atari games with limited data // Advances in neural information processing systems. 2021. Vol. 34. P. 25476–25488.

21. *Alonso E. et al.* Diffusion for world modeling: Visual details matter in Atari // Advances in Neural Information Processing Systems. 2024. Vol. 37. P. 58757–58791.

22. *Ho J., Jain A., Abbeel P.* Denoising diffusion probabilistic models // Advances in neural information processing systems. 2020. Vol. 33. P. 6840–6851.

23. *Karras T. et al.* Elucidating the design space of diffusion-based generative models // Advances in neural information processing systems. 2022. Vol. 35. P. 26565–26577.

24. *Pearce T., Zhu J.* Counter-strike deathmatch with large-scale behavioural cloning // 2022 IEEE Conference on Games (CoG). IEEE, 2022. P. 104–111.

25. *Bruce J. et al.* Genie: Generative interactive environments // Forty-first International Conference on Machine Learning. 2024.

26. *Xu M. et al.* Spatial-temporal transformer networks for traffic flow forecasting //arXiv preprint arXiv:2001.02908. 2020.

27. *Dosovitskiy A. et al.* An image is worth 16x16 words: Transformers for image recognition at scale // arXiv preprint arXiv:2010.11929. 2020.

28. *Valevski D. et al.* Diffusion models are real-time game engines // The Thirteenth International Conference on Learning Representations. 2024.

29. *Rombach R. et al.* High-resolution image synthesis with latent diffusion models // Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2022. P. 10684–10695.

30. *Yu H. et al.* Uncovering the text embedding in text-to-image diffusion models // arXiv preprint arXiv:2404.01154. 2024.

31. *Zhang R. et al.* The unreasonable effectiveness of deep features as a perceptual metric // Proceedings of the IEEE conference on computer vision and pattern recognition. 2018. P. 586–595.

32. *Decart E. et al.* Oasis: A universe in a transformer.
URL: https://oasis-model.github.io/

33. *Choi B., Jeong J.* ViV-Ano: Anomaly detection and localization combining vision transformer and variational autoencoder in the manufacturing process // Electronics. 2022. Vol. 11. No. 15. P. 2306.

34. *Pasini M. et al.* Continuous autoregressive models with noise augmentation avoid error accumulation // arXiv preprint arXiv:2411.18447. 2024.

35. *Parker-Holder J. et al.* Genie 2: A Large-Scale Foundation World Model. URL: https://deepmind.google/discover/blog/genie-2-a-large-scale-foundation-world-model/

## REAL-TIME GENERATIVE SIMULATION OF GAME ENVIRONMENT

**E. S. Bolshakov[1] [0000-0002-2208-9515], V. V. Kugurakova[2] [0000-0002-1552-4910]**

[1, 2]*Institute of Information Technologies and Intelligent Systems, Kazan (Volga Region) Federal University, ul. Kremlyovskaya, 35, Kazan, 420008*

[1]edward.bolshakov117@gmail.com, [2]vlada.kugurakova@gmail.com

***Abstract***

This paper explores the potential of generative neural network simulations, focusing on the application of reinforcement learning methods and neural world models for creating interactive worlds. Key achievements in agent training using reinforcement learning are discussed. Special attention is given to neural world models, as well as generative models such as Oasis, DIAMOND, Genie, and GameNGen, which employ diffusion networks to generate realistic and interactive game worlds. The opportunities and limitations of generative simulation models are examined, including issues related to error accumulation and memory constraints, and their impact on the quality of generation. The conclusion presents suggestions for future research directions.

***Keywords****: video games, game environment, generative simulation, reinforcement learning, generative neural networks, gameplay simulation, world models.*

**LUDOGRAPHY**

Blizzard Entertainment. (2016) [2010.] *Starcraft II*. [видеоигра][загрузка][Windows, macOS]. Blizzard Entertainment.

Id Software. (1993) [1993.] *DOOM*. [видеоигра][загрузка, CD][DOS, Windows, macOS, Linux]. Id Software.

Mojang Studios. (2025) [2011.] *Minecraft* [видеоигра][загрузка][Windows, macOS, Linux, Android, iOS, iPadOS, Xbox 360, Raspberry Pi, Windows Phone, PlayStation 3, Fire OS, PlayStation 4, Xbox One, PlayStation Vita, Wii U, Apple TV, tvOS, Samsung Gear VR, Nintendo Switch, New Nintendo 3DS, PlayStation VR]. Mojang Studios, Xbox Game Studios, Sony Interactive Entertainment.

Valve Corporation. (2025) [2013.] *Dota 2* [видеоигра][загрузка][Windows, macOS, Linux] Valve Corporation.

Valve Corporation, Hidden Path Entertainment. (2023) [2012.] *Counter Strike: Global Offensive* [видеоигра][загрузка][Windows, macOS, Linux] Valve Corporation.

**REFERENCES**

1. *Fayaz S.A. et al.* Machine learning: An introduction to reinforcement learning // Machine Learning and Data Science: Fundamentals and Applications. 2022. P. 1–22.

2. *Mnih V. et al.* Human-level control through deep reinforcement learning // Nature. 2015. Vol. 518. No. 7540. P. 529–533.

3. *Silver D. et al.* Mastering the game of go without human knowledge // Nature. 2017. Vol. 550. No. 7676. P. 354–359.

4. *Silver D. et al.* Mastering the game of Go with deep neural networks and tree search // Nature. 2016. Vol. 529. No. 7587. P. 484–489.

5. *Vinyals O. et al.* Grandmaster level in StarCraft II using multi-agent reinforcement learning // Nature. 2019. Vol. 575. No. 7782. P. 350–354.

6. *Berner C. et al.* Dota 2 with large scale deep reinforcement learning // arXiv preprint arXiv:1912.06680. 2019.

7. *Sahibgareeva G.F., Kugurakova V.V., Bolshakov E.S.* Game balancing tools [Instrumenti balansirovaniya igr] // Russian Digital Library Journal. 2023. Vol. 26. No. 2. P. 225–251 (in Russian).

8. *Rani G. et al.* A deep reinforcement learning technique for bug detection in video games // International Journal of Information Technology. 2023. Vol. 15. No. 1. P. 355–367.

9. *Wiering M.A., Van Otterlo M.* Reinforcement learning // Adaptation, learning, and optimization. 2012. Vol. 12. No. 3. P. 729.

10. *Main H., Osaki S.* Markovskie processi prinatiya reshenii. 1977.

11. *Spaan M.T.J.* Partially observable Markov decision processes // Reinforcement learning: State-of-the-art. Springer Berlin Heidelberg. 2012. P. 387–414.

12. *Cai Q. et al.* A survey on deep reinforcement learning for data processing and analytics // IEEE Transactions on Knowledge and Data Engineering. 2022. Vol. 35. No. 5. P. 4446–4465.

13. *Moerland T.M. et al.* Model-based reinforcement learning: A survey // Foundations and Trends in Machine Learning. 2023. Vol. 16. No. 1. P. 1–118.

14. *Ha D., Schmidhuber J.* World models // arXiv preprint arXiv:1803.10122. 2018.

15. *Pinheiro Cinelli L. et al.* Variational autoencoder // Variational methods for machine learning with applications to deep networks. Cham: Springer International Publishing. 2021. P. 111–149.

16. *Hafner D. et al.* Learning latent dynamics for planning from pixels // International conference on machine learning. PMLR. 2019. P. 2555–2565.

17. *Micheli V., Alonso E., Fleuret F.* Transformers are sample-efficient world models // arXiv preprint arXiv:2209.00588. 2022.

18. *Kaiser L. et al.* Model-based reinforcement learning for Atari // arXiv preprint arXiv:1903.00374. 2019.

19. *Vaswani A. et al.* Attention is all you need // Advances in neural information processing systems. 2017. Vol. 30.

20. *Ye W. et al.* Mastering Atari games with limited data // Advances in neural information processing systems. 2021. Vol. 34. P. 25476–25488.

21. *Alonso E. et al.* Diffusion for world modeling: Visual details matter in Atari // Advances in Neural Information Processing Systems. 2024. Vol. 37. P. 58757–58791.

22. *Ho J., Jain A., Abbeel P.* Denoising diffusion probabilistic models // Advances in neural information processing systems. 2020. Vol. 33. P. 6840–6851.

23. *Karras T. et al.* Elucidating the design space of diffusion-based generative models // Advances in neural information processing systems. 2022. Vol. 35. P. 26565–26577.

24. *Pearce T., Zhu J.* Counter-strike deathmatch with large-scale behavioural cloning // 2022 IEEE Conference on Games (CoG). IEEE, 2022. P. 104–111.

25. *Bruce J. et al.* Genie: Generative interactive environments // Forty-first International Conference on Machine Learning. 2024.

26. *Xu M. et al.* Spatial-temporal transformer networks for traffic flow forecasting // arXiv preprint arXiv:2001.02908. 2020.

27. *Dosovitskiy A. et al.* An image is worth 16x16 words: Transformers for image recognition at scale // arXiv preprint arXiv:2010.11929. 2020.

28. *Valevski D. et al.* Diffusion models are real-time game engines // The Thirteenth International Conference on Learning Representations. 2024.

29. *Rombach R. et al.* High-resolution image synthesis with latent diffusion models // Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2022. P. 10684–10695.

30. *Yu H. et al.* Uncovering the text embedding in text-to-image diffusion models // arXiv preprint arXiv:2404.01154. 2024.

31. *Zhang R. et al.* The unreasonable effectiveness of deep features as a perceptual metric // Proceedings of the IEEE conference on computer vision and pattern recognition. 2018. P. 586–595.

32. *Decart E. et al.* Oasis: A universe in a transformer.
URL: https://oasis-model.github.io/

33. *Choi B., Jeong J.* ViV-Ano: Anomaly detection and localization combining vision transformer and variational autoencoder in the manufacturing process // Electronics. 2022. Vol. 11. No. 15. P. 2306.

34. *Pasini M. et al.* Continuous autoregressive models with noise augmentation avoid error accumulation // arXiv preprint arXiv:2411.18447. 2024.

35. *Parker-Holder J. et al.* Genie 2: A Large-Scale Foundation World Model. URL: https://deepmind.google/discover/blog/genie-2-a-large-scale-foundation-world-model/

## СВЕДЕНИЯ ОБ АВТОРАХ

***БОЛЬШАКОВ Эдуард Сергеевич –*** ассистент кафедры индустрии разработки игр института ИТИС. Сфера научных интересов – игровая сценаристика, игровой дизайн, применение нейросетей в разработке видеоигр.

***Eduard Sergeevich BOLSHAKOV –*** assistant of the Game Development Industry Department at the ITIS Institute. Research interests – game storytelling, game design, application of neural networks in video game development.

email: edward.bolshakov117@gmail.com
ORCID: 0000-0002-2208-9515

***КУГУРАКОВА Влада Владимировна*** — доцент, к. т. н., зав. кафедрой индустрии разработки видеоигр ИТИС КФУ. Сфера научных интересов — различные аспекты проектирования игр.

***Vlada Vladimirovna KUGURAKOVA*** — Associate Professor, Head of the Video Game Development Industry Chair of ITIS KFU. Research interests include various aspects of game design.

email: vlada.kugurakova@gmail.com
ORCID: 0000-0002-1552-4910

*Материал поступил в редакцию 28 января 2025 года*

УДК 004.928+004.85+004.896

# ПРОБЛЕМЫ, РЕШЕНИЯ И ПЕРСПЕКТИВЫ АВТОМАТИЗИРОВАННОГО ПЕРЕНОСА ИГРОВЫХ СЦЕН МЕЖДУ ИГРОВЫМИ ДВИЖКАМИ

**А. О. Бондарь[1] [0009-0005-5296-9686], В. В. Кугуракова[2] [0000-0002-1552-4910]**

[1, 2]*Казанский федеральный университет, Институт информационных технологий и интеллектуальных систем*

[1]alexey.bondar.2013@mail.ru, [2]vlada.kugurakova@gmail.com

***Аннотация***

Рассмотрены технические проблемы переноса игровых сцен между различными игровыми движками. Проанализированы ключевые вызовы, связанные с различиями форматов хранения сцен, несовместимостью API рендеринга и физического моделирования, проблемами конвертации материалов, шейдеров и анимационных данных, а также различиями в системах координат. Представлены существующие инструменты и методы, включая автоматизированные решения для экспорта, преобразования и импорта данных, с особым акцентом на перенос контента из Unreal Engine в Unigine. Дополнительно обсуждены фундаментальные подходы к решению задачи – применение универсальных форматов обмена (FBX, glTF, USD), создание промежуточного слоя (middleware) и модульный дизайн игровых сцен, что открывает перспективы для будущей автоматизации процесса. Приведены результаты исследований по формальному описанию логики игровых систем и подходы к портированию VR-приложений между различными библиотеками. Полученные выводы позволяют сформулировать практические рекомендации для разработчиков и обозначить направления дальнейших исследований в области автоматизированного переноса контента между игровыми движками.

***Ключевые слова:*** *миграция игровых сцен, игровой движок, перенос контента, Unreal Engine, Unity, Unigine, Nau Engine, Godot, CryEngine, конвертация форматов.*

## ВВЕДЕНИЕ

Современная индустрия компьютерных игр характеризуется динамичным развитием технологий, что приводит к появлению новых игровых движков с уникальными возможностями. При этом разработчики часто сталкиваются с необходимостью переноса игровых сцен и ассетов с одних платформ на другие для улучшения визуальных эффектов, повышения производительности или расширения функционала проекта. Миграция контента между игровыми движками (например, такими как *Unreal Engine*, *Unigine*, *Unity* и др., включая российский релиз 2025 года – *Nau Engine*[1]), а также перенос на альтернативные системы (например, *Godot* и *CryEngine*) сопряжены с рядом технических трудностей. Фундаментальность проблемы заключается в том, что базовые архитектурные решения, используемые в каждом движке, существенно различаются: от форматов хранения сцен до особенностей рендеринга, физического моделирования и системы координат. Эти различия обусловлены глубокими концептуальными и техническими особенностями, что делает задачу переноса контента между игровыми движками крайне непростой.

Основные вызовы включают несовместимость проприетарных форматов сцен, различия в API для рендеринга и физики, а также проблемы конвертации шейдеров, материалов и анимационных данных, что требует значительных усилий для сохранения первоначального качества проекта. Фундаментальные подходы к решению названных проблем варьируются от использования универсальных форматов обмена (FBX, glTF, USD) до создания промежуточных слоев (middleware), которые абстрагируют данные от специфики конкретного движка, а также от разработки специализированных плагинов для прямого конвертирования. Кроме того, модульный дизайн игровых сцен позволяет частично решить проблему переноса, сегментируя проект на независимые компоненты. Различные инструменты и плагины (например, экспортеры из *Unity* в *Unreal* или *Unigine*)

[1] Nau Engine – специализированный игровой движок, разработанный для создания трехмерных интерактивных приложений и видеоигр на принципах открытого кода, доступный для разработчиков любой квалификации, http://nauengine.org/.

позволяют автоматизировать часть этапов миграции, однако до полного автоматизированного решения пока далеко.

В настоящей статье представлены результаты анализа существующих методов переноса игровых сцен, приведены примеры успешной и проблемной миграций, а также рассмотрены подходы к стандартизации обмена данными (например, использование форматов glTF, FBX, USD). Особое внимание уделено исследованиям нашей группы, посвященным вопросам автоматизации портирования VR-приложений и верификации игровых систем без зависимости от сред разработки.

Цель работы – сформировать практические рекомендации для разработчиков и обозначить направления дальнейших исследований в области переноса контента между игровыми движками.

## ОСНОВНЫЕ СЛОЖНОСТИ МИГРАЦИИ СЦЕН МЕЖДУ ДВИЖКАМИ

### Различия в форматах сцен и моделей

Каждый игровой движок[2] хранит сцены и объекты в собственных проприетарных форматах, что затрудняет прямой перенос. Например, *Unity* хранит сцены в формате .unity (YAML/бинарный), *Unreal Engine* – в .umap/.uasset, *Godot* – в .tscn/.scn, и ни один из них не умеет напрямую читать формат другого. Модели и ресурсы тоже часто упакованы по-разному. В результате невозможно открыть сцену из *Unity* прямо в *Unreal* или *Godot* без конвертации. Требуется экспортировать содержимое в промежуточный формат (FBX, glTF и др.) и заново импортировать в целевой движок, потеряв при этом специфические настройки сцены [1].

[2] Игровой движок (англ. game engine) – комплексная программная платформа, служащая основой для проектирования и реализации компьютерных игр. Представляет собой интегрированный набор модулей, обеспечивающих ключевую функциональность игровых приложений: рендеринг визуальных элементов, симуляцию физических процессов, распределение вычислительных ресурсов, аудиовоспроизведение, исполнение программных сценариев, анимационные системы, элементы машинного интеллекта, сетевое взаимодействие и координацию игровой механики. Ведущие современные решения в этой области – Unreal Engine, Unity, CryEngine, Unigine – функционируют как многофункциональные среды разработки, предоставляющие инструментарий для создания иммерсивного интерактивного опыта.

Даже стандартные форматы – не являются универсальным средством: раньше пытались использовать Collada, FBX и др., но столкнулись с неполной поддержкой. Как отмечает сооснователь Godot Хуан Линиетски, старые форматы были неоднозначны (например, разные единицы измерения и оси координат) и сложны в парсинге, что приводило к огромной работе при переносе сцены [2]. Таким образом, структурные различия форматов – первый серьезный барьер миграции.

**Несовместимость API рендеринга и физики**

Сцену недостаточно перенести геометрически – нужно, чтобы в новом движке она визуализировалась и продолжала работать. Здесь проявляются фундаментальные различия в *системах* рендеринга, освещения и физическом движке. У каждого движка свои API и набор компонентов: то, что было скриптом MonoBehaviour в *Unity*, это в *Unreal* нужно реализовать через C++ класс или Blueprint, т. к. прямых аналогов методов может не быть [3]. Например, у Unity есть события Update/FixedUpdate, а в *Unreal* нет их прямого аналога – разработчик Reddit с опытом в обоих движках отмечает, что *«в Unreal нет аналогов событий для каждого поведения MonoBehaviour... возможно, вам даже придется реализовать это в исходниках»*[3].

С учетом физических свойств объектов аналогично: *Unity* долгое время использовала PhysX, *Unreal* – тоже PhysX (до перехода на собственный Chaos), *Godot* – Bullet, у *CryEngine* – своя физика. Физические параметры (масса, сила, материя) и поведение столкновений могут отличаться. Даже если модель сцены перенесена, настроенные под один физический движок объекты в другом могут вести себя иначе. Инкапсуляция логики AI, триггеров, звука – всё это привязано к API исходного движка и требует переработки под целевой. Таким образом, код и ин-

[3] Точная цитата: “Someone suggested using .Net 6 integration for the code but that still doesn't even get close to what you need in order to actually convert MonoBehavoirs to UE, you'd need all of that class overhead implemented as well, but the problem with that is that at its core, Unreal doesn't have analogous events and functionality for every monobehaviour event and functionality, you might even have to implement that in source code as well”.

терактивная логика практически не мигрируют автоматически – их приходится переписывать вручную или создавать прослойки-эмуляции, что крайне сложно на практике.

**Проблемы с миграцией шейдеров, текстур и материалов**

Визуальный облик сцены при переносе страдает из-за несовместимости систем материалов. Каждый движок имеет собственный язык шейдеров и набор параметров материала. *Unity* использует ShaderLab/HLSL (Standard Shader или URP/HDRP), *Unreal* – свою нодовую систему (материалы на HLSL) с другими названиями свойств, *Godot* – Godot Shader Language (близка к GLSL).

Кастомные шейдеры практически невозможно конвертировать автоматически: их логика заточена под конкретный рендер-пайплайн. Так, конвертер Project Exodus для Unity→Unreal прямо указывает, что *нестандартные шейдеры не поддерживаются* – будет попытка перенести их параметры, но материал в Unreal, скорее всего, получится черным [4]. Только стандартные PBR-материалы переносятся относительно корректно. Текстуры чаще удаётся перенести (оба движка поддерживают PNG/JPG/TGA), но форматы, специфичные для одного движка, могут не приниматься другим. Например, *Unreal* не читает .tif-файлы из *Unity* напрямую [5]. Кроме того, разные движки используют разное сглаживание нормалей и тангентное пространство. Документация *Unigine* отмечает, что требуется пересчет касательного пространства[4], потому что собственный метод расчета нормалей не полностью совместим с *Unity* [6] – иначе материалы с нормальной картой (normal map) будут отображаться неправильно. Таким образом, материалы/шейдеры требуют ручной адаптации: перенастроить значения Roughness/Metallic (например, *Unity* использует Smoothness, а в *Unigine/Unreal* аналог – Roughness), заменить шейдеры на ближайшие стандартные и т. д. Зачастую проще заново настроить освещение и пост-эффекты в новом движке, чем конвертировать предыдущие настройки.

[4] Точная цитата “UNIGINE uses its own tangent space for normal mapping which is not fully compatible with the Unity one. It is possible to import tangent space right from an FBX file, however, different Digital Content Creation tools use different tangent spaces”

**Различия в системах координат и преобразованиях**

Пространственные параметры объектов – положение, масштаб, поворот – могут интерпретироваться по-разному в различных движках. Существуют расхождения по системе координат: *Unity* и *Unreal* используют левостороннюю систему, но в *Unity* ось *Y* направлена вверх, а в *Unreal* – Z вверх по умолчанию. *Godot* в 3D – правосторонняя (*Y* вверх), в 2D – ось *Y* вниз. Такие различия означают, что объекты сцены после переноса могут оказаться повернутыми или отраженными. Конвертеры вынуждены поправлять это автоматически: например, в утилите Unity→Godot инвертируется ось *Y* для 2D-спрайтов [7], иначе сцена «перевернётся». Дополнительная проблема – порядок применения вращения и масштаба. Если в исходном движке у объекта имелись *неравномерный масштаб и поворот родителя*, то после экспорта в FBX/глобальное пространство и импорта в другой движок может произойти искажение (shearing). Так, автор плагина Utu (Unity→Unreal) отмечает, что при наличии неравномерного масштаба (non-uniform scale) у родителя повёрнутого меша (англ. mesh – сетка 3D-модели) результат в *Unreal* окажется неправильным из-за такого способа учета трансформаций [5]. Многие инструменты поэтому предупреждают: сложные иерархии с масштабированием лучше «обнулить» перед переносом. Различия могут быть и в единицах измерения (хотя сейчас обычно 1 единица = 1 метр, но исторически могли быть другие настройки). В совокупности несовпадение систем координат приводит к необходимости конвертировать геометрию: скрипты экспорта поворачивают модели, пересчитывают местные оси, иногда добавляют пустые узлы для корректировки ориентации. Без этого сцена перенесётся криво – объекты могут оказаться не на тех местах или с неверным масштабом.

**Перенос анимационных данных и логики игры**

Анимации персонажей и объектов – ещё одна болевая точка. Скелетные анимации хранятся в разных форматах (*Unity* использует .anim контроллеры или аниматор с машиной состояний (state machine), *Unreal* – Animation Blueprints/Sequences, *Godot* – .tres анимации или AnimationPlayer). Форматы анимаций несовместимы, поэтому приходится экспортировать анимации в FBX или

glTF, теряя при этом настроенные машины состояний и логику переходов. Инструменты автоматизации могут перенести сырые клипы: например, Project Exodus пытается экспортировать Animation Clips, но не воссоздает машины состояний контроллера *Unity* в *Unreal* [4]. Таким образом, последовательности ключевых кадров могут перенестись, а вот логика проигрывания – нет. Кроме того, привязка анимаций к объектам может сломаться: различные движки по-разному именуют кости скелета, по-разному поддерживают IK-решения, морф-цели и т. д. Даже базовые вещи, например кривые анимации для свойств, могут потребовать ручной настройки заново, если название или диапазон свойств отличается. Что касается логики игры (AI, геймплейные скрипты, триггеры событий) – их переносить сложнее всего. Как отмечалось, прямого конвертера кода, который перепишет C# скрипты *Unity* в Blueprints *Unreal* или GDScript *Godot*, не существует на промышленном уровне. В научных кругах высказывались идеи парсить код *Unity* и автоматически генерировать аналогичный для другого движка [3], но реалисты отмечают, что это *«почти гарантированно не работает»* из-за тотальных различий API и сложностей поддержки. Поэтому на практике обычно лишь переносятся статика сцены и анимации, а скрипты разработчики пишут заново под новый движок. В итоге полноценная миграция динамичного проекта – чрезвычайно трудоемкий процесс, почти равносильный разработке с нуля.

## СУЩЕСТВУЮЩИЕ ИНСТРУМЕНТЫ ДЛЯ ПЕРЕНОСА СЦЕН

Несмотря на сложности, существуют частичные автоматизированные решения (как открытые, так и коммерческие), облегчающие перенос игровых сцен между движками.

### Экспорт в промежуточный формат

Базовый подход – выгрузить все объекты сцены в универсальный 3D-формат. Например, ***Khronos glTF 2.0*** сейчас широко признан удобным обменным форматом для 3D-ассетов. В [2] отмечено, что glTF 2.0 дает шанс стандартизировать перенос контента между инструментами и движками. Многие движки поддерживают glTF: Godot умеет импортировать полные сцены glTF (включая меши, мате-

риалы PBR и анимации), *Unreal Engine* имеет плагин USD/glTF Importer, *Unity* – экспорт пакет FBX Exporter (который умеет и в glTF через UnityGLTF).

***FBX*** – еще один де-факто стандарт: *Unity* и *Unreal* его поддерживают из коробки. Однако эти форматы в основном переносят геометрию, иерархию и базовые материалы. Специфичные эффекты, логика упускаются.

Существуют также специализированные форматы, например ***Open Game Engine Exchange (OpenGEX)*** – текстовый формат, созданный для переноса «сложных данных сцены» между инструментами [8]. OpenGEX пытается устранить недостатки Collada и охватывает множество функций (иерархия узлов, инстанцирование объектов, камеры/свет, несколько UV-каналов, анимация ключевых кадров, скелеты и морфы). Однако его поддержка ограничена (в основном энтузиастами и движком C4).

***Universal Scene Description (USD)*** от Pixar тоже набирает популярность как потенциальный универсальный формат сцены, поддерживается Unreal Engine для импорта/экспорта. В целом использование промежуточных форматов – основной подход, но разработчикам все равно приходится дорабатывать сцену вручную после импорта.

**Плагины для прямого конвертирования Unity→Unreal**

Появились утилиты, автоматизирующие перенос из *Unity* в *Unreal* – популярный сценарий из-за смены движка на более мощный.

Open-source проект ***Project Exodus*** переносит Unity-сцену в *Unreal Engine* [4]. Он экспортирует из *Unity* информацию о сцене и воссоздает ее в *Unreal*: статические меши (с координатами и UV), источники света, отражающие пробы, ландшафты, даже skeletal mesh частично. Стандартные материалы *Unity* конвертируются в эквивалентные материалы *Unreal* (цвет, металличность, шероховатость и пр.), текстуры перекодируются, если формат не поддерживается целевым движком. Однако, как отмечается в документации, есть множество ограничений: поддерживаются только стандартные шейдеры (Surface shaders и Shader Graph – нет), префабы *Unity* не преобразуются в Blueprint *Unreal*, ландшафт перенесется с упрощениями (детали травы/деревьев могут отличаться), сложные скелетные сетки могут разбиться на части, а анимационные FSM-контроллеры не конвертируются.

Плагин ***UtU*** (Unity To Unreal) – коммерческое решение с аналогичным подходом. Он заявляет перенос основных типов ассетов: сцену, меши, анимации, материалы, текстуры, префабы, камеры и т. д., воссоздает иерархии сцены и папок. Utu также генерирует логи и предоставляет GUI для отслеживания процесса. В примечаниях честно указано, что плагин не совершенный и таким никогда не будет, учитывая различия движков [5]: после его работы все равно потребуется ручная доработка сцены, просто объем работы будет меньше, чем переносить с нуля. Среди ограничений Utu имеются проблемы с неравномерным масштабом (см. выше), возможная несовместимость некоторых ригов анимации, отсутствие поддержки нестандартных шейдеров (они заменяются на базовый материал с переносом параметров). Тем не менее такие плагины уже позволили многим разработчикам ускорить миграцию – вместо того чтобы вручную переносить сотни объектов, значительная часть работы автоматизируется, особенно по статическому окружению.

**Инструменты для Unity→Unigine**

Движок Unigine, привлекающий своей мощной графикой, также заинтересован в перетягивании проектов. В 2024 году сообществом *Unigine* создан инструмент, сильно облегчающий переход с *Unity* [1]. Он состоит из двух плагинов: экспортера для *Unity* (выгружающего сцену в JSON) и импортера для *Unigine* (читающего JSON и собирающего сцену). Согласно официальному блогу, автоперенос охватывает всю иерархию сцены, включая скачанные и статические меши (с LODами), материалы с прикрепленными текстурами, настройки физики (RigidBody, коллайдеры, слои коллизий) и даже префабы с C# компонентами. Это примечательно – скриптовые компоненты *Unity* переносимых объектов сохраняются в *Unigine* как заглушки (видимо, с помощью аналогичного C# API *Unigine*). Разработчики оговаривают, что после автоматического переноса потребуется ручная доработка – тонкая настройка параметров, переписывание пользовательского кода на API *Unigine*. Но сам факт переноса всех объектов сцены, света и физики дает огромную экономию времени. *Unigine* также выпустила официальный Unity Importer в своем Asset Store, показывая заинтересованность в привлечении

проектов. Таким образом, для миграции на *Unigine* уже есть готовые pipeline-решения. В обратную сторону (Unigine→Unity) подобных инструментов нет, так как они реже требуются.

**Перенос на Godot Engine**

После всплеска интереса к *Godot* (с открытым исходным кодом) в сообществе появились проекты по автоматизации переноса с *Unity*. Существуют экспериментальные скрипты, например, утилита [7], которая пробует конвертировать *Unity*-проект в *Godot*-проект. Она фокусируется на 2D-играх и базовом 3D, автоматически создавая в *Godot* узлы и сцены на основе *Unity*-сцен. Этот конвертер решает множество соответствий: различает 2D- и 3D-объекты (создает Node2D вместо Spatial, если видит 2D-компоненты), разбивает Unity GameObject с несколькими компонентами на дерево Godot-нод (поскольку в *Godot* один скрипт = один узел), помещает компонент RigidBody в родительский узел (т. к. в *Godot* физический узел должен быть наверху, управляющим дочерними узлами), создает заглушки скриптов – пустые GDScript с экспортируемыми переменными, чтобы хотя бы перенести настроенные в инспекторе значения из *Unity*-сцен. Кроме того, учитываются детали вроде различия единиц в 2D (пиксели против условных единиц) – скрипт масштабирует спрайты, компенсируя количество пикселей на единицу равным 100 у *Unity* для соответствия с *Godot*. Однако сам автор утилиты признает, что это лишь пробный вариант (proof-of-concept) и *«абсолютно чудовищное количество фич ещё предстоит поддержать»*. Проект пока не покрывает 3D-рендер, сложные шейдеры, ландшафт сцены (terrain) или анимационные графы. В сообществе *Godot* в целом советуют переносить контент через *glTF/FBX*, а логику - переписывать вручную, вместо полной автоматизации. Тем не менее сама возможность частично сконвертировать сцену (хотя бы статические объекты, базовые скрипты) является большим подспорьем при миграции на открытый движок.

**Прочие инструменты и подходы**

Для некоторых пар движков существуют свои утилиты. Например, разработчики ***CryEngine/Lumberyard/O3DE*** – движков с общим наследием – предоставляют путь миграции: проекты CryEngine V можно относительно легко открыть в

Amazon Lumberyard (который был форком *CryEngine*) [9]. В частности, Amazon Lumberyard (ныне Open 3D Engine) поддерживал импорт пакетов из *CryEngine*, поэтому студия Cloud Imperium смогла перенести Star Citizen на Lumberyard примерно за один год с минимальными потерями по ассетам. Этот случай особый – движки были очень схожи. Если же движки совершенно разные, приходится создавать авторские конвертеры. Существуют и универсальные парсеры игровых ресурсов, например утилиты для извлечения контента из игр (Game Extractor и др.), которые могут считывать архивы и форматы некоторых движков. Их применяют энтузиасты, чтобы переносить уровни старых игр на новые движки – фактически реверс-инжиниринг (reverse-engineering) контента. Однако это единичные случаи, требующие ручной работы. В промышленной среде появляются сервисы по портированию: ряд студий (например, Pingle Studio [10], N-iX) предлагает услуги переноса проектов с *Unity* на *Unreal*, создавая для каждого проекта кастомный пайплайн. Обычно они комбинируют автоматический экспорт ассетов и ручную адаптацию кода.

В итоге на сегодняшний день нет универсального «конвертера всего», но есть множество специализированных решений, закрывающих часть задачи. Разработчики могут выбрать комбинацию инструментов: модели экспортировать через FBX, материалы – через glTF 2.0, сцену – JSON/скриптом, а логику восстанавливать вручную на новом API.

## ПРИМЕРЫ МИГРАЦИИ МЕЖДУ ДВИЖКАМИ

Рассмотрим несколько реальных случаев (как успешных, так и не очень) когда разработчики переносили сцены или целые проекты между движками, и выводы из их опыта.

### Unity → Unreal Engine: портирование проекта

Многие инди-разработчики задумываются о переходе на *Unreal* ради улучшения графики или возможностей. Так, в 2019 году проект ***Hello Neighbor*** сменил движок с *Unity* на *Unreal Engine* 4 на позднем этапе разработки. Разработчики поняли, что для реализации продвинутого ИИ соседа и интерактивного окружения Unity не хватает производительности, и рискнули переключиться. Это совпадало

с тем, что издатель предоставил ресурсы для переноса. Опытные девелоперы отмечают, что смена движка на поздней стадии – шаг рискованный, "*almost career suicide"* [9], ведь нужно повторно реализовать массу возможностей функционала. В случае Hello Neighbor переход все же произошел: базовые механики переписали на C++/Blueprint, а 3D-модели дома и персонажей импортировали из *Unity*-проекта (*Unity* позволяет выгружать модели в FBX, которые *Unreal* импортирует без проблем). Однако многие скрипты пришлось создавать с нуля под *Unreal*. В результате игра вышла на UE4, что улучшило графику, но команда отметила, что время разработки увеличилось примерно на год из-за миграции. Данный случай иллюстрирует, что даже при переносе ассетов основная сложность заключается в миграции логики и поведений.

**Unity → Unreal Engine: использование конвертера**

В 2023 году, после объявленной смены политики лицензирования *Unity*, сразу несколько инди-команд попробовали перевести свои проекты на *Unreal* с помощью автоматических инструментов. Один из показательных примеров – проект, описанный в сообществе разработчиков, который перенес уровень из *Unity* в *Unreal Engine 5* за два дня с помощью FBX-Exporter и ручной настройки [11]. Команда экспортировала всю сцену *Unity* в FBX (сохранив геометрию и размещение объектов), затем импортировала в *Unreal*. Все меши и коллизии перенеслись, но материалы пришлось перенастраивать: разработчики выбрали в *Unreal* готовый PBR-шейдер и вручную назначили текстуры, опираясь на оригинал. Скрипты (враги, двери, головоломки) переписали на Blueprint за выходные, благо логика была относительно простая. В результате прототип уровня работал в новом движке, но качество было далеко от финального. Этот случай считается успешным быстрым портом, однако требует оговорки: переносилась только демосцена, а не весь проект. Полный проект потребовал бы значительно больше времени. Особо подчеркнем, что инструменты ускоряют перенос арт-содержимого, но не избавляют от переписывания геймплея.

**Unity → Unigine : автоматический перенос кейсов симуляций**

В сообществе *Unigine* отмечаются случаи, когда с помощью вышеупомянутого инструмента (JSON Exporter/Importer) переносили учебно-тренажерные сцены. Например, разработчики промышленного симулятора отмечали, что смогли перенести базовую сцену завода из *Unity* в *Unigine* за считанные часы. Иерархия объектов (цех, оборудование, краны) восстановилась автоматически, пришлось лишь перенастроить материалы под более продвинутый рендер *Unigine*. Скрипты же логики станков переписали на C++ с использованием *Unigine* API. Проект заработал в новом движке с заметно лучшим фотореализмом. Успешность этого примера во многом обусловлена схожестью подходов *Unity* и *Unigine*: оба поддерживают C#, объектно-ориентированную структуру, PBR-материалы, то есть выявить разницу было проще. Тем не менее даже здесь понадобились ручное исправление касательных пространств нормалей (иначе немного искажалось освещение на моделях) и настройки освещения под другую модель солнца/неба. Таким образом, при наличии специализированного инструмента перенос уровня может быть относительно быстрым, но финальная доводка занимает время.

**CryEngine → Lumberyard: пример Star Citizen**

Особым случаем является миграция между родственными движками. Игра Star Citizen начиналась на CryEngine, но в 2016 году студия Cloud Imperium объявила переход на Amazon Lumberyard (форк[5] *CryEngine*) на версии альфа 2.6. Благодаря тому, что Lumberyard основан на кодовой базе CryEngine, большая часть сцены и систем перешла *без серьезных переделок*: формат уровней *.pak, скрипты на Lua/C++, материалы – все было совместимо или требовало минимальной адаптации. Миграция заняла около года, шла постепенно, и игроки почти не

[5] Форк (от англ. fork – «вилка», «ответвление») – процесс создания новой версии программного проекта на основе исходного кода существующего проекта, в результате которого образуется независимая ветвь разработки. Форк позволяет разработчикам модифицировать оригинальный проект для собственных целей или развивать его в направлении, отличном от основного. Особенно распространен в среде открытого программного обеспечения, где часто используется для создания альтернативных версий программ или внесения существенных изменений в изначальную кодовую базу.

заметили разницы (альфа-версия просто обновилась). Этот успешный пример показывает, что при общем происхождении движков миграция сцен упрощается – унифицированные форматы и API позволяют перенести контент пакетно. Однако даже тут были нюансы: некоторые собственные модификации *CryEngine*, сделанные разработчиками, пришлось заново внедрять в Lumberyard. Тем более, такой случай – скорее исключение, ведь мало движков, столь совместимых друг с другом.

**Неудачные и прерванные попытки миграции**

Есть и примеры, когда затея миграции не доводилась до конца. Многие проекты на Kickstarter обещали *«переключиться на Unreal Engine для лучшего качества»*, но потом отменялись или выходили все же на старом движке, потому что команда не справлялась с переносом. Например, амбициозный инди-RPG ***midora*** планировал перейти с Game Maker на *Unity*, но этот шаг усложнил разработку, и проект заморозили (по отзывам, время ушло на переписывание кода вместо создания контента).

Другой пример – попытка портирования фанатами игры Morrowind на *Unity* (OpenMW): сообщество с открытым движком OpenMW фактически заново реализовало движок, способный читать ресурсы Morrowind. Хотя это технически не миграция одним нажатием кнопки, а полностью новое открытое внедрение, он демонстрирует сложность: потребовались годы, чтобы написать конвертеры форматов, интерпретатор скриптов и воспроизвести всю логику оригинальной игры. Этот проект все же увенчался успехом – OpenMW теперь запускает игровые сцены Morrowind с улучшениями, но трудозатраты сравнимы с созданием игры с нуля. Общий вывод из неудачных попыток: если нет достаточных ресурсов или инструментов, перенос сцен может занять слишком много времени, ставя под угрозу весь проект. Иногда рациональнее оставить игру на старом движке или выпустить как есть, чем пытаться мигрировать и не выпустить вовсе.

## СВЯЗАННЫЕ РАБОТЫ

Проблематика миграции контента между игровыми движками привлекает внимание как индустрии, так и исследователей, хотя специальных научных работ об автоматической конвертации сцен немного. Тем не менее ряд публикаций и исследований затрагивает смежные вопросы: сравнение движков, стандартизация форматов, перенос опыта разработки, которые важны для понимания текущего состояния проблемы.

### Сравнительный анализ движков и влияние на переносимость

Чтобы понять, как сложно мигрировать сцены, ученые сравнивают сами движки. Например, в [12] представлен эксперимент по созданию двух идентичных приложений – виртуальных 3D-выставок – одно в *Unity*, другое в *Unreal Engine*, с использованием одинаковых 3D-сканов моделей. Этот сравнительный анализ показал различия в эффективности и рабочих процессах: *Unity* оказался более эффективным в их тестах, однако важнее то, что авторам пришлось вручную воссоздавать сцену в каждом движке заново. Работа подтвердила, что даже при наличии одинаковых исходных ассетов поведение сцены (производительность, качество) отличается, и требуется адаптация под каждый движок. В [13] в контексте виртуальных хирургических тренажеров обсуждался выбор движка. Авторы отмечают, что для VR-проекта движок следует выбирать на старте, учитывая особенности каждого, поскольку переход на поздних этапах затруднителен. Косвенно это подтверждает главную мысль: из-за различий (графика, физика, поддержка VR-устройств) перенос между *Unity* и *Unreal* – нетривиальная задача, и решение о миграции должно приниматься взвешенно. Таким образом, сравнения движков подчёркивают фундаментальную неспособность к взаимодействию (*неинтероперабельность)* их экосистем, из-за которой и возникает проблема миграции.

### Стандартизация форматов и интероперабельность

В ответ на блокирующую проблему, когда проект привязан к одному движку, в индустрии и исследовательском сообществе предлагают стандарты обмена данными.

Khronos Group продвигает glTF 2.0 как *унифицированный формат 3D-сцен*, а Pixar – USD (Universal Scene Description). В статье-сообщении *Godot* [2] прямо говорится, что ранее не существовало хорошего формата обмена игровыми сценами и что glTF 2.0 – первый претендент на эту роль.

В работах по компьютерной графике также сравнивают форматы: отмечено, что Collada, задуманный как открытый стандарт, не оправдал ожиданий из-за размытых спецификаций и разных трактовок осей, масштабов и т. п.

Более новые разработки, такие как OpenGEX, glTF, стремятся зафиксировать единое описание сцены (сетки, материалы, анимация).

На международных конференциях по 3D-платформам, таких как SIGGRAPH и Game Developers Conference (GDC), постоянно обсуждается идея промежуточного представления игровой сцены (intermediate representation), независимого от движка, чтобы облегчить перенос. Например, формат USD изучается для интерактивных приложений: в форуме Alliance for OpenUSD экспериментируют с использованием USD Scene Graph в игровых движках, чтобы описывать поведение независимо [14]. Пока эти идеи находятся еще на ранней стадии разработки, но очевидна тенденция: стандартизация сцен рассматривается как решение проблемы миграции в долгосрочной перспективе.

**Методы автоматизации переноса кода и логики**

Хотя перенос кода – наиболее сложная часть, исследователи предпринимали попытки его упростить.

В частности, инженеры в сфере dev-tools [15] опубликовали proof-of-concept: использование ИИ (ChatGPT 3.5) для автоматической конвертации скриптов *Unity* C# в *Godot* GDScript. Результаты частично успешны на простых классах, но далеки от полной автоматизации.

В работе [16] по сохранению VR-арт объектов рассмотрен гипотетический сценарий миграции кода VR-приложения на другой движок для цели сохранения произведений искусства виртуальной реальности. Отмечено, что прямой перенос движкового кода крайне труден, и предложено сохранять не сам код, а описание сценариев и медиаактивов, чтобы в будущем воспроизвести опыт на другом

движке. Другими словами, даже хранение VR-проекта на десятилетия вперед может потребовать его портирования на новый движок, и это признано серьезным вызовом.

В ряде работ по обучающим VR-системам (см., например, [13]) также фигурирует идея абстрагирования логики от движка: например, создание модульных тренажеров, где движок можно заменить без переписывания методики обучения. Однако реализовать это сложно – по сути, нужны метадвижок или слой совместимости.

Отметим также подходы формального описания логики игры, позволяющего абстрагироваться от платформы реализации [17]. Кроме того, в работе [18] представлен подход для портирования VR-приложений между различными гарнитурами, что подчеркивает актуальность автоматизации миграции игровых сцен с учетом различий API.

**Выводы и текущее состояние проблемы**

На сегодняшний день научно-техническое сообщество сходится во мнении, что полностью автоматическая миграция игровой сцены между разными движками все еще не решена: слишком много аспектов завязано на конкретную реализацию движка. Тем не менее есть значительный прогресс в части обмена контентом: современные движки поддерживают стандартные 3D-форматы, развивается экосистема конвертеров и плагинов. Открытые форматы (glTF, OpenGEX, USD) постепенно снижают «барьер входа» при переносе моделей и анимаций. Опыт практических портирований и исследования показали, что автоматический перенос художественных данных (геометрия, текстуры, анимация) возможен примерно на 70–80%, тогда как перенос геймплейной логики и поведения – максимум на 10–20%, остальное – вручную. Разработчики все чаще учитывают эту проблему при планировании: либо выбирают движок с учетом долгосрочных целей (чтобы не мигрировать), либо закладывают ресурсы для возможного переноса (например, пишут логику в абстрактном виде, контролируют зависимость от специфических функций). Исследования продолжаются, в том числе в академической среде анализируют подходы к унификации, например, VR-тренажеров. В целом текущее состояние можно охарактеризовать как стадию частичного решения:

есть инструменты для переноса ассетов и сценовой геометрии, но универсального решения для полной миграции игр пока нет. Специалисты рекомендуют использовать комбинацию существующих методов и быть готовыми к существенной доработке при переносе, что подтверждается и практикой (кейсы миграции), и публикациями. Эта проблема известна, частично решаема, но окончательно не устранена на данный момент.

## ПЕРЕНОС ИГРОВЫХ СЦЕН ИЗ UNREAL ENGINE В UNIGINE

Одной из конкретных задач, рассмотренных нами, является перенос игровых сцен из *Unreal Engine* в *Unigine*. Ранее такого решения представлено не было. Исходный код конвертера и примеры использования доступны в открытом репозитории [19]. Этот процесс демонстрирует типичные технические вызовы, с которыми сталкиваются разработчики при миграции контента между различными игровыми движками.

### Сбор и экспорт данных из Unreal Engine

В исходном проекте сцена в *Unreal Engine* анализируется с использованием встроенного Python API. Программа, написанная на Python, извлекает ключевую информацию об объектах уровня: названия, классы, трансформации и пути к сеткам, после чего данные сохраняются в формате JSON (см. листинг 1).

Листинг 1. Пример экспорта данных в JSON

```python
# Пример скрипта на Python для экспорта данных из Unreal Engine в JSON
import unreal
import json

def export_level_data(output_file):
    level_actors = unreal.EditorLevelLibrary.get_all_level_actors()
    scene_data = []
    for actor in level_actors:
        actor_data = {
            "name": actor.get_name(),
            "class": actor.get_class().get_name(),
            "transform": {
                "location": actor.get_actor_location().to_tuple(),
                "rotation": actor.get_actor_rotation().to_tuple(),
                "scale": actor.get_actor_scale3d().to_tuple()
            }
        }
        scene_data.append(actor_data)
    with open(output_file, 'w', encoding='utf-8') as f:
        json.dump(scene_data, f, indent=4)

# Пример вызова функции
export_level_data("C:/Export/scene_data.json")
```

Такой формат позволяет структурировать информацию об иерархии объектов, материалах, коллайдерах и прочих атрибутах, необходимых для последующего восстановления сцены в целевом движке.

**Преобразование ассетов**

Одной из существенных проблем является несовместимость форматов ассетов: *Unreal Engine* использует формат .uasset, который применим исключительно внутри его среды. Для обеспечения совместимости с *Unigine* необходимо конвертировать эти файлы. В предлагаемом решении реализован метод, который преобразует .uasset в .fbx (см. листинг 2), что позволяет сохранить геометрию и анимационные данные, а также обеспечивает возможность повторного импорта в *Unigine*.

Дополнительно производится пересчет касательного пространства (т. е. параметров нормалей и касательных в соответствии с требованиями целевого движка).

Листинг 2. Пример конвертации ассета

```
# Пример функции для конвертации файла .uasset в .fbx
import unreal

def convert_uasset_to_fbx(uasset_path, output_fbx_path):
    # Предполагается, что в Unreal Engine имеется вызов метода конвертации,
    # доступного через Python API (данный код иллюстративный)
    converter = unreal.AssetToolsHelpers.get_asset_tools()
    success = converter.export_asset(uasset_path, output_fbx_path)
    if success:
        unreal.log("Конвертация завершена: " + output_fbx_path)
    else:
        unreal.log_error("Ошибка конвертации: " + uasset_path)

# Пример вызова функции
convert_uasset_to_fbx("C:/Project/Content/MyAsset.uasset", "C:/Export/MyAsset.fbx")
```

**Импорт данных в Unigine**

После экспорта JSON-файла и конвертации ассетов следующим шагом является импорт полученных данных в *Unigine*. Для этого разработан графический интерфейс, позволяющий указать пути к проектам *Unreal Engine* и *Unigine*, а также автоматизировать перенос ассетов. Программа анализирует JSON-файл, извлекает информацию об объектах, а затем создаёт соответствующие узлы в файле сцены (.world) *Unigine*, корректируя трансформации с учётом различий в системах координат между движками. Особое внимание уделяется корректному восстановлению иерархии сцены, а также обеспечению согласованности параметров освещения и материалов.

При переносе возникает весь спектр ранее описанных трудностей:

- *Несовместимость форматов* – необходимость конвертации .uasset в .fbx и последующая адаптация ассетов под систему *Unigine*.

- *Различия в системах координат* – автоматическая корректировка положений и поворотов объектов для соблюдения новых правил преобразования (ср. рисунки 1 и 2).
- *Пересчёт касательного пространства* – для корректного отображения шейдеров и нормалей требуется адаптация касательного пространства.
- *Сохранение сцены* – восстановление полной иерархии и атрибутов объектов из JSON-файла для точного воспроизведения сцены в новом движке.

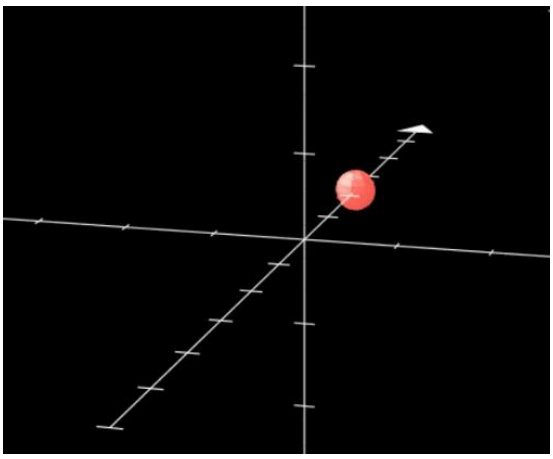


Рис. 1. Система координат в *Unreal Engine*

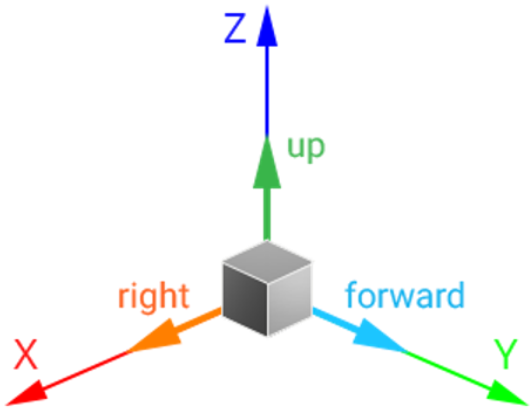



Рис. 2. Система координат в *Unigine*

Таким образом, перенос игровых сцен из *Unreal Engine* в *Unigine* является наглядным примером сложностей миграции контента между игровыми движками (ср. рисунки 3 и 4).

Рис. 3. Тестовая игровая сцена в *Unreal Engine*

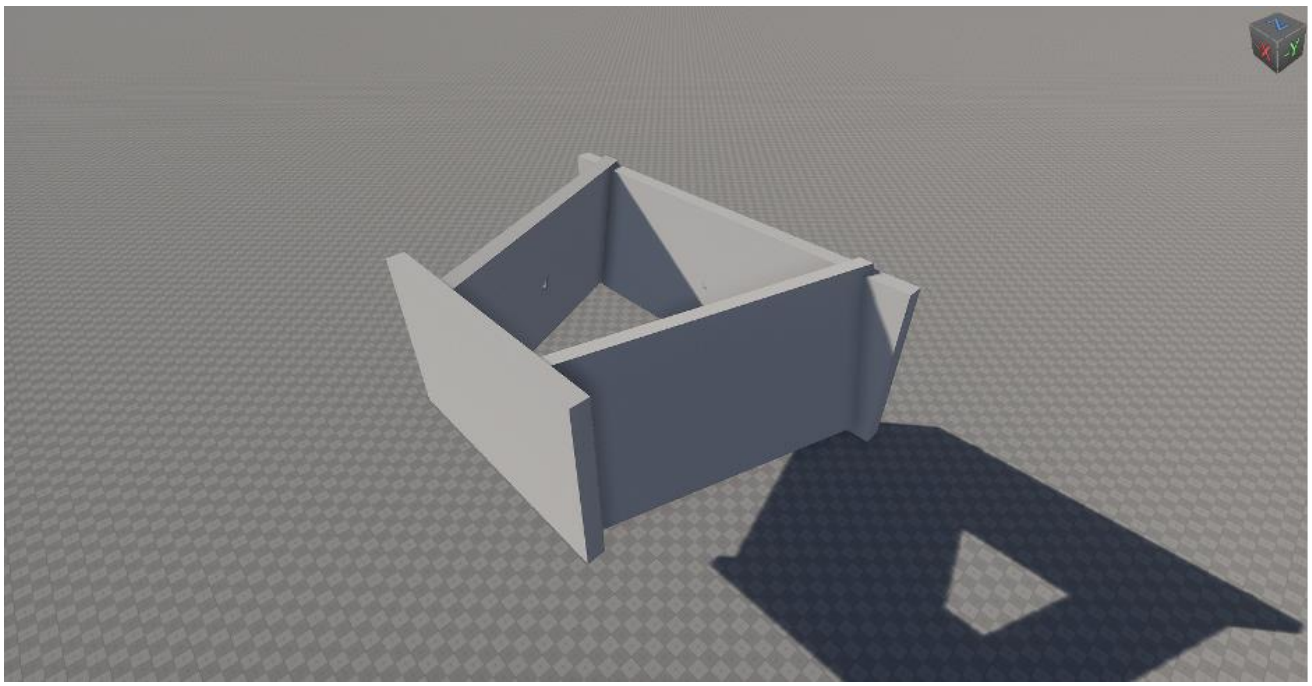


Рис. 4. Перенесенная в *Unigine* тестовая игровая сцена

Реализованные решения демонстрируют, что использование промежуточных форматов (JSON, FBX) и специализированных конвертеров может существенно облегчить задачу, однако окончательная адаптация часто требует значительных усилий и ручной доработки.

### Будущие направления автоматизации переноса игровых сцен

Несмотря на значительный прогресс в разработке инструментов для экспорта и импорта игровых сцен, полный перенос контента «под ключ» остаётся масштабной задачей, требующей дальнейших исследований и совершенствования методов.

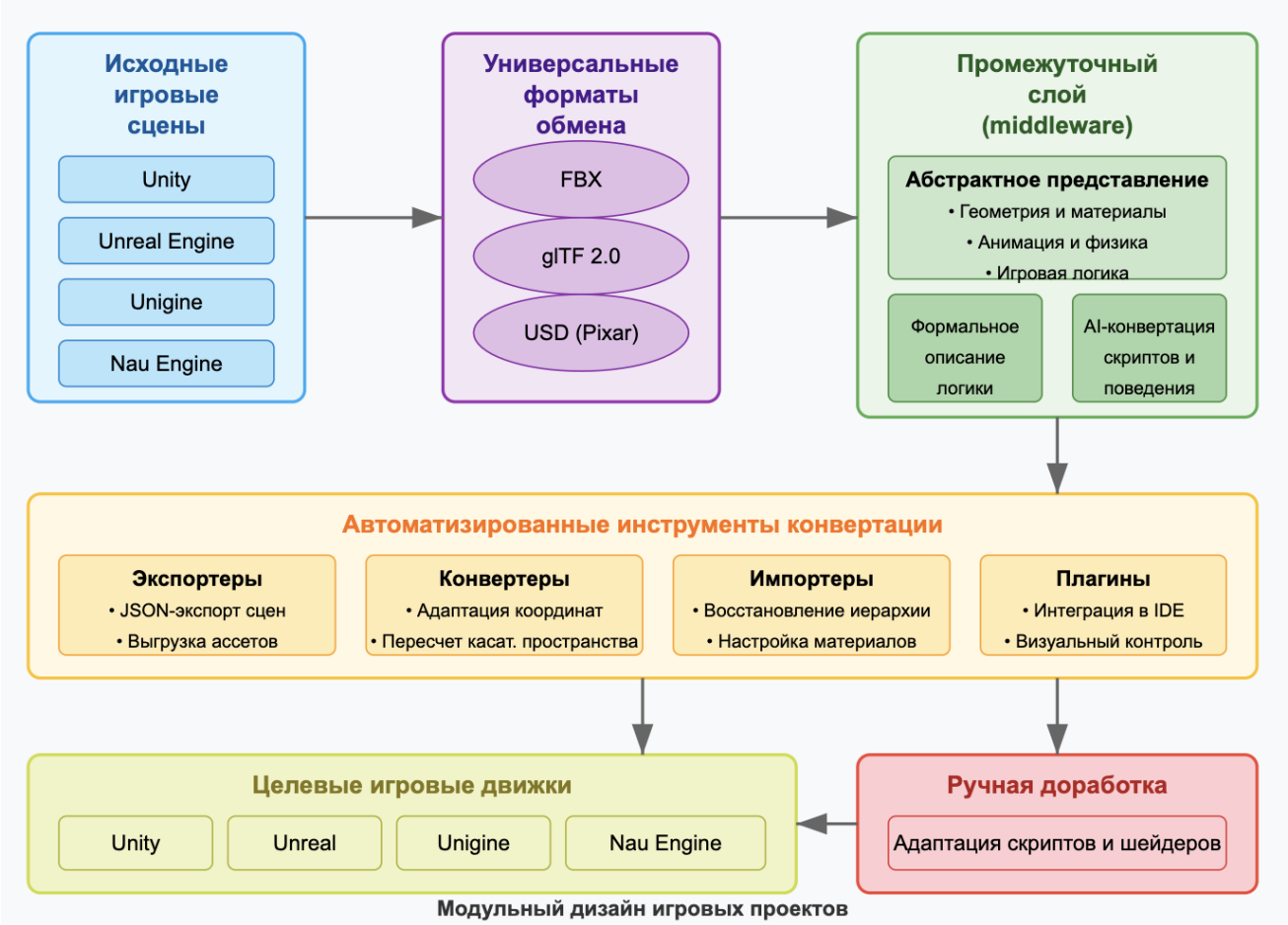



Рис. 5. Ключевые этапы и направления дальнейших исследований

В этом контексте можно выделить следующие перспективные направления (рис. 5).

1. **Унификация форматов обмена данными.**
   Разработка и внедрение универсальных форматов, таких как USD или расширенные версии glTF, способных полноценно описывать геометрию, материалы, анимацию и даже логику игровых сцен, позволяет снизить зависимость от проприетарных решений. Стандартизация обмена данными

между движками является ключевым шагом к автоматизации переноса.

2. **Разработка промежуточного слоя (middleware).**
   Создание абстрактного представления игровых сцен, которое отделяет содержание (ассеты, анимации, логику) от платформенной реализации, может значительно упростить процесс миграции. Такой слой позволит адаптировать одни и те же данные под требования разных движков посредством набора адаптеров.
3. **Автоматизация переноса игровой логики.**
   На данный момент статические данные (геометрия, материалы, анимация) переносятся относительно успешно, тогда как перенос кода и логики требует значительной ручной работы. Перспективным направлением является использование методов искусственного интеллекта для автоматической конвертации скриптов и описания поведения объектов, что позволит автоматизировать и эту часть процесса. Возможно, это необходимо сочетать с промежуточным этапом формального описания игровой логики.
4. **Модульный дизайн игровых проектов.**
   Планирование архитектуры игр с учетом возможности будущей миграции может существенно упростить перенос. Разделение проекта на независимые модули, где логика, визуальные ассеты и физические параметры обрабатываются отдельно, позволит переносить лишь отдельные компоненты, минимизируя усилия при адаптации под новый движок.
5. **Интеграция в рабочие конвейеры разработки.**
   Разработка плагинов и утилит, которые легко интегрируются в существующие системы разработки, такие как *Unity*, *Unreal Engine*, *Unigine* и *Nau Engine*, позволит автоматизировать не только этап переноса, но и его последующую адаптацию что, в свою очередь,позволит разработчикам быстрее переключаться между платформами при изменении требований проекта.

В рамках дальнейших исследований планируется разработать универсальное решение для переноса игровых сцен, которое будет поддерживать миграцию между *Unity*, *Unreal Engine*, *Unigine* и *Nau Engine*. При этом особое внимание будет уделено сохранению заложенной игровой логики, что является особенно важным

для проектов с богатым интерактивным функционалом. Достижение этой цели потребует глубокого анализа специфики каждого движка, создания гибкой архитектуры промежуточного представления данных и разработки адаптивных конвертеров для автоматического преобразования контента.

## ЗАКЛЮЧЕНИЕ

В статье дан всесторонний анализ технических аспектов переноса игровых сцен между различными игровыми движками. Рассмотрены основные проблемы, связанные с несовместимостью форматов данных, различиями в API для рендеринга, физики и логики, а также сложностями конвертации материалов, шейдеров и анимационных данных. Изучение существующих инструментов (таких как Unity→Unreal, Unity→Unigine, Unity→Godot) показало, что автоматизация статических аспектов сцены уже достигла заметного прогресса. Однако динамическая логика и интерактивные элементы остаются областью, требующей значительной ручной доработки.

Особое внимание уделено фундаментальным подходам к решению проблемы: использование универсальных форматов обмена (FBX, glTF, USD), разработка промежуточных слоев для абстрагирования данных от специфики конкретного движка и применение модульного дизайна игровых сцен. Эти направления открывают возможности для создания комплексного решения «под ключ», способного обеспечить автоматизированный перенос не только графических, но и логических компонентов игровых проектов.

В качестве примера практической реализации рассмотрен перенос игровых сцен из *Unreal Engine* в *Unigine*, для которого разработан собственный pipeline с использованием Python-скриптов. В дальнейших исследованиях планируется расширение функционала переносчика для поддержки миграции между *Unity*, *Unreal Engine*, *Unigine* и *Nau Engine* с сохранением заложенной игровой логики. Эти усилия позволят существенно снизить трудозатраты при адаптации проектов под новые платформы и обеспечить их высокое качество.

## СПИСОК ЛИТЕРАТУРЫ

1. Export From Unity to Unigine in Seconds // Game From Scratch. 2024.

URL: https://gamefromscratch.com/export-from-unity-to-unigine-in-seconds/

2. *Linietsky J.* Why we should all support glTF 2.0 as THE standard asset exchange format for game engines // GodotEngine. 2017.
URL: https://godotengine.org/article/we-should-all-use-gltf-20-export-3d-assets-game-engines/

3. @Coldwalker37. Creating a Unity to Godot converter, and a Unity to Unreal converter // Reddit. GameDev. 2023. URL: https://www.reddit.com/r/gamedev/comments/16j7xc8/creating_a_unity_to_godot_converter_and_a_unity/

4. @NegInfinity. Project Exodus – Unity to unreal scene converter //Unreal Engine. Forum. Plugins. 2019. URL: https://forums.unrealengine.com/t/plugin-project-exodus-unity-to-unreal-scene-converter/125362

5. Quevillon A. Utu Plugin – Unity to Unreal Project Converter. 2023. URL: https://alexquevillon.gumroad.com/l/UtuPlugin

6. Migrating to UNIGINE from Unity: Content Creation // Unigine. URL: https://developer.unigine.com/en/docs/future/migration/from_unity/content

7. *Zylann M.* Unity Engine to Godot Engine exporter // GitHub. 2019. URL: https://github.com/Zylann/unity_to_godot_converter

8. OpenGEX. 2022. URL: https://opengex.org/

9. *Torres J.C.* Star citizen space sim moves to amazon lumberyard game engine // Lash Gear. Gaming. 2016. URL: https://www.slashgear.com/star-citizen-space-sim-moves-to-amazon-lumberyard-game-engine-26468764/

10. Unity to Unreal Engine Game Transfer // Pungle Studio.
URL: https://pinglestudio.com/service/unity-to-unreal-engine-transfer

11. How to transfer your level from Unity3D to Unreal Engine 5 (and vice versa) // Youtube. iBrews. 2023.
URL: https://www.youtube.com/watch?v=H9cN0m-p8zM

12. *Ciekanowska A., Kiszczak-Gliński A., Dziedzic K.* Comparative analysis of Unity and Unreal Engine efficiency in creating virtual exhibitions of 3D scanned models // Journal of Computer Sciences Institute. 2021. Vol. 20. P. 247–253.

13. *Шараева Р.А. и др.* Подходы к проектированию виртуальных тренажёров хирургических операций // Электронные библиотеки. 2022. Т. 25. №. 5. С. 489–532.

14. *Selva P.E.* GameEngine as a SceneIndex plugin // Alliance for OpenUSD. 2024. URL: https://forum.aousd.org/t/gameengine-as-a-sceneindex-plugin/1238

15. *Shikin B.* Migrating from Unity to Other Game Engines // AppLovin. 2023. URL: https://www.applovin.com/blog/migrating-from-unity-to-other-game-engines/

16. *McConchie J., Ensom T.* Preserving virtual reality artworks: a museum perspective // ACM SIGGRAPH 2019 Talks. 2019. P. 1–2.

17. *Kugurakova V.V.* A formal approach to spatio-temporal modeling of game systems // Uchenye Zapiski Kazanskogo Universiteta. Seriya Fiziko-Matematicheskie Nauki. 2024. V. 166. № 4. P. 532–554.
https://doi.org/10.26907/2541-7746.2024.4.532-554.

18. *Kugurakova V., Vasilov T., Khafizov M.* Approaches to automating VR applications porting using common techniques // BIO Web of Conferences. 2024. Vol. 84. Art. No. 02016.

19. *Бондарь А.* UE to Unigine Exporter // GitHub. 2025.
URL: https://github.com/Bonndii/UEtoUnigineExporter

# PROBLEMS, SOLUTIONS AND PERSPECTIVES OF AUTOMATED TRANSFER OF GAME SCENES BETWEEN GAME ENGINES

**A.O. Bondar[1] [0009-0005-5296-9686], V.V. Kugurakova[2] [0000-0002-1552-4910]**

[1, 2]*Institute of Information Technologies and Intelligent Systems, Kazan Federal University*

[1]alexey.bondar.2013@mail.ru, [2]vlada.kugurakova@gmail.com

***Abstract***

This article examines the technical challenges involved in transferring game scenes between various game engines. It analyzes the key issues arising from differences in scene formats, incompatibilities in rendering and physics APIs, as well as problems in converting materials, shaders, and animation data, and discrepancies in coordinate systems. Existing tools and methods, including automated solutions for exporting, converting, and importing data, are presented with a particular focus on migrating content from Unreal Engine to Unigine. Furthermore, the paper discusses fundamental approaches to solving the problem, such as the use of universal exchange formats (FBX, glTF, USD), the development of middleware, and the modular design of game scenes, which pave the way for future automation. The work also highlights our group's research results on the formal description of game logic and approaches to porting VR applications across different libraries. The conclusions provide practical recommendations for developers and outline future research directions in the area of automated content transfer between game engines.

***Keywords:*** *game scene migration, game engine, content migration, Unreal Engine, Unity, Unigine, Nau Engine, Godot, CryEngine, format conversion, data standardization.*

**REFERENCES**

1. Export from Unity to Unigine in Seconds // Game From Scratch. 2024. URL: https://gamefromscratch.com/export-from-unity-to-unigine-in-seconds/

2. *Linietsky J.* Why we should all support glTF 2.0 as THE standard asset exchange format for game engines // GodotEngine. 2017. URL: https://godotengine.org/article/we-should-all-use-gltf-20-export-3d-assets-game-engines/

3. @Coldwalker37. Creating a Unity to Godot converter, and a Unity to Unreal converter // Reddit. GameDev. 2023.
URL: https://www.reddit.com/r/gamedev/comments/16j7xc8/creating_a_unity_to_godot_converter_and_a_unity/

4. @NegInfinity. Project Exodus – Unity to unreal scene converter // Unreal Engine. Forum. Plugins. 2019. URL: https://forums.unrealengine.com/t/plugin-project-exodus-unity-to-unreal-scene-converter/125362

5. *Quevillon A.* Utu Plugin – Unity to Unreal Project Converter. 2023.
URL: https://alexquevillon.gumroad.com/l/UtuPlugin

6. Migrating to UNIGINE from Unity: Content Creation // Unigine.
URL: https://developer.unigine.com/en/docs/future/migration/from_unity/content

7. *Zylann M.* Unity Engine to Godot Engine exporter // GitHub. 2019.
URL: https://github.com/Zylann/unity_to_godot_converter

8. OpenGEX. 2022. URL: https://opengex.org/

9. *Torres J.C.* Star citizen space sim moves to amazon lumberyard game engine // Lash Gear. Gaming. 2016. URL: https://www.slashgear.com/star-citizen-space-sim-moves-to-amazon-lumberyard-game-engine-26468764/

10. Unity to Unreal Engine Game Transfer // Pungle Studio.
URL: https://pinglestudio.com/service/unity-to-unreal-engine-transfer

11. How to transfer your level from Unity3D to Unreal Engine 5 (and vice versa) // Youtube. iBrews. 2023.
URL: https://www.youtube.com/watch?v=H9cN0m-p8zM

12. *Ciekanowska A., Kiszczak-Gliński A., Dziedzic K.* Comparative analysis of Unity and Unreal Engine efficiency in creating virtual exhibitions of 3D scanned models // Journal of Computer Sciences Institute. 2021. Vol. 20. P. 247–253.

13. *Sharaeva A. et al. Approaches to the development of virtual surgical training* // Russian Digital Libraries. 2022. V. 25. No. 5. P. 489–532.
https://doi.org/10.26907/1562-5419-2022-25-5-489-532 (In Russian).

14. *Selva P.E.* Game Engine as a Scene Index plugin // Alliance for OpenUSD. 2024. URL: https://forum.aousd.org/t/gameengine-as-a-sceneindex-plugin/1238

15. *Shikin B.* Migrating from Unity to Other Game Engines // AppLovin. 2023. URL: https://www.applovin.com/blog/migrating-from-unity-to-other-game-engines/

16. *McConchie J., Ensom T.* Preserving virtual reality artworks: a museum perspective // ACM SIGGRAPH 2019 Talks. 2019. P. 1–2.

17. *Kugurakova V.V.* A formal approach to spatio-temporal modeling of game systems // Uchenye Zapiski Kazanskogo Universiteta. Seriya Fiziko-Matematicheskie Nauki. 2024. Vol. 166. No. 4. P. 532–554. https://doi.org/10.26907/2541-7746.2024.4.532-554.

18. *Kugurakova V., Vasilov T., Khafizov M.* Approaches to automating VR applications porting using common techniques // BIO Web of Conferences. 2024. Vol. 84. Art. No. 02016.

19. *Bondar A.* UE to Unigine Exporter // GitHub. 2025. URL: https://github.com/Bonndii/UEtoUnigineExporter

## СВЕДЕНИЯ ОБ АВТОРАХ

**БОНДАРЬ Алексей Олегович** – магистрант Институт информационных технологий и интеллектуальных систем Казанского федерального университета. Направление исследований: миграция игровых сцен между платформами и игровыми движками.

**Alexey Olegovich BONDAR** – Master's student at the Institute of Information Technologies and Intelligent Systems of Kazan Federal University. Research area: migration of game scenes between platforms and game engines.

email: alexey.bondar.2013@mail.ru

ORCID: 0009-0005-5296-9686

**КУГУРАКОВА Влада Владимировна** – доцент, кандидат технических наук, зав. кафедрой индустрии разработки видеоигр Института ИТИС КФУ. Зона научных интересов: формальная верификация видеоигр, иммерсивность виртуальных миров.

**Vlada Vladimirovna KUGURAKOVA** – Ph. D. of Engineering Sciences, Head of the Video Game Development Industry Department of ITIS KFU.

email: vlada.kugurakova@gmail.com

ORCID: 0000-0002-1552-4910

*Материал поступил в редакцию 30 января 2025 года*

УДК 004.928+004.85+004.896

# СИНТЕТИЧЕСКИЙ ДАТАСЕТ MetaHuman ДЛЯ ОПТИМИЗАЦИИ СКИННИНГА 3D-МОДЕЛЕЙ

**Р. Р. Газизов[1] [0000-0002-8349-264X], М. Д. Белов[2] [0009-0008-7680-1839]**

[1, 2]*Казанский федеральный университет, Институт информационных технологий и интеллектуальных систем*

[1]gazizov782@gmail.com, [2]mak.bel.2002@mail.ru

***Аннотация***

Представлена методика создания синтетического набора данных с использованием системы MetaHuman для оптимизации скиннинга 3D-моделей. Основное внимание уделено улучшению качества привязки (скиннинга) геометрии к скелетам персонажей за счет разнообразия генерируемых высокореалистичных моделей. С помощью MetaHuman сформирован обширный датасет, включающий десятки виртуальных персонажей с различными антропометрическими характеристиками и точно заданными весовыми параметрами скиннинга. На основе этих данных обучен алгоритм, оптимизирующий распределение весов между костями и поверхностью модели.

Предложенный подход автоматизирует процесс настройки весов, что позволяет значительно сократить ручной труд риггеров и повысить точность деформаций при анимации. Эксперименты показали, что использование синтетических данных приводит к сокращению ошибок скиннинга и более плавным движениям модели по сравнению с традиционными методами. Результаты работы имеют непосредственное применение в индустрии видеоигр, анимации, виртуальной реальности и симуляций, где требуется быстрый и качественный риггинг множества персонажей. Предложенный метод может быть интегрирован в существующие графические движки и конвейеры разработки в виде плагина или инструмента, облегчая внедрение технологии в практические проекты.

***Ключевые слова****: синтетический датасет, MetaHuman, нейронные сети, скиннинг 3D-моделей, компьютерная анимация, машинное обучение.*

## ВВЕДЕНИЕ

Современные технологии 3D-моделирования и компьютерной графики активно используются в самых различных областях – от видеоигр и кино до медицины и виртуальных прототипов. Важной задачей в данной сфере является создание реалистичных анимаций персонажей, что требует точного и качественного процесса скиннинга.

Скиннинг (skinning, от англ. skin – кожа) – это процесс привязки 3D-модели к скелету (риггинг, от англ. rig – оснастка, упряжка) с назначением весовых коэффициентов вершин для костей, что позволяет анимировать модель с реалистичными деформациями (рис. 1). Традиционно скиннинг – трудозатратная ручная работа, требующая от художника аккуратного «раскрашивания» весов для каждой кости. Назначение вершинных весов – крайне трудоёмкая и сложная задача [1], поэтому актуально применение автоматизированных (процедурных) методов для её облегчения.

В последние годы появилось несколько подходов к автоматизации скиннинга: от специализированных инструментов (например, **MetaHuman**[1] от Epic Games для реалистичных людей) до алгоритмических методов (авториггинг по геометрическим правилам) и методов на основе машинного обучения (обученных нейросетей, предсказывающих риги и веса).

В статье представлен метод создания синтетического датасета с использованием технологии MetaHuman для автоматического скиннинга 3D-моделей в сравнении с альтернативными решениями, включая классические алгоритмы и нейросетевые методы.

[1] https://www.unrealengine.com/en-US/metahuman

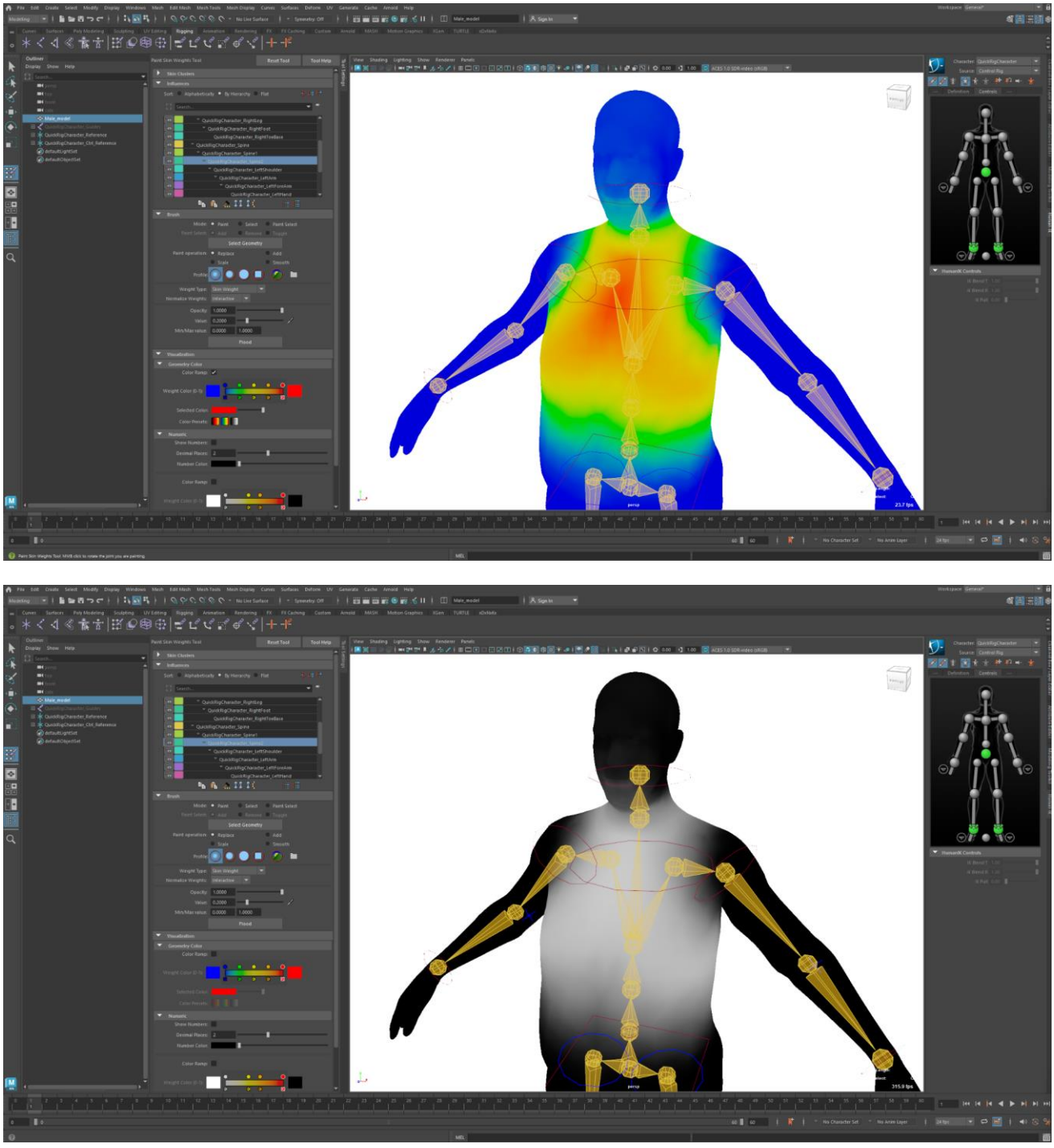


Рис. 1. Скиннинг модели

## MetaHuman ДЛЯ АВТОМАТИЗИРОВАННОГО СКИННИНГА

MetaHuman – это фреймворк от Epic Games для создания высокореалистичных цифровых людей, предоставляющий готовые фотореалистичные модели с настроенным ригом. MetaHuman Creator (облачный сервис) позволяет за минуты получить анимируемый персонаж с качественной геометрией, кожей, волосами и одеждой [2]. Ключевая особенность состоит в том, что все созданные персонажи используют стандартный высокодетализированный скелет MetaHuman, включающий специальные дополнительные кости (helper joints) и коррективные морфы (morph targets) для улучшения деформаций. По комментариям разработчиков, тело MetaHuman использует *линейный скиннинг с дополнительными вспомогательными суставами* и узлами управления позой (PoseDriver-nodes) для правдоподобной деформации мышц [3]. Лицо в MetaHuman в основном анимируется суставами (косточками лица) с добавлением морфинга с плавными переходами (blendshape-morphs) на высоком уровне детализации для тонкой мимики (рис. 2).

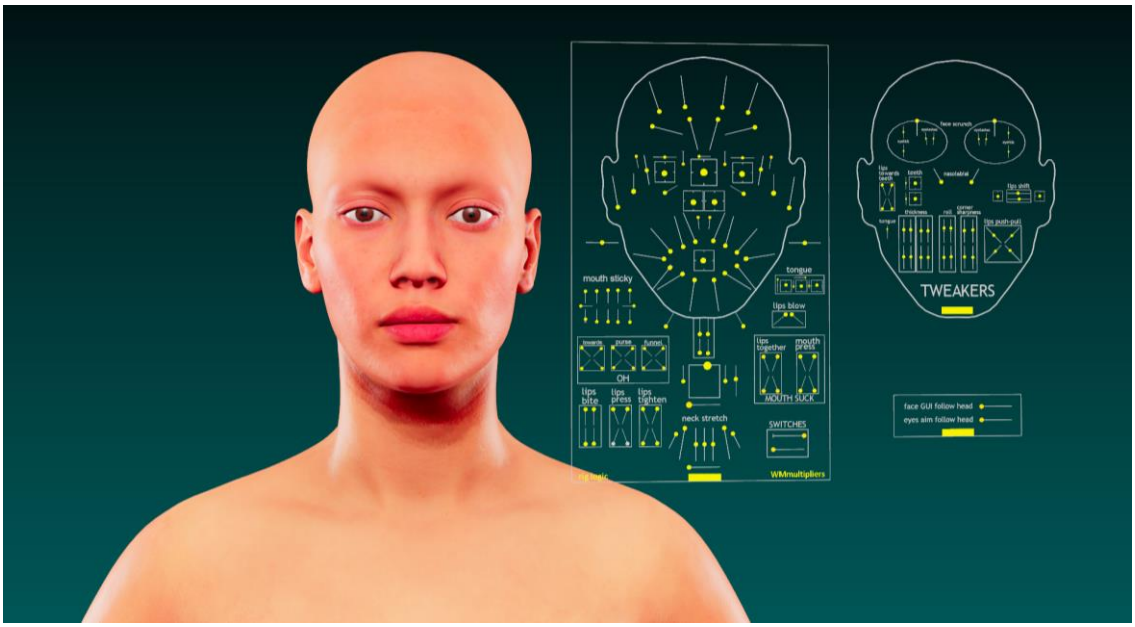



Рис. 2. Интерфейс Metahuman
(практически идентичный в UE5 и Autodesk Maya)

Недавно компания Epic Games представила инструмент **Mesh to MetaHuman**, расширяющий возможности автоматизации. Он позволяет взять произвольную пользовательскую 3D-модель головы/лица и конвертировать её в MetaHuman, автоматически подгоняя под топологию MetaHuman и связывая с его скелетом. Как отмечено в официальном блоге, этот инструмент выполняет автоматическое трекинговое совмещение черт лица и подгонку шаблонной сетки MetaHuman к загруженной модели, а затем генерирует персонажа, полностью оснащённого ригом и готового к анимации.

Пользователь может доработать результат в MetaHuman Creator – например, поправить черты, повторно наложить текстуры кожи или добавить прическу. Важным преимуществом является скорость: получение полностью готового к анимации цифрового человека из статичной модели занимает считанные минуты. Это подтверждает практическую целесообразность MetaHuman как инструмента автоматизированного скиннинга, по крайней мере для человекоподобных моделей, так как MetaHuman ориентирован только на реалистичных гуманоидов. Скелет и деформационные настройки заточены под человеческую анатомию; сильно стилизованные персонажи или существа с нестандартной морфологией могут не вписаться в шаблон. Нужно подчеркнуть, что MetaHuman – проприетарная технология, тесно интегрированная с Unreal Engine (UE). Для использования результатов (например, рига) вне экосистемы UE может потребоваться конвертация. Тем не менее в своей нише (авториггинг реалистичных людей) MetaHuman задаёт высокую планку качества. Экспериментально продемонстрировано, что модели MetaHuman могут быть сразу анимированы и дают правдоподобные движения. Например, нельзя переоценить использование таких инструментов, как MetaHuman Creator, для генерации 3D-датасетов [4] для быстрой генерации реалистичных 3D-персонажей, что позволяет создавать массивы синтетических персонажей для анимации и обучения нейросетей практически без ручного риггинга.

В контексте производства игровых персонажей MetaHuman может значительно ускорить этап создания персонажей [5], процесс разработки которых является предварительным и базовым по отношению к созданию его одежды и других атрибутов. Наличие готового рига (например, полученного через MetaHuman) уже на ранней стадии упрощает последующий дизайн одежды и анимацию, что

подтверждается практикой индустрии. Итак, если стоит задача быстро получить реалистичную 3D-модель человека с качественным скиннингом, использование MetaHuman крайне целесообразно (однако для негуманоидных моделей или вне Unreal Engine могут потребоваться альтернативы).

## АЛГОРИТМИЧЕСКИЕ МЕТОДЫ АВТОМАТИЧЕСКОГО СКИННИНГА

До появления нейросетевых подходов автоматизация скиннинга опиралась на алгоритмические решения, основанные на геометрическом анализе модели. Классический пример это алгоритм **Pinocchio** [6], часто упоминаемый как пионер автоскиннинга.

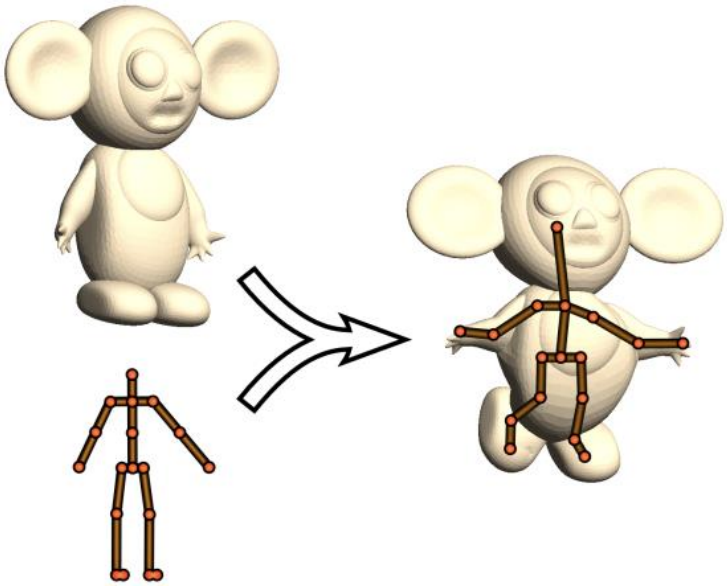


Рис. 3. В треугольную сетку чебурашки встраивается скелет, позволяющий наложить движения на изначально статичную фигуру [6]

Этот метод берет на вход произвольную 3D-сетку персонажа (см. рис. 3) и заданный шаблон скелета (например, гуманоидный скелет с определенным количеством костей), после чего: (1) автоматически встраивает скелет внутрь модели, находя оптимальные позиции суставов, и (2) рассчитывает веса привязки вершин к костям на основе расстояний, сглаженных решением уравнения диффузии тепла по поверхности модели. Такая техника тепловой диффузии (heat diffusion) распределяет влияние костей плавно и оказалась настолько удачной, что её внедрили во многие инструменты 3D-графики. Pinocchio продемонстрировал впечатляющие результаты: на типовом ПК того времени алгоритм автоматически

оснащал персонаж менее чем за минуту, а качество получаемой анимации было сопоставимо с уровнем современных видеоигр. Система была протестирована на 16 различных моделях – для 13 из них риг был построен корректно без вмешательства, а для оставшихся трех потребовалась лишь минимальная правка (добавление одной подсказки по положению сустава). Таким образом, уже более 15 лет назад было доказано, что авториггинг на основе шаблонов и диффузионных весов способен работать достаточно универсально и эффективно.

Другой подход это автоматическое *выделение* скелета из геометрии (skeleton extraction). Ряд алгоритмов по форме модели пытается вычислить её «скелет» (например, анализируя медиальную ось объекта или граф разбиения поверхности). Однако на практике такие методы менее надежны для анимации: полученный скелет может не совпадать с желаемой структурой (особенно для гуманоидов, где лучше задать заведомо анатомически правильный скелет). Поэтому современные алгоритмические решения чаще используют подгонку (fitting) шаблона: задается основа – типовой скелет человека, четвероногого животного и т. п., которая подгоняется под модель. Помимо Pinocchio, такой принцип реализован в ряде инструментов: например, облачный сервис **Adobe Mixamo**[2] позволяет пользователю загрузить статичную 3D-модель персонажа (обычно гуманоидного), после чего автоматически рассчитывает для нее скелет (пользователь может указать позиции ключевых точек – кисти, локти, колени – для улучшения результата) и весовые коэффициенты. Mixamo использует библиотеку готовых анимаций, созданных методом захвата движений, что гарантирует реалистичность получаемых движений [1]. Mixamo использует подход, похожий на метод *Pinocchio* [6], который применяется для встраивания скелета: оптимизируется положение суставов внутри модели путем минимизации специальной функции стоимости. Перед этим пользователь отмечает несколько ключевых точек (подбородок, запястья, локти, колени, пах) на Т-модели[3] (см. рис. 4) загруженного персонажа [7].

[2] https://www.mixamo.com/

[3] Т-модель — для удобства текстурирования и риггинга в 3D-моделировании используется базовая поза персонажа, при которой руки разведены в стороны горизонтально, а ноги слегка расставлены, образуя силуэт, напоминающий букву «Т».

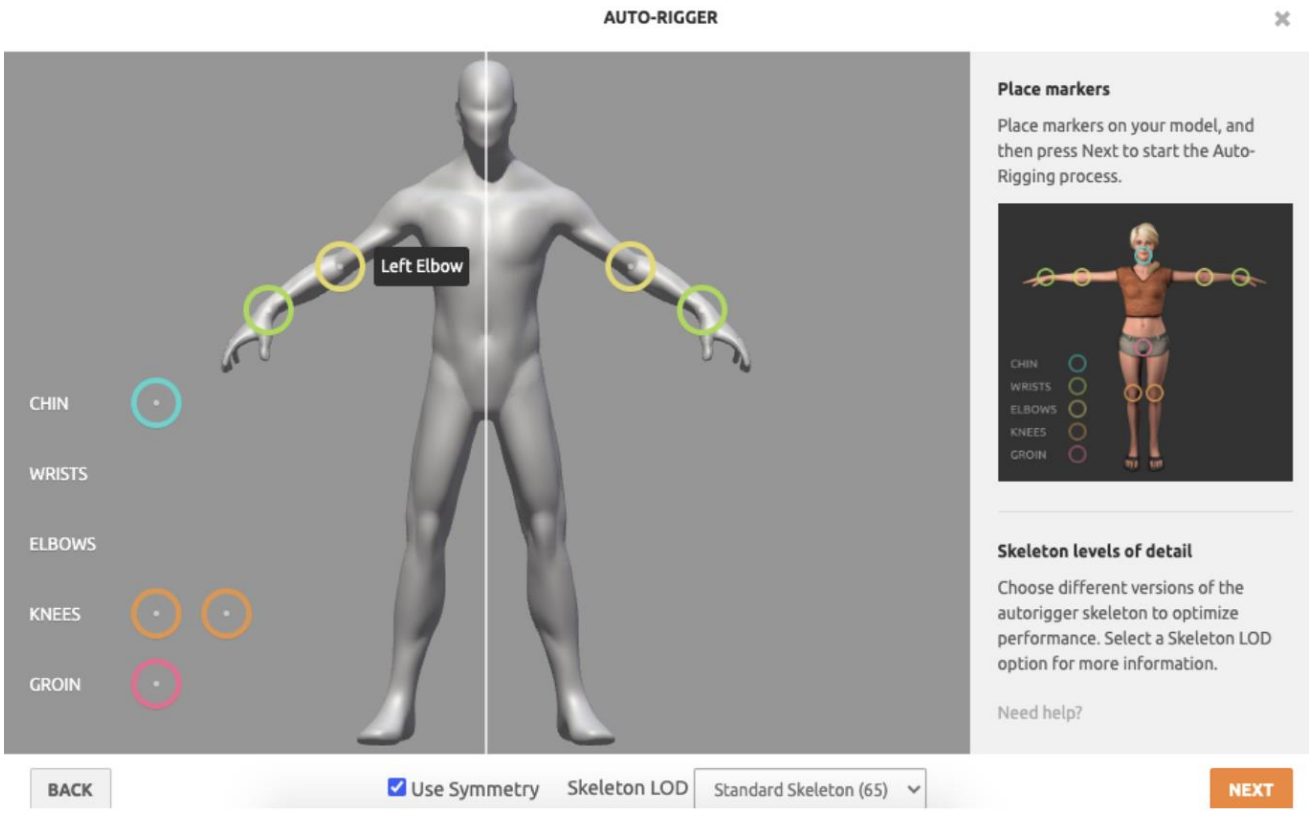



Рис. 4. Полуавтоматический алгоритм построения моделей Mixamo использует T-pose и указание ключевых точек

Этот метод сначала выполняет дискретный подбор размещения костей внутри объёма модели, а затем уточняет позиции непрерывной оптимизацией. В функции стоимости учитываются различные эвристики (например, положение суставов внутри тела, предотвращение выхода кости за пределы поверхности и др.), а её весовые коэффициенты оптимизируются методом опорных векторов (Support Vector Machine, SVM[4]) на обучающих примерах, который применяется для задач классификации и регрессии.

Таким образом, алгоритм комбинирует геометрические эвристики с элементами обучения, чтобы надёжно определить ключевые суставы для широкого класса персонажей. В результате определяется иерархия скелета – набор костей

[4] SVM работает путём нахождения гиперплоскости, которая максимально разделяет различные классы данных, максимизируя расстояние между ближайшими точками обучающих данных разных классов.

и связей между суставами, соответствующий форме модели. Согласно патентной документации Adobe, процесс включает генерацию упрощённой представительной сетки модели, разбиение её на сегменты (части тела) и определение ключевых точек (суставов); на основе этого формируется набор костей, который затем накладывается на исходную высокополигональную модель [8]. Система рассчитана на работу в облаке и задействует значительные вычислительные ресурсы (многопоточно на CPU), что позволяет выполнить весь процесс (~2 минуты на модель) онлайн.

Преимуществом подобных сервисов являются простота (не требует навыков риггера) и скорость (минуты на расчет), недостатком – ограниченная применимость (как правило, только гуманоидные двуногие) и типичные артефакты: неправильное влияние кости на соседние части тела, «сплющивание» в суставах и др., которые затем приходится вручную исправлять.

Широко используются и встроенные алгоритмы в пакетах 3D-моделирования. Autodesk Maya[5] предоставляет методы автоматического связывания кожи: *Heat Map Skinning* (на основе теплового распространения влияния костей) и *Geodesic Voxel Binding* (вокселизация объема модели для оценки ближайших костей) [5]. Эти подходы аналогично пытаются распределить веса по геометрии, исходя из расстояния до костей и топологии модели. В работе [1] был проведен подробный анализ инструментария Maya для скиннинга, перечислены его преимущества и недостатки. В частности, был описан ряд типичных дефектов, возникающих при использовании стандартных алгоритмов, например объемные потери при изгибе в области сустава или неправильное влияние при близко расположенных частях модели.

Для решения некоторых проблем предлагаются методы улучшения (например, добавление корректирующих деформеров), хотя полностью без участия человека алгоритмические методы не всегда работают. Тем не менее в сочетании с минимальной ручной правкой автоматические алгоритмы (Pinocchio, Mixamo, Maya Auto Skinning и др.) значительно ускоряют процесс риггинга по сравнению с полным ручным весовым раскрашиванием.

[5] Редактор трёхмерной графики. https://www.autodesk.com/products/maya/

## МЕТОДЫ СКИННИНГА НА ОСНОВЕ МАШИННОГО ОБУЧЕНИЯ

Современные достижения в области машинного обучения стимулировали появление методов, где нейронные сети выполняют скиннинг, то есть сами обучаются по примерам и затем предсказывают структуру рига или веса деформации для новых моделей. Одно из ключевых преимуществ таких методов состоит в способности учитывать сложные нелокальные зависимости и особенности формы, которые трудно учесть вручную прописанными правилами. Кроме того, нейросети можно обучить на большом количестве примеров ручной работы опытных риггеров, тем самым переняв их опыт.

Mixamo был одним из первых сервисов, где применили машинное обучение (Machine learning, ML) для автоматизации риггинга персонажей. Эта интеллектуальная система (в Adobe её называли *smart algorithms*) обобщает знания из множества примеров, позволяя надёжно обработать различные по форме модели. К примеру, указанный алгоритм [6] использовал обучаемую функцию штрафа для оценки качества встраивания скелета – фактически это применение *supervised learning* (обучение с учителем) для выбора оптимальной конфигурации костей. Таким образом, машинное обучение в Mixamo помогает определить расположение суставов и соответствие шаблону скелета по полученным входным данным модели. В то же время этап назначения весов скиннинга решается более стандартными средствами – эвристическими алгоритмами из графики, основанными на геометрии. Распределение влияния костей на поверхность – задача, хорошо решаемая методами интерполяции (например, тепловой диффузии) без необходимости обучать модель на данных. Эти алгоритмы гарантируют корректные деформации при движении сустава (вершины следуют за ближайшими костями с плавным затуханием влияния).

Таким образом, Mixamo сочетает машинное и классические подходы: ML – для высокоуровневого понимания формы (автоматическое определение скелета), эвристики – для низкоуровневого расчёта весов скиннинга и финальной привязки «кожа-кости». Этот гибридный подход позволил существенно автоматизировать рутинный труд риггеров, сохранив при этом физическую правдоподобность деформаций. Стоит отметить, что современные разработки движутся ещё дальше в сторону ML.

**RigNet** [9] – другая заметная работа в этом направлении. Эта нейросетевая система end-to-end автоматического риггинга, которая предсказывает как оптимальную скелетную структуру *произвольной топологии*, так и соответствующие ей веса привязки поверхности «кожи» (skin-weights) непосредственно по входному 3D-мешу. На вход подается 3D-модель персонажа (без скелета), на выходе алгоритм генерирует скелет (с суставами и топологией, близкой к той, что выбрал бы человек-аниматор) и рассчитывает веса привязки поверхности к костям (рис. 5).

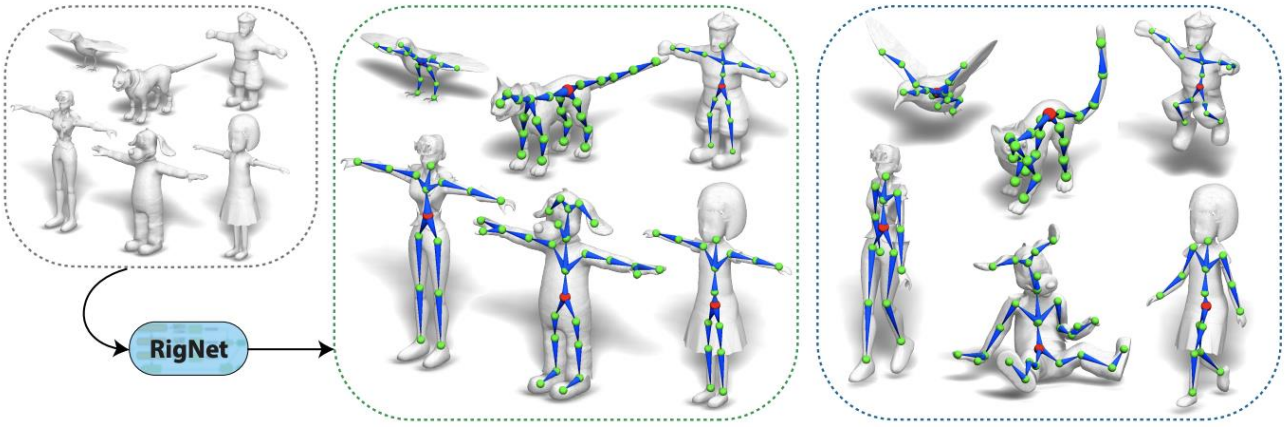



Рис. 5. RigNet создает анимационный скелет и веса кожи,
соответствующие структуре сочленений входной 3D-сетки персонажа [9]

Архитектура RigNet основана на глубокой сети, которая оперирует непосредственно на графе меша, без жёстких предположений о классе формы или структуре персонажа. Сеть обучена на большом и разнообразном наборе моделей с готовым ригом, то есть моделей, содержащих полную информацию о геометрии, скелетной структуре и весовых коэффициентах. Результаты RigNet превосходят все результаты предыдущих алгоритмических подходов: в количественном сравнении с эталонными ручными ригами RigNet показал наилучшее совпадение, а в качественном отношении автоматически полученные риги позволяют правдоподобно позировать и анимировать модели разных типов. Иначе говоря, нейросеть научилась расставлять кости «как человек» и привязывать кожу так, что в движении персонаж выглядит естественно.

В Mixamo используется более традиционный шаблонный подход, ограниченный человеческой анатомией и проверенный на практике, что обеспечивает

Mixamo быстроту и универсальность автоматического риггинга на уровне индустриального стандарта. Другой подход, фокусирующийся именно на назначении весов, представлен в работах типа **NeuroSkinning** [10] и **SkinningNet** [11]. Эти методы предполагают, что скелет модели уже известен (задан), и решают задачу распределения вершинных весов между костями с помощью глубоких нейросетей (как правило, графовых сверточных сетей, учитывающих структуру меша и скелета).

**SkinningNet** использует двухпотоковую графовую нейросеть (рис. 6): один поток обрабатывает геометрию меша, другой – структуру скелета; затем признаки объединяются для предсказания весов привязки.

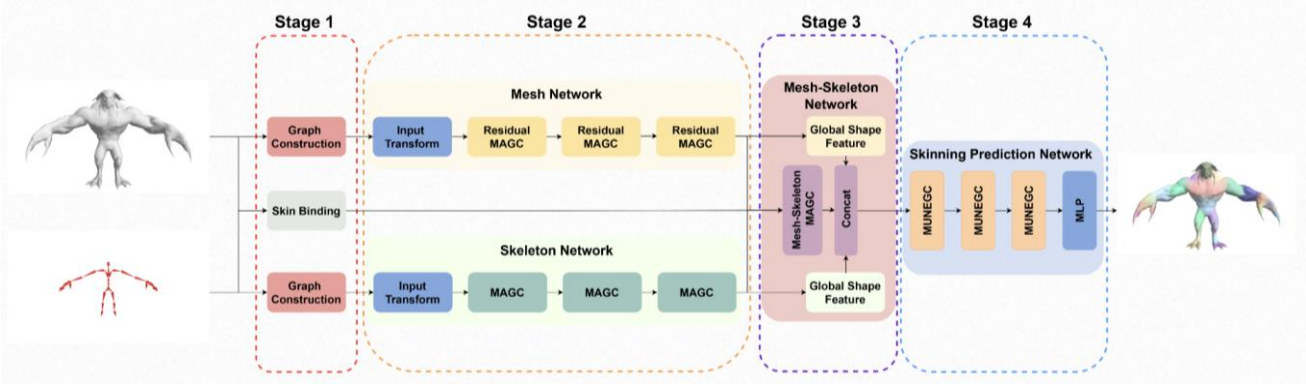



Рис. 6. Двухпотоковая графовая конволюционная нейронная сеть SkinningNet автоматически генерирует веса скинов для входной сетки и связанного с ней скелета [11]

За счет специального слоя Multi-Aggregator GCN эта сеть лучше обобщает на новых топологиях и сложных формах. По нашим данным, SkinningNet смогла снизить ошибку деформации меша более чем на 20% по сравнению с лучшими предыдущими подходами, а также эффективнее переносит обучение на персонажей новых доменов (не встречавшихся в обучающей выборке).

**NeuroSkinning** аналогично применяет глубокий графовый подход для расчета скиннинговых весов производственных персонажей, демонстрируя качество, приемлемое для индустрии (рис. 7).

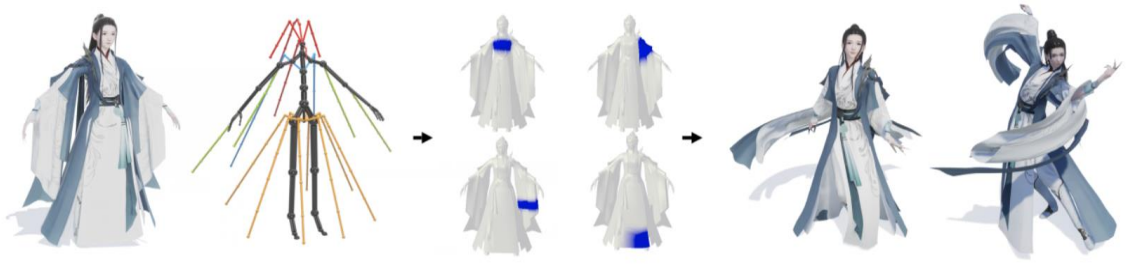


Рис. 7. Метод глубокого обучения для автоматического расчета веса кожи NeuroSkinning при скелетной деформации производственных персонажей [10]

Отметим также работу [12]. В ней представлена новая дата-ориентированная методика, позволяющая подготовить любую 3D-гуманоидную модель к анимации менее чем за одну секунду, независимо от её формы и позы (рис. 8).

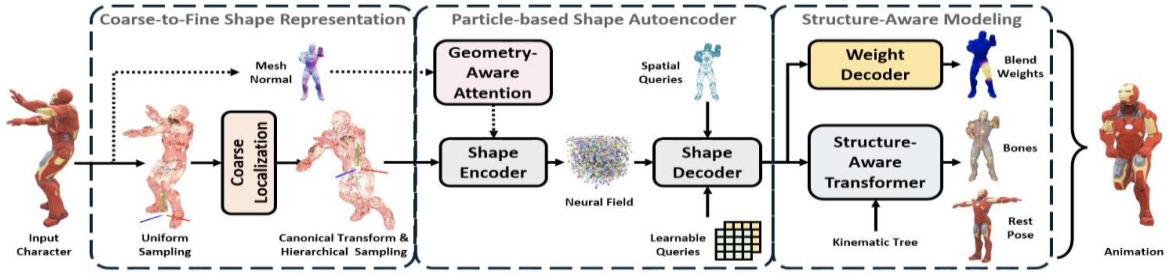



Рис. 8. Make-It-Animatable делает любую 3D-модель гуманоида готовой к анимации персонажа менее чем за секунду [12]

Предлагаемая система автоматически генерирует высококачественные веса скиннинга, костную структуру и преобразования поз. Используя автоэнкодер формы на основе частиц, подход поддерживает различные 3D-представления, включая сетки и трёхмерные гауссовые функции (3D Gaussian splats[6]). Кроме того, для обеспечения точности и надёжности применяются иерархическое представление и стратегия моделирования с учётом структуры, даже для персонажей с нестандартными скелетными структурами. Авторы утверждают, что предложенный

[6] 3D Gaussian splatting — метод визуализации объёмных данных, позволяющий создавать фотореалистичные сцены в реальном времени на основе ограниченного набора изображений. Вместо использования традиционных полигонов этот подход представляет сцену с помощью миллионов трёхмерных гауссовых функций (сплэтов), каждая из которых характеризуется положением, цветом, размером и прозрачностью. Такая техника обеспечивает плавное и непрерывное отображение сложных объектов и сцен.

подход значительно превосходит существующие методы как по качеству, так и по скорости.

Необходимо также упомянуть работу [13], в которой предложено использование контроллеров обратной кинематики (Inverse Kinematics, IK) для модификации анимации скелетных персонажей, что позволяет улучшить реалистичность движений и гибкость анимации. Интеграция таких подходов в системы скиннинга на основе машинного обучения может значительно повысить качество создаваемых анимаций. Эти нейросетевые методы особенно эффективны в сложных случаях, где классические алгоритмы дают артефакты: например, при тонких деталях геометрии, близко расположенных конечностях, нестандартных пропорциях. Сеть, обученная на подобных случаях, способна «догадаться», как правильно распределить веса, тогда как жёсткий алгоритм может ошибиться.

Важно отметить, что успех методов машинного обучения во многом зависит от наличия обширных датасетов для обучения, т. е. от множества моделей с корректно настроенными ригами. Формирование таких наборов – непростая задача, но она активно решается. В частности, сообщество исследователей использует наборы моделей из игр и онлайн-библиотек (например, Mixamo, Adobe Fuse), а также генерацию синтетических данных. В [4] описан процесс разработки универсального инструментария для генерирования 3D-синтетических датасетов, где отмечена эффективность такого подхода для обучения нейросетей различным задачам. Генерируя тысячи вариаций 3D-гуманоидов (в том числе с помощью MetaHuman, Blender[7] и игровых движков), можно обучить нейросеть решать задачу скиннинга более универсально. Текущие исследования по автоматизации создания игровых персонажей затрагивают и нейронные сети как инструмент риггинга, например, интегрируемые в Maya для ускорения скиннинга. Таким образом, ML-методы уже сегодня способны выполнять скиннинг на уровне, близком к ручной работе, и ожидается, что их роль будет расти.

[7] Профессиональное свободное и открытое программное обеспечение для создания трёхмерной компьютерной графики. https://www.blender.org/

## СРАВНИТЕЛЬНЫЙ АНАЛИЗ ПОДХОДОВ

Технология MetaHuman отлично подходит для реалистичных человеческих персонажей. Она обеспечивает непревзойденную детализацию и готовые решения для мимики, но ограничена рамками человекоподобной анатомии. Алгоритмические методы (Pinocchio, Mixamo, Maya Auto-Rig и др.), как правило, рассчитаны на гуманоидов или стандартных существ (четвероногих персонажей, робототехнику и т. д.), их шаблоны можно подобрать под нужный класс.

Некоторые алгоритмы более универсальны: например, RigNet позиционируется как независимый от класса формы (обучался на разнообразных моделях), в тестах он успешно осуществлял настройку скелетной системы как людей, так и животных и фантастических существ. Таким образом, MetaHuman используется для получения датасета для человека, а алгоритмы/ML – для всего остального (и человека тоже, если нужна альтернатива).

### Качество деформаций

MetaHuman задает эталон качества для человекоподобных моделей – благодаря дополнительным костям (для мышц, суставов) и морфам, деформации тела и лица выглядят очень реалистично, без эффектов «конверсии объема» (collapsing volume) в суставах. Алгоритмические методы базового уровня (линейный скиннинг по ближайшим костям) могут страдать от эффекта «конфетной обёртки» (candy-wrapper) при кручении конечностей (рис. 9) или от провалов объёма при сгибании локтя/колена (рис. 10).

Однако многие улучшения, такие как «двойной кватернион» (dual quaternion skinning), коррекция поз, физически обоснованные модели мышц, могут быть наложены сверху вручную. Нейросетевые подходы имеют потенциал автоматического учёта таких эффектов: например, если в обучающих данных были правильные деформации с сохранением объёма, сеть может перенести эти свойства на новые модели. SkinningNet, как указывалось, снижает среднюю ошибку деформации, что говорит о более точном воспроизведении формы.

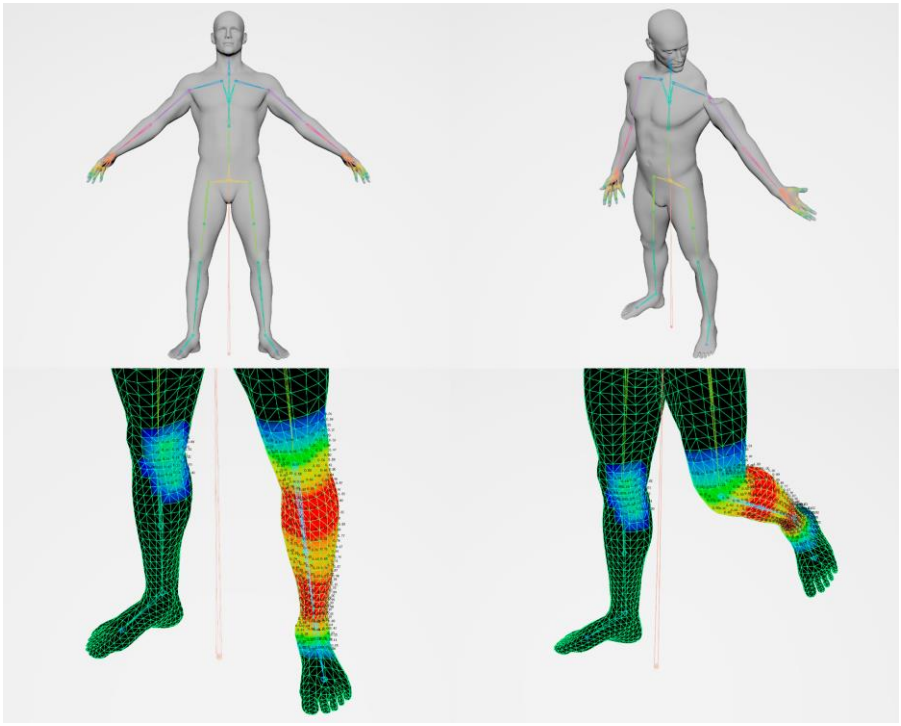


Рис. 9. Сверху: эффект «конфетной обёртки» на трёхмерной модели, снизу: неправильное распределение весов на конечности

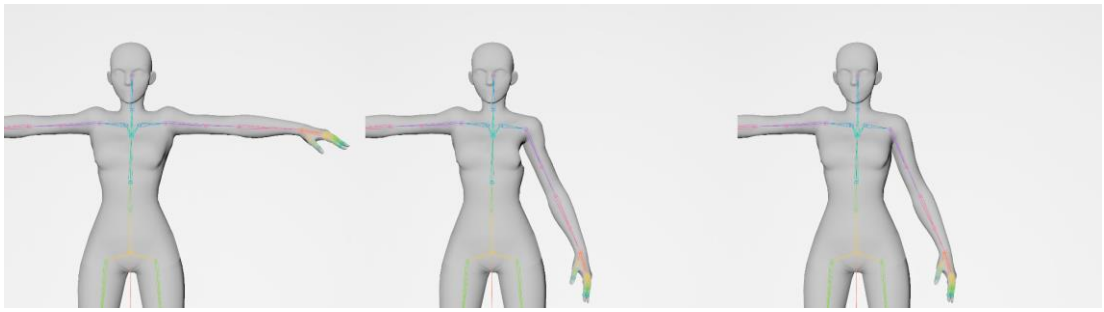


Рис. 10. Потеря объема при сгибе конечности

В целом по качеству можно расположить так: **ручной риг + MetaHuman** – наивысший (кинематографический) уровень; **ML-риг** (RigNet и др.) уже близок к ручному; **алгоритмический авториг** может потребовать дальнейшей доработки для сложных участков модели.

**Скорость и трудозатраты**

Все рассмотренные автоматизированные подходы работают значительно быстрее ручного скиннинга. MetaHuman Creator позволяет получить готового персонажа за считанные минуты. Алгоритмы вроде Pinocchio также укладываются в

секунды или минуты на одну модель. Нейронные сети после обучения выполняют предсказание почти мгновенно – сотые доли секунды на выполнение вывода, то есть могут осуществлять настройку скелетной системы («риггировать») для 3D-моделей в режиме реального времени. Различие проявляется в трудозатратах на подготовку: MetaHuman требует либо воспользоваться редактором (несколько часов на тонкую настройку черт лица вручную, если нужен специфичный облик), либо подготовить 3D-скана/модели лица для загрузки. Алгоритмические методы зачастую требуют минимальной подготовки (например, ориентации модели, пометки суставов в Mixamo) – это минуты работы. ML-методы требуют предварительного обучения модели на большом датасете (это разовая вычислительно затратная процедура), но потом могут риггировать поток моделей автоматически без участия человека. В условиях производства игр/фильмов, где типовые персонажи риггируются многократно, нейросетевой подход может окупиться.

**Доступность и интеграция**

MetaHuman интегрирован с Unreal Engine (UE4/UE5) и благодаря **Control Rig** и IK-ретаргетингу в UE5 позволяет удобно перенасыщать анимации между персонажами и применять готовые мокап-анимации. Это делает его привлекательным для тех, кто работает на UE. Алгоритмические инструменты встроены в популярные DCC-пакеты (Maya, Blender) или доступны как облачные сервисы (Mixamo), что тоже довольно удобно – художники могут ими воспользоваться без глубоких знаний программирования. ML-решения пока представлены в виде научных прототипов или отдельных плагинов. Например, имеются исследовательские плагины к Maya на основе нейросетей, но широкого промышленного внедрения они еще не получили. Тем не менее ряд компаний (Adobe, Autodesk) явно движется в сторону ML в инструментах: так, Adobe презентовала прототип *AI Auto-Rigger*, а Autodesk инвестирует в технологии машинного обучения для Character Generator. В академической среде уже используются стандартные бенчмарки[8] по автомати-

[8] Бенчмарк (англ. benchmark) – стандартный набор тестов и методик, позволяющих объективно оценить и сравнить производительность, точность и эффективность различных систем, алгоритмов или моделей.

ческому риггингу, позволяющие сравнивать традиционные алгоритмические методы и подходы на основе машинного обучения (при этом RigNet и аналогичные решения обычно показывают более высокую точность).

**Валидация на базе экспериментов**

Научные работы предоставляют количественное сравнение подходов. RigNet (2020) показал, что автоматически предсказанные им скелеты и весовые распределения близки к ручным: отклонения минимальны, а персонажи успешно принимают сложные позы. SkinningNet (2022) продемонстрировал улучшение метрик деформации по сравнению с алгоритмом веса по тепловому влиянию и даже по сравнению с NeuroSkinning (2019), приблизившись по точности к «эталонной» привязке. С другой стороны, алгоритм Pinocchio (2007) в свое время доказал надежность, автоматически правильно обработав ~80% разнообразных моделей без вмешательства, это отличный результат, показавший возможность авторигинга задолго до эпохи глубокого обучения. MetaHuman как продукт прошел проверку индустрией: персонажи MetaHuman используются в кинопроизводстве, игровых проектах и научных экспериментах (например, для генерации датасетов), демонстрируя высокую реалистичность. В эксперименте [14] по реконструкции 3D-модели человека по единственной фотографии был применен такой датасет – фактически был выполнен авториггинг реконструированной фигуры. Это подтверждает, что даже для нейросетевых методов реконструкции (например, по изображению) задача скиннинга легко решается существующими авториг-инструментами, интегрируя результаты в анимацию.

**ВЫВОДЫ О ПРИМЕНИМОСТИ**

**MetaHuman** представляет собой эффективное решение для автоматизированного скиннинга человекоподобных 3D-моделей, обеспечивая высочайшее качество рига «из коробки» и минимальные затраты времени. Его использование целесообразно, когда нужен быстрый результат в рамках Unreal Engine и когда модель соответствует человеческой морфологии.

**Алгоритмические методы** остаются важной альтернативой: они хорошо зарекомендовали себя для типовых персонажей, широко доступны и понятны.

В случаях, где MetaHuman неприменим (нестандартные существа, собственные скелеты или отсутствие UE), такие методы способны автоматически сгенерировать скелет и веса, сократив ручной труд на порядки, хотя могут потребовать некоторой доводки, особенно в сложных случаях.

**Методы на базе машинного обучения** – наиболее *продвинутый* на сегодня подход. Они уже показывают качество риггинга, сравнимое с ручной работой, и продолжают улучшаться по мере накопления данных и совершенствования моделей. Нейросети обещают более интеллектуальный скиннинг: учитывающий контекст формы, избегающий типовых ошибок классических алгоритмов и приспосабливающийся к новым видам персонажей. Однако внедрение ML-риггинга в повседневные конвейеры пока ограничено сложностью обучения и недостатком готовых инструментов.

Итак, для задачи автоматизированного скиннинга нет универсального «лучшего» решения – выбор зависит от контекста. MetaHuman оптимален для реалистичных людей в UE-среде, классические алгоритмы – для быстрого авторигга широкого спектра моделей с контролем со стороны художника, а нейросетевые системы – перспективное направление, которое, вероятно, станет золотым стандартом в будущем, когда технологии дозреют. Комбинация подходов тоже возможна: например, можно использовать алгоритмический авториг как базу, а затем применять нейросетевые корректировки для улучшения деформаций. В настоящее время технологии скиннинга активно развиваются, и появление инструментов вроде MetaHuman уже сейчас существенно облегчает путь «от идеи к реализации» 3D-персонажа, интегрируя его в анимационный процесс практически мгновенно. Такое слияние индустриальных и научных достижений делает автоматизированный скиннинг все более целесообразным и эффективным на практике.

## СОЗДАНИЕ СИНТЕТИЧЕСКОГО ДАТАСЕТА

Представим метод создания синтетического датасета на основе 3D-моделей MetaHuman для обучения нейронной сети, которая будет оптимизировать процесс скиннинга 3D-моделей. Этот датасет должен быть достаточно разнообразным, чтобы нейронная сеть могла эффективно обучаться на нём, обеспечивая

высокое качество скиннинга для различных типов 3D-моделей и различных поз.

**Требования к датасету**

Для обучения нейросети, предназначенной для автоматического риггинга, необходим специализированный датасет, соответствующий ряду требований, в частности:

- 3D-модели
  - должны быть представлены в распространённых форматах (.obj, .fbx, .binvox и др.);
  - необходимо предоставить разнообразие форм, размеров и уровней детализации, чтобы обеспечить высокое качество обучения.
- Данные для риггинга
  - включают трёхмерные координаты суставов для каждой модели;
  - содержат иерархическую структуру рига, отображающую связи «родитель — потомок» между суставами.
- Необходимы дополнительные сведения для учёта объёмных геодезических расстояний в алгоритме нейросети, воксельные представления моделей.

Ключевые характеристики датасета:

1. Точность и качество
   - Все модели и сопутствующие данные должны быть максимально точными и полными. Ошибки в позиционировании суставов или распределении весов могут негативно повлиять на процесс обучения.
   - Данные проходят ручную или автоматическую проверку для исключения дубликатов и ошибок.

2. Объём данных

- Для эффективного обучения требуется большое количество разнообразных моделей.
- Датасет должен быть разделен на обучающую, валидационную и тестовую выборки для корректной оценки производительности модели.

**Структура датасета**

Структуру датасета Dataset_Rigging_preproccessed можно представить следующей иерархией папок и файлов:

```
Dataset_Rigging_preproccessed/
├── vox/
│ └── model.binvox
├── volumetric_geodesic/
│ └── model_volumetric_geo.npy
├── rig_info_remesh/
│ └── model.txt
├── rig_info/
│ └── model.txt
├── pretrain_attention/
│ └── model.txt
├── obj_remesh/
│ └── model.obj
├── obj/
│ └── model.obj
├── val_final.txt
├── train_final.txt
└── test_final.txt
```

Здесь vox включает воксельные представления 3D-моделей в формате .binvox; volumetric_geodesic содержит данные объёмных геодезических измерений, представленные в виде массивов NumPy (.npy); rig_info_remesh включает информацию о переработанной сетке, а также данные о суставах и весовых коэффициентах кожи; rig_info предоставляет подробные сведения о риге, включая:

- списки суставов с их трёхмерными координатами,
- информацию о весах кожи, привязанных к каждому суставу,

- иерархическую структуру рига, описывающую связи «родитель — потомок» между суставами;

pretrain_attention содержит бинарные маски, определяющие области внимания для предварительного обучения; obj_remesh включает переработанные 3D-модели в формате .obj; obj содержит оригинальные или обработанные 3D-модели в форматах .obj; val_final.txt, train_final.txt, test_final.txt – файлы, содержащие списки числовых идентификаторов, соответствующих наборам данных для валидации, обучения и тестирования.

**Выбор и подготовка базовых 3D-моделей**

Первым этапом создания синтетического датасета (рис. 11) являются выбор и подготовка базовых 3D-моделей.

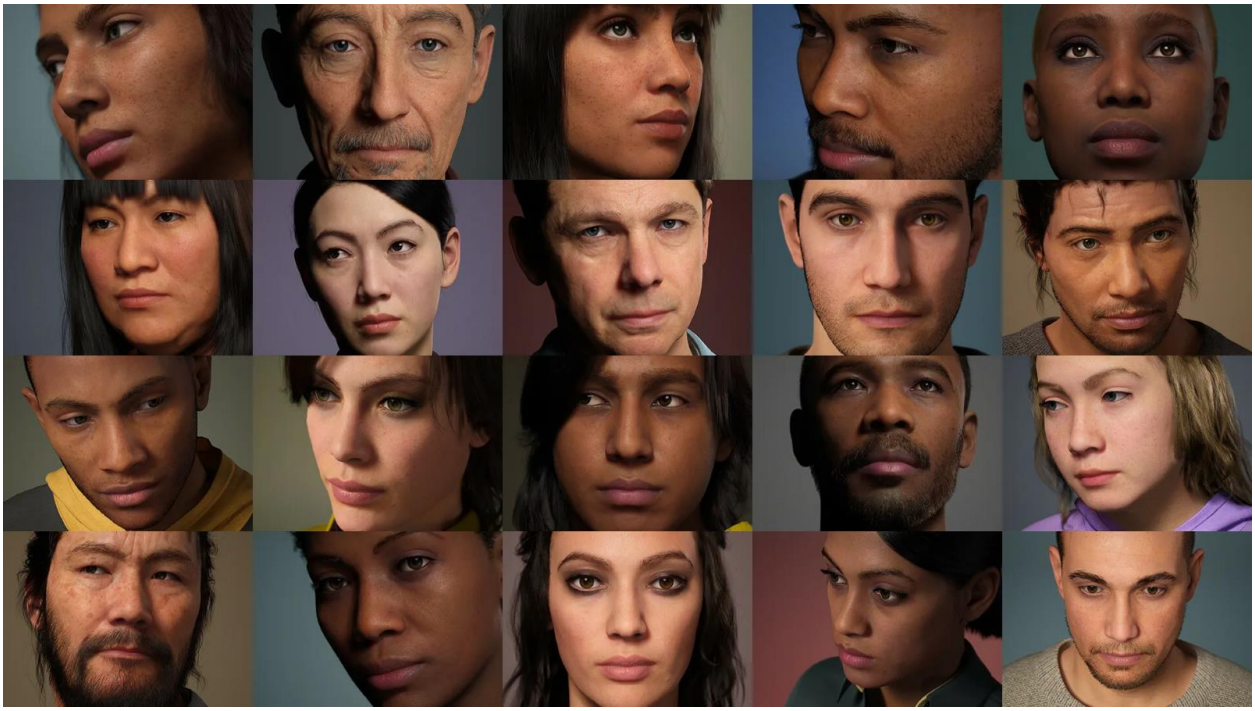


Рис. 11. Разнообразные модели MetaHuman

В качестве основы были выбраны модели MetaHuman от Epic Games, обладающие следующими преимуществами:

- высокая детализация и реалистичность моделей;
- возможность гибкой настройки параметров внешности;
- наличие готовой скелетной структуры с правильно настроенной системой весов;

- поддержка различных мимических выражений и движений.

Для увеличения разнообразия датасета (более 50 уникальных персонажей) были настроены различные характеристики моделей, включая:

- пол (мужской, женский);
- возраст (от молодого до пожилого);
- этническую принадлежность;
- тип телосложения (от худощавого до полного);
- мимические особенности.

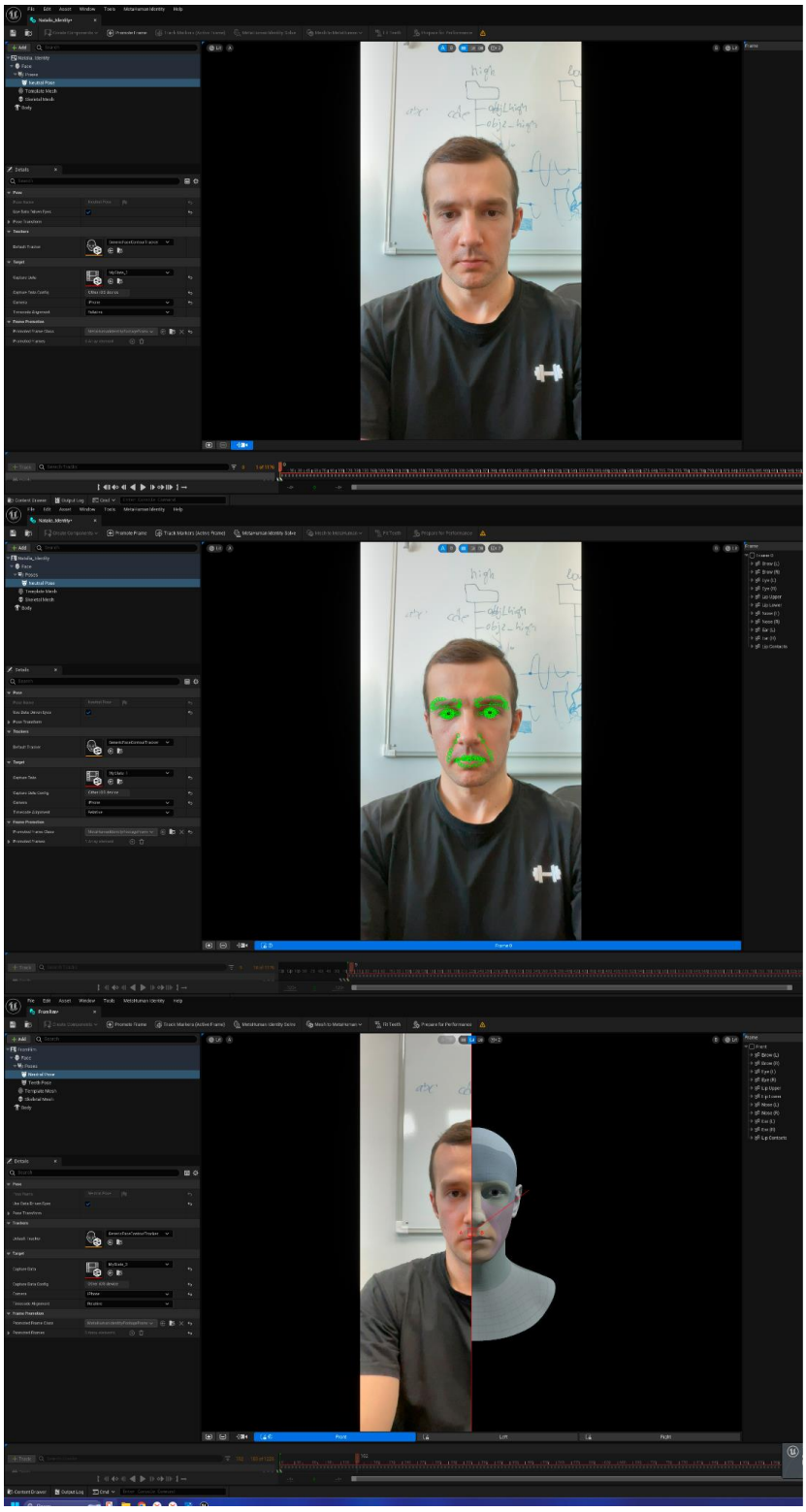


Рис. 12. Захват движений

**Создание синтетических анимаций**

На втором этапе были созданы синтетические анимации для подготовленных 3D-моделей, для чего использовались:

- библиотеки стандартных анимаций (ходьба, бег, прыжки и т. д.);
- данные захвата движений (Motion Capture), полученные с помощью специализированного оборудования (рис. 12);
- процедурная генерация движений с использованием физического моделирования.

Для каждой модели было сгенерировано по 20–30 различных анимационных последовательностей, включающих:

- простые движения (повороты, наклоны);
- сложные последовательные движения (танцы, спортивные упражнения);
- экстремальные позы для тестирования предельных возможностей скиннинга.

Особое внимание уделялось созданию анимаций, вызывающих значительные деформации модели, что позволяет более точно оценить качество скиннинга при сложных движениях.

**Оптимизация процесса скиннинга**

Следующим этапом стала оптимизация процесса скиннинга для подготовленных моделей и анимаций, в рамках которого:

1. разработан метод автоматического анализа качества скиннинга, основанный на определении проблемных областей (артефактов) при деформациях;
2. создан алгоритм автоматической коррекции весов скиннинга для минимизации визуальных артефактов;
3. сформированы аннотированные данные, включающие:
    - исходную геометрию модели;
    - скелетную структуру;
    - матрицы трансформации костей;
    - карты весов скиннинга;
    - результирующую деформированную геометрию.

**Метрика оценки качества скиннинга**

Для оценки качества скиннинга была разработана специальная метрика, учитывающая как геометрические характеристики деформаций, так и визуальное восприятие результатов анимации.

1. Геометрические характеристики деформаций
    - Сохранение объема – измерение изменения объема сетки при деформации (идеальная деформация сохраняет объем);
    - Гладкость поверхности – оценка локальных искажений нормалей поверхности во время анимации;
    - Лапласиан деформации – анализ сохранения локальных геометрических деталей;
    - Метрика растяжения/сжатия – измерение неестественного растяжения или сжатия полигональной сетки.
2. Визуальное восприятие результатов
    - Перцептивная оценка артефактов – взвешенная система баллов для различных типов видимых артефактов:
        - «схлопывание» суставов – 0.35;
        - «конфетная обёртка» – 0.25;
        - проникновение геометрии – 0.20;
        - неестественные изгибы – 0.20.
3. Интегрированный показатель качества

$$Q=\alpha G+\beta V,$$

где G – нормализованный показатель геометрической точности (0-1), V – нормализованный показатель визуального качества (0-1), α и β – весовые коэффициенты (α+β=1), определяющие приоритет геометрической точности или визуального восприятия.

4. Сравнение с эталоном происходит следующим образом:
    - Вычисление среднеквадратичного отклонения от эталонной анимации, созданной вручную экспертами;
    - Измерение отклонения вершин от ожидаемых позиций в ключевых позах.

Метрика автоматизирована и может применяться как для оценки качества

обучения нейронной сети, так и для сравнения результатов с другими методами скиннинга.

Такой подход позволяет:

1. количественно измерить, насколько автоматический скиннинг приближается к профессиональной ручной работе,
2. выявить проблемные области, требующие дополнительной оптимизации,
3. иметь объективный критерий для сравнения различных алгоритмов и итераций нейронной сети.

**Подготовка данных для обучения нейронной сети**

На заключительном этапе подготовки датасета все полученные данные были структурированы для обучения нейронной сети.

1. Исходные данные:
    - 3D-координаты вершин модели;
    - Топология модели (соединения между вершинами);
    - Иерархия скелета и его параметры;
    - Позиции суставов в пространстве.
2. Целевые данные:
    - Оптимизированные веса скиннинга;
    - Результирующие позиции вершин после деформации (рис. 13).

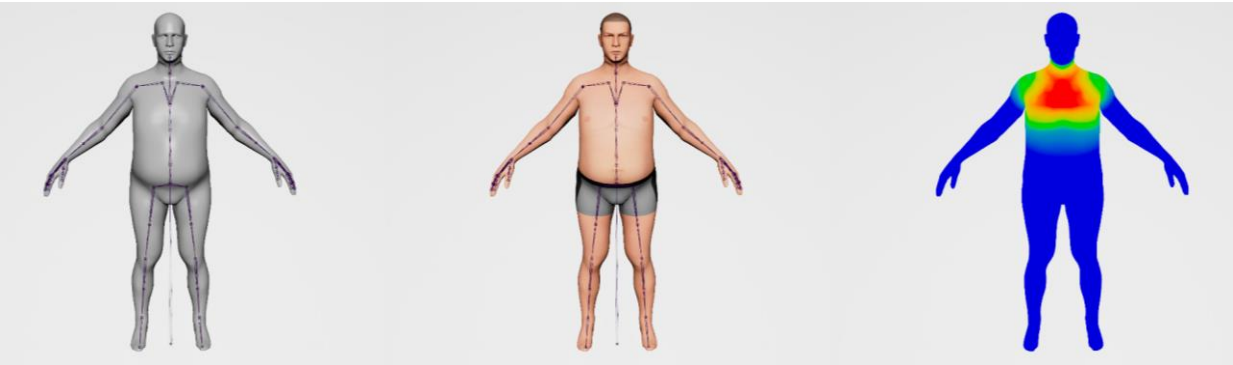


Рис. 13. Упрощенная трёхмерная модель должна иметь форму, идентичную исходной. На скелете должны быть удалены кости без анимации и деформирующие мелкие участки (лицевые мышцы и т. д.). Веса должны быть скопированы с исходной модели

При подготовке данных датасета были применены два способа упрощения моделей:

(1) Ручная ретопология с использованием алгоритмов уменьшения полигонов и создание упрощенной версии скелета (рис. 14).

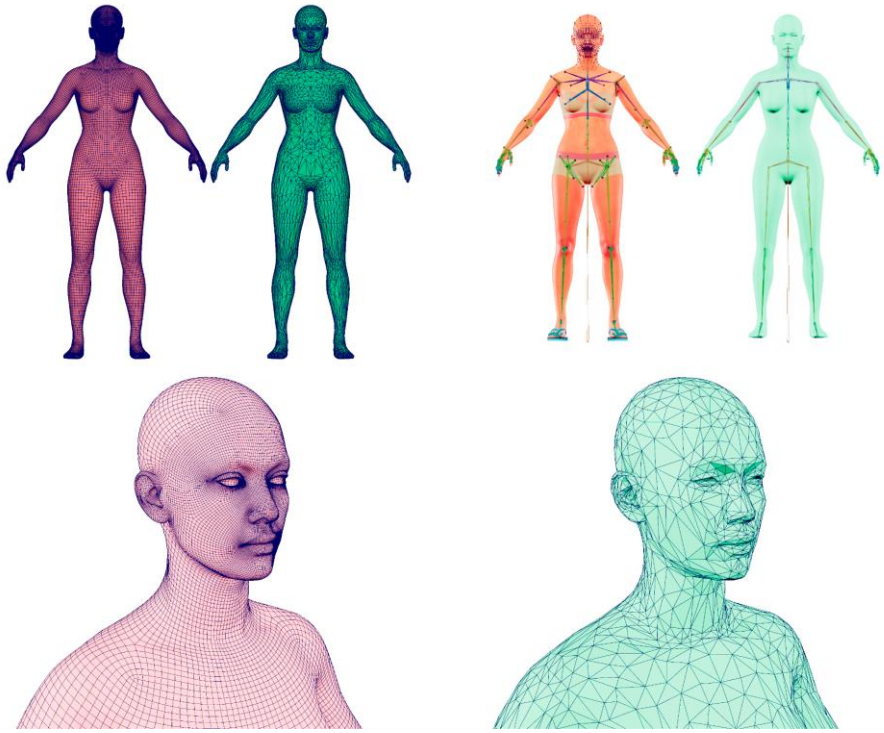


Рис. 14. Упрощенная модель и скелет с использованием ручных методов ретопологии

(2) Использование уровней детализации (level of details, LOD) моделей MetaHuman и создание упрощённого скелета с последующим копированием весов скина (рис. 15).

Датасет был разделен на обучающую (70%), валидационную (15%) и тестовую (15%) выборки для обеспечения корректной оценки обучения нейронной сети.

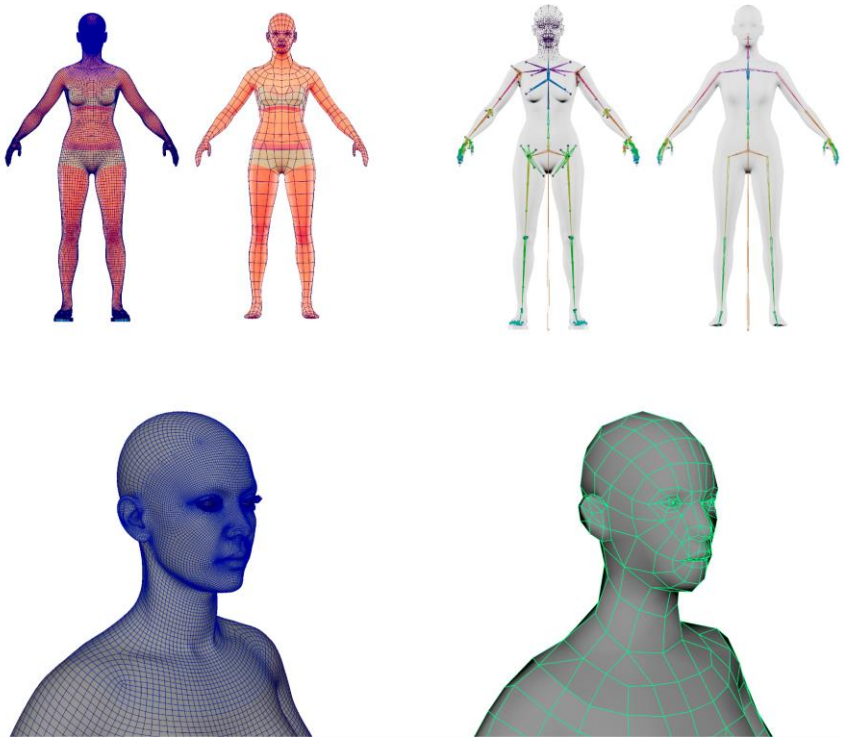


Рис. 15. Упрощённая модель и скелет с использованием LOD

**НЕЙРОННАЯ СЕТЬ ДЛЯ ОПТИМИЗАЦИИ СКИННИНГА**

Для решения задачи оптимизации скиннинга была разработана специализированная архитектура нейронной сети, учитывающая пространственную структуру 3D-моделей и скелетных систем, которая включает:

1. блок обработки геометрии модели на основе графовых нейронных сетей (Graph Neural Networks), позволяющий учитывать топологические особенности 3D-моделей;
2. блок анализа скелетной структуры, обрабатывающий иерархические связи между костями;
3. блок предсказания весов скиннинга, оптимизирующий распределение влияния различных костей на каждую вершину модели.

Предложенная архитектура позволяет эффективно обрабатывать модели с различной топологией и скелетной структурой, обеспечивая высокое качество скиннинга даже для сложных анимаций и экстремальных поз.

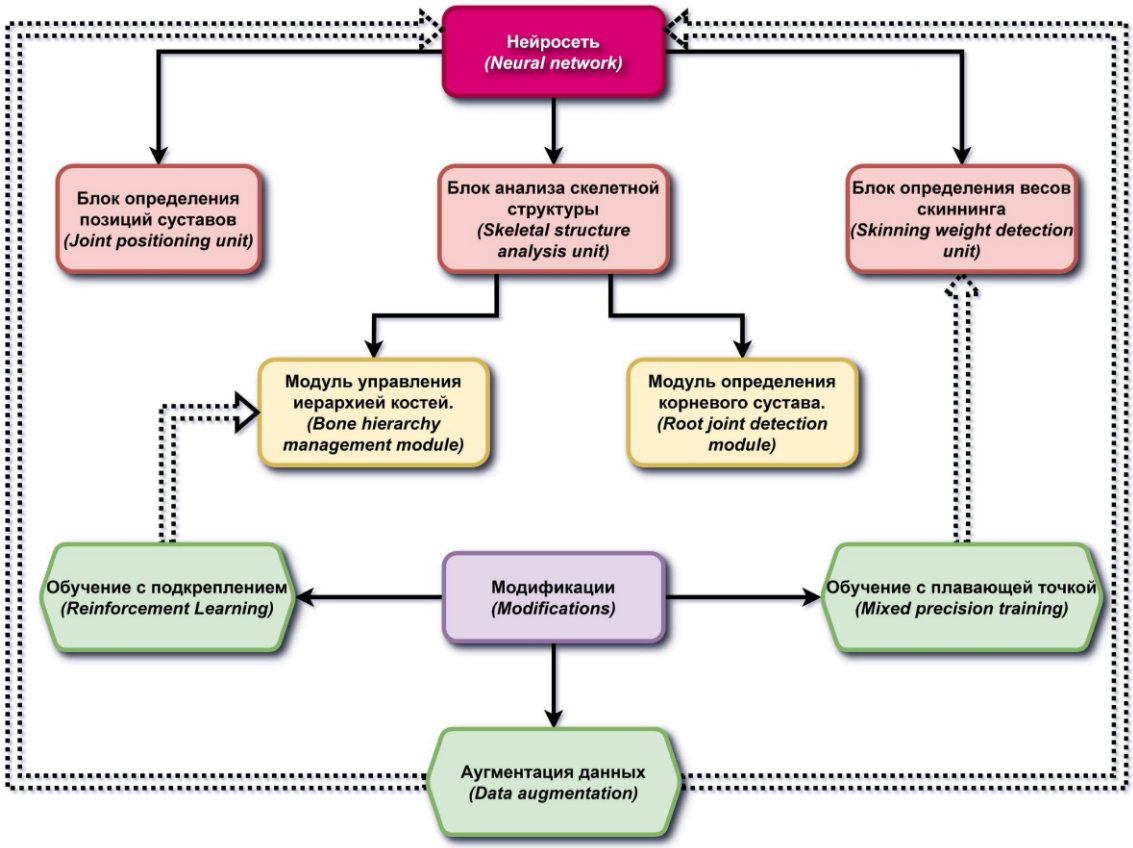



Рис. 16. Модификации нейронной сети RigNet

Были выбраны три основные модификации для улучшения производительности модели: внедрение обучения с подкреплением в модуль BoneNet, ускорение обучения с использованием смешанного обучения с плавающей точкой, а также аугментация данных (рис. 16).

## ЭКСПЕРИМЕНТАЛЬНЫЕ РЕЗУЛЬТАТЫ

Для оценки эффективности предложенного метода были проведены эксперименты на созданном синтетическом датасете. Обучение нейронной сети проводилось с использованием оптимизатора Adam с начальной скоростью обучения 0.001 и экспоненциальным снижением скорости обучения. Размер батча составлял 32 примера, обучение проводилось в течение 100 эпох.

Результаты экспериментов показали:

1. значительное сокращение времени на скиннинг новых моделей (в среднем в 8–10 раз по сравнению с традиционными методами);

2. высокое качество автоматически созданного скиннинга, сопоставимое с результатами ручной настройки опытными специалистами;
3. хорошую обобщающую способность нейронной сети – возможность корректно обрабатывать модели, отличающиеся от представленных в обучающей выборке.

Сравнительный анализ с существующими методами автоматического скиннинга показал превосходство предложенного подхода по следующим критериям:

- точность позиционирования вершин при деформациях;
- отсутствие визуальных артефактов при экстремальных позах;
- сглаженность переходов между различными позами при анимации;
- время, затрачиваемое на скиннинг новой модели.

**ЗАКЛЮЧЕНИЕ**

В результате настоящего исследования был создан обширный синтетический датасет на основе цифровых персонажей MetaHuman, предназначенный для оптимизации скиннинга 3D-моделей. Проведённые эксперименты продемонстрировали, что использование этого датасета значительно улучшает качество деформаций персонажей по сравнению со стандартными методами скиннинга. Достигается снижение типичных дефектов скиннинга (например, провалов или искажений геометрии в области суставов), что обеспечивает более реалистичное движение модели. Полученные результаты подтверждают эффективность методологии, основанной на синтетических данных, для решения сложных задач трёхмерного моделирования и анимации, что согласуется с успешным опытом применения подобных подходов в смежных областях.

Кроме того, разработанный комплексный подход к генерации данных с помощью MetaHuman позволил преодолеть ряд ограничений традиционного скиннинга. Ручная настройка весовых коэффициентов при скиннинге — чрезвычайно трудоёмкая и сложная задача для художника 3D-моделей. При этом стандартные подходы неизбежно приводят к появлению характерных дефектов деформации модели. Предложенный метод устраняет эти проблемы посредством нейросетевой модели, обученной на синтетическом наборе данных. Эта модель автомати-

чески прогнозирует и корректирует веса скиннинга, уменьшая артефакты деформации. Такой подход согласуется с современными идеями применения машинного обучения для улучшения качества скиннинга и ускорения процесса скиннинга моделей.

Благодаря широкому разнообразию сгенерированных примеров (множества поз, вариантов телосложения и сцен) наш метод обеспечивает стабильное качество деформаций для различных типов движений и персонажей, повышая надёжность алгоритма в широком спектре сценариев анимации.

Также перспективным направлением является интеграция разработанного метода в популярные графические движки (такие как Unreal Engine и Unity) и системы разработки. Это позволит не только автоматизировать процесс скиннинга в рамках единых производственных конвейеров, но и обеспечить более тесное взаимодействие между инструментами моделирования, анимации и машинного обучения, что существенно ускорит цикл разработки высококачественных 3D-персонажей.

**СПИСОК ЛИТЕРАТУРЫ**

1. *Газизов Р.Р., Шубин А.В.* Процедурные методы скиннинга гуманоидных персонажей // Электронные библиотеки. 2022. Т. 25, No. 5. С. 404–440. https://doi.org/10.26907/1562-5419-2022-25-5-404-440

2. Epic Games. New release brings Mesh to MetaHuman to Unreal Engine … MetaHuman Mesh to MetaHuman Announcement. 2022. URL: https://www.unrealengine.com/en-US/blog/new-release-brings-mesh-to-metahuman-to-unreal-engine-and-much-more

3. Epic Games Forums. Skinning Method in MetaHuman (developer reply), April, 2021. // Unreal Engine forum. 2021. URL: https://forums.unrealengine.com/t/skinning-method-in-metahuman/226222

4. *Кугуракова В.В., Абрамов В.Д. и др.* Генерация трехмерных синтетических датасетов // Электронные библиотеки. 2021. Т. 24. No. 4. С. 622–652. https://doi.org/10.26907/1562-5419-2021-24-4-622-652

5. *Сулейманова Е.А., Газизов Р.Р., Кугуракова В.В. и др.* От идеи до реально-

сти: процесс создания одежды для персонажа компьютерной игры // Известия вузов. Технология текстильной промышленности. 2024. №5(413). С. 13–19. https://doi.org/10.47367/0021-3497_2024_5_13

6. *Baran I., Popović J.* Automatic Rigging and Animation of 3D Characters. ACM Trans. Graph. 2007. T. 26. No. 3. C. 72-es.

*7. McGrane C.* Auto-rigging? Still something only Mixamo can do? // ThreeJS Forum. 2022. URL: https://discourse.threejs.org/t/auto-rigging-still-some thing-only-mixamo-can-do/43709/8

8. Automatic rigging of three dimensional characters for animation: пат. США № 11,170,558 B2 / Adobe Inc.; опубл. 09.11.2021. URL: https://patents.google.com/patent/US11170558B2/en

9. *Xu Z., Zhou Y., Kalogerakis E., Landreth C., Singh K.* RigNet: Neural Rigging for Articulated Characters //arXiv preprint arXiv:2005.00559. 2020.

10. *Liu L., Zheng Y., Tang D., Yuan Y., Fan C., Zhou K.* Neuroskinning: Automatic skin binding for production characters with deep graph networks //ACM Transactions on Graphics (ToG). 2019. Vol. 38. No. 4. P. 1–12.

11. *Mosella-Montoro A., Ruiz-Hidalgo J.* Skinningnet: Two-stream graph convolutional neural network for skinning prediction of synthetic characters // Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2022. P. 18593–18602.

12. *Guo Z. et al.* Make-It-Animatable: An Efficient Framework for Authoring Animation-Ready 3D Characters //arXiv preprint arXiv:2411.18197. 2024. URL: https://jasongzy.github.io/Make-It-Animatable/

13. *Gazizov R., Shubin A.* Modification of Skeletal Character Animation Using Inverse Kinematics Controllers //2024 International Russian Smart Industry Conference (SmartIndustryCon). IEEE, 2024. P. 553–557. https://doi.org/10.1109/SmartIndustryCon61328.2024.10515984

14. *Тарасов А.С., Кугуракова В.В.* Реконструкция трехмерной модели человека по единственному изображению// Электронные библиотеки. 2021. Т. 24, № 3. С. 485–504. https://doi.org/10.26907/1562-5419-2021-24-3-485-504

# METAHUMAN SYNTHETIC DATASET FOR OPTIMIZING 3D MODEL SKINNING

**R. R. Gazizov[1] [0000-0002-8349-264X], M. D.Belov[2] [0009-0008-7680-1839]**

[1, 2]*Institute of Information Technologies and Intelligent Systems,*
*Kazan Federal University*

[1]gazizov782@gmail.com, [2]mak.bel.2002@mail.ru

***Abstract***

In this study, we present a method for creating a synthetic dataset using the MetaHuman framework to optimize the skinning of 3D models. The research focuses on improving the quality of skeletal deformation (skinning) by leveraging a diverse ar-ray of high-fidelity virtual human models. Using MetaHuman, we generated an exten-sive dataset comprising dozens of virtual characters with varied anthropometric fea-tures and precisely defined skinning weight parameters. This data was used to train an algorithm that optimizes the distribution of skinning weights between bones and the character mesh.

The proposed approach automates the weight rigging process, significantly reducing manual effort for riggers and increasing the accuracy of deformations during animation. Experimental results show that leveraging synthetic data reduces skinning errors and produces smoother character movements compared to traditional methods. The outcomes have direct applications in the video game, animation, virtual reality, and simulation industries, where rapid and high-quality rigging of numerous characters is required. The method can be integrated into existing graphics engines and development pipelines (such as Unreal Engine or Unity) as a plugin or tool, facilitating the adoption of this technology in practical projects.

***Keywords****: synthetic dataset, Metahuman, neural networks, 3D model skinning, computer ani-mation, machine learning.*

**REFERENCES**

1. *Gazizov R.R., Shubin A.V.* Procedural methods for skinning humanoid characters // Russian Digital Libraries Journal. 2022. Vol. 25. No. 5. P. 404–440.

https://doi.org/10.26907/1562-5419-2022-25-5-404-440 (In Russian).

2. Epic Games. New release brings Mesh to MetaHuman to Unreal Engine… MetaHuman Mesh to MetaHuman Announcement. 2022. URL: https://www.unrealengine.com/en-US/blog/new-release-brings-mesh-to-metahuman-to-unreal-engine-and-much-more

3. Epic Games Forums. Skinning Method in MetaHuman (developer reply), April, 2021 // Unreal Engine forum. 2021.
URL: https://forums.unrealengine.com/t/skinning-method-in-metahuman/226222

*4. Kugurakova V.V., Abramov V.D., Kostyuk D.I., Sharaeva R.A., Gazizov R.R., Khafizov M.R.* Generation of three-dimensional synthetic datasets // Russian Digital Libraries Journal. 2021. Vol. 24. No. 4. P. 622–652.
https://doi.org/10.26907/1562-5419-2021-24-4-622-652 (In Russian).

5. *Suleymanova E.A., Gazizov R.R., Kugurakova V.V.* From idea to reality: the process of clothing design for a computer game character //Izvestiya Vysshikh Uchebnykh Zavedenii, Seriya Teknologiya Tekstil'noi Promyshlennosti. 2024. Is. 5-413. P. 13–19. https://doi.org/10.47367/0021-3497_2024_5_13 (In Russian).

6. *Baran I., Popović J.* Automatic Rigging and Animation of 3D Characters. ACM Trans. Graph. 2007. Vol. 26. No. 3. P. 72-es.

*7. McGrane C.* Auto-rigging? Still something only Mixamo can do? // ThreeJS Forum. 2022. URL: https://discourse.threejs.org/t/auto-rigging-still-some thing-only-mixamo-can-do/43709/8

8. Automatic rigging of three dimensional characters for animation: пат. США № 11,170,558 B2 / Adobe Inc.; опубл. 09.11.2021.
URL: https://patents.google.com/patent/US11170558B2/en

9. *Xu Z., Zhou Y., Kalogerakis E., Landreth C., Singh K.* RigNet: Neural Rigging for Articulated Characters //arXiv preprint arXiv:2005.00559.2020.

10. *Liu L., Zheng Y., Tang D., Yuan Y., Fan C., Zhou K.* Neuroskinning: Automatic skin binding for production characters with deep graph networks // ACM Transactions on Graphics (ToG). 2019. Vol. 38. No. 4. P. 1–12.

11. *Mosella-Montoro A., Ruiz-Hidalgo J.* Skinningnet: Two-stream graph convolutional neural network for skinning prediction of synthetic characters // Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2022. P. 18593–

18602.

12. *Guo Z. et al.* Make-It-Animatable: An Efficient Framework for Authoring Animation-Ready 3D Characters //arXiv preprint arXiv:2411.18197. 2024.
URL: https://jasongzy.github.io/Make-It-Animatable/

13. *Gazizov R., Shubin A.* Modification of Skeletal Character Animation Using Inverse Kinematics Controllers // 2024 International Russian Smart Industry Conference (SmartIndustryCon). IEEE, 2024. P. 553–557.
https://doi.org/10.1109/SmartIndustryCon61328.2024.10515984

14. *Tarasov A.S., Kugurakova V.V.* Reconstruction of a three-dimensional human model from a single image // Russian Digital Libraries Journal. 2021. Vol. 24. No. 3. P. 485–504. https://doi.org/10.26907/1562-5419-2021-24-3-485-504 (In Russian).

## СВЕДЕНИЯ ОБ АВТОРАХ

***ГАЗИЗОВ Рим Радикович*** – старший преподаватель ИТИС КФУ. Сфера научных интересов: автоматизация процесса скиннинга с использованием машинного обучения.

***Rim Radikovich GAZIZOV*** – Senior Lecturer at the Institute of Information Technologies and Intelligent Systems of Kazan Federal University. Area of research interests: automation of skinning process using machine learning.

email: gazizov782@gmail.com
ORCID: 0000-0002-8349-264X

***БЕЛОВ Макар Дмитриевич*** – студент ИТИС КФУ. Сфера научных интересов: автоматизация рутинных процессов 3D-моделирования с использованием нейронных сетей.

***Makar Dmitrievich BELOV*** – Student at the Institute of Information Technologies and Intelligent Systems of Kazan Federal University. Area of research interests: automation of routine 3D modeling processes using neural networks.

email: mak.bel.2002@mail.ru
ORCID: 0009-0008-7680-1839

*Материал поступил в редакцию 31 января 2025 года*

УДК 004.928+004.85+004.93

# КЛЮЧЕВЫЕ АСПЕКТЫ РАЗРАБОТКИ ДОКУМЕНТА ИГРОВОГО ДИЗАЙНА ДЛЯ МНОГОПОЛЬЗОВАТЕЛЬСКИХ ИГР В ВИРТУАЛЬНОЙ РЕАЛЬНОСТИ

**Ю. А. Карпеева[1] [0009-0005-1632-2914], М. Р. Хафизов[2] [0000-0001-7275-9102]**

[1, 2]*Казанский федеральный университет, Институт информационных технологий и интеллектуальных систем*

[1]leyaleit@gmail.com, [2]murkorp@gmail.com

***Аннотация***

Рассмотрены уникальные аспекты разработки игрового дизайн-документа для массовой многопользовательской онлайновой ролевой игры в виртуальной реальности (VR). Обсуждены актуальность и проблемы проектирования таких игр, описаны существующие подходы к созданию и адаптации документации под VR, а также определены такие ключевые вопросы в VR, как сетевое взаимодействие, погружение, взаимодействие игроков и неигровых персонажей, управляемые искусственным интеллектом. Предложен методологический подход к построению игрового дизайн-документа для игр этого жанра на примере технической спецификации нескольких игр с акцентом на структурирование и стандартизацию документа. Представлены будущие направления разработок, включая формальные описания игровой логики для переносимости сцен, перспективы автоматизированного портирования VR и инструменты для автоматической генерации. В заключении подчеркнута важность стандартизации технических спецификаций и предложены направления для дальнейших исследований.

***Ключевые слова****: GDD, VR, MMORPG, игровой дизайн, иммерсивность, виртуальная реальность, автоматизация разработки, сетевые технологии, UX, стандартизация документации.*

## ВВЕДЕНИЕ

Виртуальная реальность стремительно развивается и привлекает все больше пользователей. По данным исследований, VR-технологии опробовали

уже около 25% взрослых в США [1]. На этом фоне жанр MMORPG[1], традиционно обеспечивающий многопользовательские «постоянно живущие» (persistent[2]) миры, приобретает новый потенциал благодаря VR, обещая невиданную степень погружения. Появляются первые проекты VR-MMORPG (например, *Zenith: The Last City*[3]), ставящие целью создать *«большой мир, в котором тысячи игроков на одном шарде*[4]*»* без компромиссов по части иммерсивности. Однако разработка полномасштабных VR MMO – это фактически «неизведанная территория» для индустрии [2], что ведет к рискам при разработке.

Проблематика разработки GDD (Game Design Document – документ игрового дизайна) для VR-MMORPG обусловлена объединением двух сложных областей: VR-технологий и MMO-игр. С одной стороны, VR предъявляет особые требования к дизайну (иммерсивность, комфорт, интерфейсы), с другой – специфика MMO-игр добавляет сложности сетевой архитектуры, баланса для множества игроков и управления обширным игровым миром. Разработчикам необходимо заранее решить, как представить в одном дизайн-документе все аспекты VR-механик (например, физические жесты, обзор 360°) и MMO-систем (прокачка, экономика, взаимодействие тысяч игроков). Без тщательного GDD велика вероятность разногласий в команде и срыва целостности видения проекта. Грамотно составленный GDD служит «дорожной картой» для команды, позволяя достигнуть требуемого результата с минимальными отклонениями.

---

[1] MMORPG (сокр. от англ. Massively Multiplayer Online Role-Playing Game) – массовая многопользовательская онлайновая ролевая игра.

[2] Persistent-мир – это виртуальная среда, которая продолжает существовать и развиваться независимо от действий или присутствия игроков. События и изменения в таком мире происходят в реальном времени, даже когда игроки находятся оффлайн. Это означает, что игровой мир динамичен: экономика, экосистема, политические альянсы и другие аспекты могут изменяться без непосредственного участия конкретного игрока.

[3] Все упоминаемые в статье игры будут описаны в разделе Лудография.

[4] В контексте MMORPG термин «шард» (от англ. shard – «осколок») обозначает отдельный сервер, или экземпляр игрового мира, на котором взаимодействуют игроки. Каждый шард представляет собой независимую копию вселенной игры, позволяя распределять нагрузку и управлять большим количеством пользователей. Термин «шард» впервые был использован в игре Ultima Online. Согласно предыстории игры, мир был заключен в «Кристалл Бессмертия», который затем был разбит на осколки (shards). Каждый осколок содержал свою версию мира, что послужило объяснением наличия нескольких серверов для игроков.

В академическом контексте тема носит фундаментальный характер, так как лежит на пересечении гейм-дизайна, человеко-компьютерных взаимодействий (human-computer interaction, HCI) и сетевых технологий. Она связана с текущими исследованиями в областях UX[5] для VR-систем, методов документирования дизайна игр и инструментов автоматизации проектирования [3].

Таким образом, основные задачи исследования включают: (1) анализ существующих подходов к созданию GDD и их применимость в VR; (2) выявление специфических проблем при разработке VR-MMORPG и поиск решений; (3) разработку методологии построения GDD на основе частного технического задания; (4) обзор перспектив автоматизации и стандартизации в данной сфере.

## ОБЗОР СУЩЕСТВУЮЩИХ ПОДХОДОВ К СОЗДАНИЮ GDD

Дизайн-документ игры – это детальное описание концепции и устройства разрабатываемой игры, служащее ориентиром для всей команды.

Классический GDD включает сведения о ключевых аспектах проекта: жанр и целевая аудитория, основные механики и игровой процесс (геймплей[6]), сюжет и персонажи, уровни и мир, визуальный стиль и интерфейс, технические требования и т. д.

Целью дизайн-документа является однозначно зафиксировать все требования к игре и способы их реализации, чтобы у всех членов команды разработки было единое понимание задачи.

GDD обычно создается на этапе предварительной проработки концепта и затем постоянно дополняется и изменяется в ходе производства – это «живой» документ, эволюционирующий вместе с проектом.

В индустрии отсутствует строгий стандарт оформления GDD: каждая студия использует удобный для себя формат. Например, документ может вестись в тек-

[5] UX (User eXperience) – пользовательский опыт, ощущения от использования приложения.

[6] Геймплей (от англ. gameplay) – совокупность игровых элементов, правил, механик и способов взаимодействия игрока с игровым миром, определяющая непосредственный процесс игры и формирующая игровой опыт.

стовом редакторе, в виде презентации с рисунками и диаграммами либо в онлайн-системе (например, Confluence[7]). Важно, что дизайн-документ нередко выступает частью соглашения между издателем и разработчиком, определяя пределы того, какой должна быть игра. Соблюдение утвержденного GDD – залог того, что конечный продукт будет соответствовать изначально согласованной концепции

**Особенности адаптации GDD под VR-игры**

Появление VR потребовало пересмотра устоявшихся подходов к гейм-дизайну и, соответственно, содержанию GDD.

Многие элементы дизайна, привычные для классических, не VR игр, плохо воспринимаются игроками в рамках виртуальной реальности, поэтому их приходится переосмыслять или изобретать заново. Например, обычные способы представления интерфейса (HUD[8], мини-карта, всплывающие окна) в VR нарушают погружение и могут вызывать дискомфорт. Интерфейсы в VR-игре должны органично вписываться в окружающий мир – «являться частью мира игры, не разрушая погружение». Разработчики адаптируют UI[9] под VR: делают его диегетическим[10], например, размещая элементы интерфейса в качестве части виртуальной

[7] Confluence – https://www.atlassian.com/software/confluence — корпоративная система для совместного управления знаниями и документацией, разработанная компанией Atlassian, предоставляющая функционал для создания, организации и совместного редактирования контента с возможностями интеграции с другими инструментами разработки.

[8] HUD (от англ. Heads-Up Display – «индикатор на лобовом стекле») в контексте видеоигр — это часть визуального интерфейса, отображающая важную информацию поверх игрового процесса. Элементы HUD предоставляют игроку данные, необходимые для эффективного взаимодействия с игрой, без необходимости прерывать игровой процесс.

[9] UI (от англ. User Interface) – это совокупность визуальных, аудиальных и тактильных средств, с помощью которых пользователь взаимодействует с цифровым продуктом.

[10] Термин «диегетический» происходит от греческого слова *διήγησις* (диегезис), что означает «повествование». В контексте видеоигр и других медиа он используется для описания элементов, которые являются частью внутреннего мира произведения и воспринимаются его персонажами.

сцены – так, в игре *Into the Radius* показатели здоровья и выносливости отображаются на наручном браслете персонажа вместо привычных полос на экране. Дизайн-документ VR-игры должен отражать эти особенности, описывая новые подходы к интерфейсу и взаимодействию.

Кроме интерфейса, основные механики требуют адаптации: движение игрока, взаимодействие с предметами, камера – все реализуется иначе. Классические кат-сцены[11] и заскриптованные события, завязанные на фиксированной камере, трудно применимы: свободный взгляд от первого лица и 360°-сцена означают, что пролеты камеры или резкая смена ракурса вызовут морскую болезнь у игрока. Поэтому GDD для VR-игры должен включать разделы, описывающие механику передвижения (например, телепортация вместо свободной ходьбы), меры против укачивания, систему обзора и т. д.

Кроме того, совместно с технологией VR часто используются новые устройства ввода – контроллеры движения, отслеживание рук, голосовой ввод, которые тоже нужно учесть в документе игрового дизайна. Таким образом, при создании GDD VR-игры к традиционному перечню разделов добавляются новые: модели управления взглядом и движением, схемы VR-контроллеров, настройки комфорта (границы игрового пространства, режимы телепортации, виньетки при движении и пр.), требования к оборудованию (HMD[12], трекинг) и пр. Все эти нюансы должны быть описаны, чтобы команда с самого начала закладывала правильные решения под VR.

[11] Кат-сцена (от англ. cutscene – «вырезанная сцена») – это эпизод в видеоигре, во время которого управление персонажем частично или полностью передаётся игре, а игрок становится наблюдателем. Основная цель кат-сцен – продвижение сюжета, демонстрация важных событий или развитие персонажей без непосредственного участия игрока.

[12] HMD (Head-Mounted Display) — это устройство, надеваемое на голову, оснащенное дисплеями и акустической системой, предназначенное для погружения пользователя в виртуальную реальность. Оно обеспечивает отображение объёмного изображения и звука, создавая эффект присутствия в виртуальной среде. Современные HMD, такие как шлемы виртуальной реальности, оснащены датчиками отслеживания движений головы, что позволяет пользователю взаимодействовать с виртуальным миром естественным образом.

**Влияние VR на игровые механики и UX-дизайн**

VR коренным образом изменяет пользовательский опыт в игре, что отражается на механиках. В дизайне появляется принцип «иммерсивность прежде всего». Пользователь буквально находится внутри виртуального мира, поэтому любые неестественные элементы интерфейса или механики ощутимо выбивают из роли. GDD VR-проекта должен предусматривать, как сохранить баланс между удобством игрока и реалистичностью происходящего.

Например, в обычных MMO принято показывать всплывающие имена персонажей, окна чата, множественные панели с навыками. В VR такие элементы, плавающие перед глазами, разрушают эффект присутствия. Решение – интегрировать их в окружающую среду или использовать альтернативы. В дизайн-документе рекомендуется описывать диалоговую систему через голосовой чат вместо текстового и указывать, как игроки будут получать информацию об окружающих (например, за счет реалистичных жестов или диалогов с NPC[13] вместо текстовых подсказок).

Взаимодействие с объектами в VR становится физическим – игрок не кликает кнопкой «взять предмет», а протягивает виртуальную руку и нажимает триггер контроллера, имитируя хватание [4]. Это более интуитивно, но требует проработки множества сценариев: что если объект выскользнул, как реагируют физические свойства, ощутит ли игрок вибрацию при взаимодействии и т. д. Современные разработки в области устройств с тактильной обратной связью, таких как цифровые перчатки для VR, могут значительно усилить этот эффект, обеспечивая более естественное и интуитивное взаимодействие с виртуальными объектами [16].

UX-дизайн в VR также уделяет внимание эргономике: продолжительные однообразные действия могут вызвать усталость в реальном мире, а резкие движения камеры – дискомфорт. Поэтому механики (например, лазание, бой, перемещение) нужно проектировать с учетом физических возможностей игрока. GDD

[13] NPC (от англ. Non-Player Character — «неигровой персонаж») — это персонаж в играх, которым не управляет игрок. В видеоиграх поведение NPC определяется программно, а в настольных ролевых играх — ведущим или мастером. NPC могут быть дружественными, нейтральными или враждебными по отношению к игроку.

должен фиксировать ограничения по продолжительности сессии, предусматривать опции комфорта (например, настройки чувствительности, калибровки под рост игрока и т. п.).

В целом VR повышает требования к качеству проработки деталей: если в обычной игре благодаря специфике камеры видимый объект может быть упрощен, то в VR игрок может подойти вплотную и рассмотреть его со всех сторон, заглянуть под любой угол. Это влияет на графический дизайн и оптимизацию – GDD должен устанавливать баланс между детализацией и производительностью для поддержания 90+ FPS[14], необходимых для избежания морской болезни (или motion sickness – укачивание).

Таким образом, отметим, что существующие подходы к GDD расширяются: к традиционным разделам добавляется описание VR-специфичных UX-решений, а старые дополняются требованиями иммерсивности. Этот симбиоз обеспечивает основу для разработки VR-MMORPG, учитывающую особенности среды VR.

## ПРОБЛЕМЫ РАЗРАБОТКИ GDD ДЛЯ VR-MMORPG

### Сетевые технологии и их влияние на дизайн игры

Многопользовательский характер MMORPG накладывает тяжелые сетевые требования, которые в VR-контексте еще более критичны. Низкая задержка передачи данных сети – один из ключевых факторов, от которого зависит комфорт игроков. Даже небольшие задержки в VR заметно нарушают синхронность и погружение: если один игрок поворачивает голову или машет рукой, а другие видят это движение с опозданием, возникает эффект рассинхронизации и «дёрганости» взаимодействия [5]. В MMO, где игроки должны действовать согласованно (например, совместно сражаться с врагом), небольшой лаг[15] в несколько сотен

[14] FPS (от англ. frame per second) – показатель, определяющий количество кадров, отображаемых на экране в течение одной секунды. Чем выше значение FPS, тем плавнее и отзывчивее выглядит игровой процесс. Оптимальным считается показатель в 60 FPS, обеспечивающий комфортную игру. Низкий FPS (менее 30) может приводить к «лагам» и ухудшению впечатления от игры.

[15] Лаг (от англ. lag – «запаздывание», «задержка») – задержка в работе компьютерного приложения, когда оно не реагирует на пользовательский ввод вовремя.

миллисекунд может привести к серьезному диссонансу в восприятии и фрустрации – персонажи будут «телепортироваться» или реагировать несвоевременно.

Дизайн-документ VR-MMORPG обязан учитывать эти ограничения: в разделе сетевой архитектуры описываются целевые метрики задержки (например, не более 50 мс для локальных игроков), а также стратегии компенсации лагов.

Одно из распространенных решений – это предсказание на стороне клиента и сглаживание: клиентская часть игры пытается предугадать движение других игроков и объектов на основе последних данных, не дожидаясь новых пакетов от сервера. Например, при быстром движении аватара система интерполирует траекторию, а при получении точных данных корректирует позицию плавно, чтобы игроки не заметили скачка. GDD должен описывать, какие именно механизмы предсказания и интерполяции используются и как они влияют на геймплей (возможны ли расхождения, «фантомы» и т. п.).

Кроме того, важно указать модель сетевого обмена: централизованный сервер, пиринговая сеть или гибрид – от этого зависит масштабируемость MMO. В VR-MMORPG используется клиент-серверная модель с мощными серверами (шардами), обслуживающими зоны мира. В GDD необходимо детализировать максимальное число игроков на сервер, сценарии поведения при перегрузке, а также и как игра распределяет пользователей по шардам.

Кроме того, необходимо учитывать такой вопрос доступа, как кроссплатформенность. Если планируется поддержка различных VR-устройств (ПК VR, standalone[16]-шлемы или классическая версия без VR), это влечет поддержку разных протоколов или версий клиента. Например, как отмечается в отрасли, разнообразие VR-девайсов (HMD разных производителей) усложняет единый мультиплеерный опыт [1], поэтому внедрение стандартов (таких как OpenXR) или отдельных серверов для разных платформ должно быть отражено в дизайне игры.

[16] Standalone-гарнитура (или «шлем») – автономное устройство виртуальной или дополненной реальности, функционирующее без подключения к внешним компьютерам или консолям, со встроенными процессором, памятью и аккумулятором для полностью независимой работы.

**Баланс многопользовательского опыта и иммерсивности**

Иммерсивность VR иногда конфликтует с удобствами привычных MMO. Задача гейм-дизайна – добиться, чтобы социальные и игровые функции MMO не «ломали» погружение.

В GDD требуется детально проработать, какие компромиссы допускаются, например общение между игроками: текстовый чат в VR неудобен (отрывает от игрового процесса и требует виртуальной клавиатуры), поэтому упор делается на голосовой чат и жесты. Однако голосовой чат может мешать иммерсивности, если подается «из ниоткуда»; решением является позиционированный 3D-звук, когда голос другого игрока слышен со стороны его аватара в виртуальном пространстве. В дизайн-документе следует указать, что голосовая связь реализована с эффектом присутствия (как будто звук идет от местоположения персонажа), а также предусмотреть альтернативы для случаев, когда голосовой канал недоступен (систему контекстных жестов или эмотиконов[17], встроенных в мир игры).

Кроме того, требуется баланс между интерфейсами групп, гильдий, инвентаря. В MMO проблема избыточной информации на экране обычно решается через панели UI. В VR вместо этого применяются *встроенные элементы интерфейса*: например, инвентарь может быть реализован как визуально отображаемый рюкзак или меню, появляющееся на виртуальном планшете в руках персонажа. Такой подход был успешно реализован: когда интерфейс становится объектом внутри игры, он меньше отвлекает и воспринимается естественнее. GDD должен включать описание диегетических интерфейсов – сценарии взаимодействия игрока с меню (например, взять предмет из рюкзака движением руки, настроить параметры на виртуальном планшете, и т. д.), такие естественные взаимодействия значительно повышают погружение и реалистичность пользовательского опыта [17].

*Баланс сложности и доступности* – еще один важный момент. MMO славятся глубиной механик (различные классы, системы прогрессии, экономики), но

[17] Эмотиконы (от англ. emoticon: emotion – «эмоция» и icon – «иконка») – это комбинации символов, используемые в текстовых сообщениях для выражения эмоций, настроений или действий. Они помогают передать эмоциональный контекст, который может быть утрачен в письменной речи.

в VR игроку физически труднее управлять очень сложным интерфейсом. Поэтому некоторые системы могут быть упрощены для VR. Например, крафтинг: вместо многокомпонентных рецептов с перетаскиванием десятков предметов UI в VR это можно преподнести как интуитивный *«физический»* процесс – игрок комбинирует объекты на верстаке. Дизайн-документ описывает такие изменения механик, обосновывая их необходимостью поддерживать погружение без ущерба для игрового опыта.

Таким образом, в настоящем разделе указано, какие классические элементы MMO сохранены, какие адаптированы под VR и каким образом обеспечен полноценный MMO-функционал (взаимодействие, конкуренция, кооперация), не ощущая при этом диссонанса между реальностью и игрой.

**Взаимодействие игроков и управление игровым миром**

В VR-MMORPG особое внимание уделяется тому, как игроки будут взаимодействовать друг с другом и с окружением. В традиционных MMO взаимодействие во многом опосредовано недиегетическим интерфейсом (нажатием клавиш для торговли, диалоговых окон для общения, иконок для дружбы, атаки и т. п.).

В VR взаимодействия стремятся сделать естественными: социальные жесты, разговор «лицом к лицу», совместные физические действия, а GDD должен описывать правила этих взаимодействий. Например, система жестов: можно предусмотреть жест рукопожатия двумя игроками, который игра распознает и активирует, скажем, меню обмена предметами (симулируя реальное рукопожатие как начало торговли). Или жест поднятой руки, распознаваемый как приветствие.

Такие механики повышают ощущение присутствия в мире. Но возникает и новая проблема – безопасность и этикет в VR. Аватары игроков могут подходить очень близко друг к другу, что иногда приводит к чувству дискомфорта или домогательств. Индустрия уже выработала подходы для защиты игрока, и это тоже часть дизайна: вводятся «личные зоны» и другие контроли безопасности. Например, множество VR-приложений внедрило функции вроде виртуального «личного пузыря», при активации которого другие аватары не могут приблизиться вплотную или пересечь модель игрока [6].

В GDD MMO необходимо указать, есть ли у персонажей персональные границы, как реализована коллизия между игроками (можно ли проходить сквозь друг друга или нет), какие есть средства модерации (вплоть до жалоб на агрессивных игроков). Управление разработчиками игровым миром тоже осложняется: постоянно активный виртуальный мир требует мониторинга и обновлений.

В GDD можно описать план будущих мероприятий, например регулярные ивенты, возникающие в мире, и их презентация (диегетично, через виртуальных глашатаев и др.), механизмы обновления контента без отключения сервера и т. д.

Нужно также учитывать вопрос, как игроки влияют на мир. MMO часто ограничивают взаимодействие с окружением (чтобы предотвратить хаос), но VR-завязанные задачи, например совместно строить что-то в виртуальном пространстве или решать головоломки физически, могут стать частью геймплея.

Следовательно, GDD должен установить границы манипулирования миром: какие объекты интерактивны, сохраняются ли изменения, есть ли физика между игроками (например, возможность толкать друг друга или соревноваться в перетягивании каната).

Каждая такая механика несет сетевую нагрузку и гейм-дизайнерские последствия (вопрос баланса и появления эксплойтов[18]). Все ключевые решения (например, включить или отключить урон по союзникам) должны быть зафиксированы в документе, чтобы у разработчиков и гейм-дизайнеров было чёткое руководство к действию.

В итоге этот подраздел GDD описывает социальный геймплей VR-MMORPG: от дружеских взаимодействий и формирования сообществ до конфликтов и способов их разрешения внутри виртуального мира.

**Применение AI-помощников и NPC в VR**

Персонажи, неуправляемые игроками (NPC), и различные AI-помощники

[18] Эксплойт (англ. exploit – эксплуатировать) – использование ошибки или недоработки таким образом, чтобы получить существенное несправедливое преимущество для использующих его игроков.

традиционно играют важную роль в MMO – они населяют мир, дают задания, торгуют, сопровождают игрока. В VR их роль может быть еще более значимой, поскольку реалистичное поведение NPC усиливает чувство присутствия. Новейшие достижения в области искусственного интеллекта позволяют создавать «умных» виртуальных персонажей, способных поддерживать диалог и реагировать на действия пользователя практически как живые собеседники [7].

GDD для VR-MMORPG должен рассмотреть, как используется AI для оживления мира. Например, можно описать AI-напарника – виртуального ассистента, сопровождающего игрока (скажем, в виде питомца-дрона или персонажа-наставника). Такой помощник мог бы объяснять интерфейс, напоминать о заданиях, реагировать на голосовые команды. Если планируется подобная функция, в GDD указываются возможности AI: распознавание речи игрока, степень понимания контекста, набор стандартных реплик или даже генеративная модель диалога. Интерес представляет использование генеративного AI для NPC-диалогов: вместо фиксированных фраз NPC могут динамически генерировать ответы, делая общение уникальным. Уже существуют системы, позволяющие NPC общаться естественным языком, воспринимать окружение и выполнять сложные действия [8].

В рамках MMORPG это могло бы означать более богатый PvE[19]-контент: NPC-герои реагируют на поведение игрока, враги координируются разумно. GDD должен оговорить, будет ли применяться ИИ такого рода и как сохранить при этом сюжетную составляющую.

Кроме того, VR позволяет сделать NPC интерактивными на уровне жестов: торговец-NPC может отреагировать на кивок или помахивание рукой игрока. Такие детали желательно включить в описание поведения NPC.

Отдельно стоит раздел об AI-дирижере (AI Dungeon Master) – невидимом руководителе событий, который подстраивает сложность под группу игроков, генерирует динамические события. Это тоже набирающий популярность подход, и в VR он может повысить непредсказуемость мира [9].

Если в игре предусмотрено наличие AI-дирижера, то GDD должен подробно

[19] PvE (сокр. англ. Player versus Environment – Игрок против Мира) – жанр игр, когда игрок противостоит NPC.

описывать его алгоритмы адаптации сложности: какие параметры ИИ отслеживает (уровень навыков игроков, их стиль игры, скорость прогресса) и в каком диапазоне он может вмешиваться в игровой процесс. Это включает настройку частоты динамических событий, балансировку вызовов в зависимости от действий игроков и управление сложностью таким образом, чтобы игра оставалась интересной, но не чрезмерно сложной или, наоборот, слишком простой.

Наконец, отметим, как AI участвует в управлении MMO: возможно применение алгоритмов машинного обучения для поиска «читеров» (от англ. cheat – мошенничество), оптимизации сетевого кода (предсказания движения, как упоминалось ранее) и даже модерации сообщества (автоматическое выявление оскорблений по голосовому чату). Все эти применения желательно перечислить, по крайней мере, в виде возможности, чтобы команда имела представление, где AI может снизить нагрузку на людей.

Таким образом, включение современных AI-технологий обещает более динамичный и живой VR-мир, и их описание в GDD обеспечивает планирование необходимых ресурсов (вычислительных мощностей, интеграции соответствующих SDK[20]) на ранней стадии. Согласно отраслевым обзорам, внедрение AI-персонажей заметно обогащает интерактивность и персонализацию опыта игроков, поэтому VR-MMORPG, использующая такие технологии, может достичь нового уровня вовлечения аудитории.

## АНАЛИЗ СУЩЕСТВУЮЩИХ ШАБЛОНОВ GDD

Классические шаблоны GDD широко используются в индустрии и были опубликованы на таких ресурсах, как GameDev.net [10] и Gamasutra [11]. Они включают следующие основные разделы:

- ***Концепция игры*** (жанр, целевая аудитория, ключевые механики).
- ***Игровой процесс и механики*** (геймплейные элементы, прогрессия, правила).

[20] SDK (сокр. англ. Software Development Kit) – набор инструментов разработки ПО, включающий библиотеки, документацию, примеры кода и утилиты, предоставляемые разработчикам для создания приложений на определенной платформе или с использованием конкретной технологии.

- ***Описание мира и уровней*** (структура игрового пространства, NPC, окружение).
- ***Пользовательский интерфейс*** (HUD, меню, системы инвентаря).
- ***Аудиовизуальные компоненты*** (графический стиль, анимация, музыка).

Эти структуры удобны, но не охватывают специфики VR. Например, традиционные GDD не содержат информации о *motion sickness*, системах перемещения в VR, тактильных взаимодействиях и эргономике. В VR важны не только визуальные элементы, но и реалистичность ощущений, что требует отдельного подхода к документированию.

Перечислим ограничения традиционных GDD в VR.

1. Отсутствие разделов про VR UX – традиционные GDD не учитывают влияние на комфорт игрока, способы устранения укачивания и адаптацию интерфейса для VR.
2. Локомоция и механики передвижения – VR требует особых решений для передвижения (телепортаций, плавного движения, трекинга тела), которые должны быть задокументированы.
3. Иммерсивные интерфейсы – стандартные HUD и меню не подходят для VR, требуется диегетическое отображение интерфейса.
4. Физическое взаимодействие – в VR игроки взаимодействуют с объектами напрямую (хватают, кидают, крутят), что должно быть прописано в GDD.

## АНАЛИЗ СПЕЦИФИКИ УСПЕШНЫХ VR-ИГР

**Half-Life: Alyx.** Компания Valve в *Half-Life: Alyx* не использовала традиционный GDD, а применила *итеративный* процесс, тестируя каждую игровую механику в реальном времени [12]. Конечно, крупная компания не станет выкладывать GDD своей игры в открытый доступ, поэтому невозможно точно утверждать о содержании документации видеоигры. При этом до сих пор развиваютигровой процесс относительно специфики виртуальной реальности и удобного UX:

- Гравитационные перчатки – VR-альтернатива подбора предметов с помощью жестов. Такое решение не только обосновано сюжетом видеоигры, но и является более удобным для игрока, избавляя от необходимости наклоняться.

- Адаптивная локомоция – три доступных варианта режима передвижения (телепорт, смещение, скольжение). Данный подход позволяет игроку самостоятельно выбрать наиболее удобный режим в зависимости от степени привыкания к VR и интересующем игровом опыте.
- Диегетический интерфейс – отказ от классических HUD[21]-элементов в пользу естественного взаимодействия и более натуральной обратной связи.

С учётом проработки и качества данных аспектов видеоигры можно с уверенностью предположить, что если бы GDD Half-Life: Alyx был опубликован, то он содержал бы подробное описание.

**VR Psychological Game.** Следующий проект представляет собой студенческую работу по разработке дизайн-документа по разработке игры в жанре психологического ужаса в виртуальной реальности [13]. Учитывая, что данный проект является незаконченной разработкой, некоторые аспекты, затрагивающие непосредственно разрабатываемую игру, отсутствуют. При этом в данном документе присутствует обзор других проектов, идеи которых автор предлагает использовать в своей игре:

- Замена руки подобранным предметом – позволяет сконцентрироваться на объекте взаимодействия без помех.
- Подстраивание перемещения под игровую зону – удобная функция в рамках ограниченной игровой зоны, позволяющая перемещаться в виртуальном пространстве, не отвлекаясь на объекты в реальности.
- Пространственные звуки, которые позволяют создать более привычную для игрока среду, не обременяя его игровыми условностями.

Хотя автор чётко не описывает наличие данных аспектов в своём проекте, их наличие в рамках разработки приложения для виртуальной реальности позволит улучшить UX от пребывания в виртуальной среде.

[21] HUD (Heads-Up Display) – графический интерфейс, отображающий важную информацию непосредственно в поле зрения пользователя без необходимости отвлекаться от основного действия, часто используемый в видеоиграх, авиации и современных технологических устройствах.

**Unzen Dungeon VR.** Данный проект [14] также представлен в виде студенческой работы по разработке VR-игры в жанре экшн[22]-приключения. Несмотря на краткость и обобщенность описания, в документе отмечаются конкретные особенности, которые специфичны именно для виртуальной реальности:

- Естественное управление жестами – необходимость, в рамках специфики используемых устройств ввода. VR-контроллеры повторяют движение кистей и отслеживают нажатие пальцев, что является взаимодействием с предметами, приближенным к реальному.
- Минималистичный интерфейс – ограничение в использовании недиегетических элементов, которые отрицательно влияют на погружение и игровой опыт.
- Возможность выбрать тип перемещения – удобная функция, позволяющая улучшить UX за счет выбора, более подходящего для конкретного игрока.
- Реалистичная графика и динамическая акустика – позволяют без затруднений погрузиться в виртуальный мир благодаря знакомой атмосфере.

**Mind The Game VR: Immersive Museum Experience.** Данный GDD [15] предоставляет очень подробное, структурированное описание VR-приложения для посещения виртуального музея. Документ для этого приложения выделяет несколько ключевых аспектов, связанных именно с особенностями виртуальной реальности:

- Безопасность и комфорт игрока
    - Детально проработаны вопросы возможного возникновения морской болезни, усталости и ограничений по физическому пространству. Предусмотрены предупреждения о необходимости игры в безопасном пространстве и советы по регулярным перерывам.
    - Включены опции по регулировке поля зрения, размера текста и использования различных методов перемещения (например, свободное перемещение и телепортация), чтобы снизить дискомфорт.

[22] Экшн (англ. action) — жанр, характеризующийся динамичным развитием сюжета, обилием напряженных событий, зрелищных сцен и физической активности героев, часто включающий элементы погонь, сражений и преодоления опасностей.

- Контроль и взаимодействие в VR
    - Игрок способен выбирать более удобный режим перемещения.
    - Сделан фокус на физическую активность, позволяющий взаимодействовать с экспонатами. Игрок может использовать VR-контроллеры для захвата, вращения и изучения объектов, что усиливает ощущение присутствия и вовлеченности.
- Интерфейс и обратная связь
    - Интерфейс минималистичен и построен таким образом, чтобы не нарушать погружение в виртуальную среду. Доступные меню и подсказки появляются только в ключевые моменты (например, при приближении к экспонату или смене режима перемещения).
    - Динамическое освещение, анимация и звуковое сопровождение (например, звуки шагов, шум толпы, атмосферные эффекты) помогают создать правдоподобную и иммерсивную атмосферу музея.

## МЕТОДОЛОГИЯ ПОСТРОЕНИЯ GDD ДЛЯ VR-MMORPG

Проведенный анализ проблематики VR-MMORPG и существующих подходов для разработки таких игр позволил выработать структуру GDD. Итак, предлагается следующая структура GDD для VR-MMORPG, учитывающая особенности как виртуальной реальности, так и многопользовательской составляющей:

1. *Базовая концепция и техническая инфраструктура*

- Концепция, целевая аудитория и уникальное торговое предложение (USP[23])
- Сетевая архитектура, модель серверов и шардов
- Требования к оборудованию (минимальные спецификации HMD) и сети
- Стратегии масштабирования при высокой нагрузке

2. *VR-специфические элементы*

- Система передвижения (локомоции) и телепортации

[23] USP (сокр. от Unique Selling Proposition или Unique Selling Point) — уникальное торговое предложение, ключевая маркетинговая концепция, которая определяет отличительные характеристики продукта или услуги, делающие их особенными на фоне конкурентов.

- Предотвращение укачивания (через виньетирование, FOV-ограничения[24])
- Физическое взаимодействие с объектами – схемы управления
- Эргономические аспекты и продолжительность игровых сессий

*3. Диегетический интерфейс и UX*

- Интеграция UI в игровой мир (вместо классического HUD)
- Пространственный звук и аудио-навигация
- Тактильная обратная связь (использование контроллеров)
- Инвентарь и меню как физические объекты внутри VR

*4. Многопользовательское взаимодействие*

- Система визуальной идентификации игроков
- Жестовая коммуникация и язык тела аватаров
- Голосовой чат с пространственным позиционированием
- Правила социального взаимодействия и модерации

*5. Интеграция AI и NPC*

- ИИ для NPC с естественным VR-взаимодействием
- Системы распознавания естественного языка и жестов
- Адаптивный AI-дирижер для подстройки сложности
- Процедурная генерация контента для поддержания интереса

*6. Контент-план и технологии реализации*

- Зонирование мира и структура локаций
- Технологии LOD[25] для VR-оптимизации
- Система квестов с учетом VR-взаимодействия
- Экономика и прогрессия персонажа

*7. Безопасность и комфорт*

- «Личные зоны» и защита от нежелательного взаимодействия
- Настройки доступности для разных физических возможностей

[24] FOV-ограничения FOV – сокр. от Field of View) – техника в VR, при которой искусственно сужается поле зрения пользователя во время движения, чтобы уменьшить нагрузку на вестибулярный аппарат.

[25] LOD (сокр. от Level of Detail) – техника оптимизации в компьютерной графике, при которой детализация 3D-модели или объекта автоматически уменьшается по мере удаления от наблюдателя, что позволяет снизить вычислительную нагрузку, сохраняя высокую детализацию только для объектов, находящихся в непосредственной близости.

- Система отчетов о нарушениях и автоматической модерации
- Родительский контроль и возрастные ограничения

Особое внимание в 7-м разделе GDD («Безопасность и комфорт») следует уделить эргономике VR-взаимодействия.

Документ должен учитывать физиологические особенности пользователей, включая оптимальные зоны досягаемости для контроллеров, комфортные углы обзора и продолжительность удержания определенных поз.

Правильно спроектированные эргономические решения позволяют значительно снизить усталость игрока и повысить комфорт длительных игровых сессий, что критически важно для удержания аудитории в MMORPG.

На рис. 1 представлены основные зоны эргономического взаимодействия пользователя в VR-пространстве. При разработке GDD важно учитывать, что игровые механики должны быть спроектированы с учетом этих зон: основные часто используемые элементы интерфейса следует размещать в зоне комфортного доступа, в то время как второстепенные элементы могут находиться в периферийных зонах. Эта схема служит визуальным руководством для разработчиков при проектировании интерфейсов и механик взаимодействия

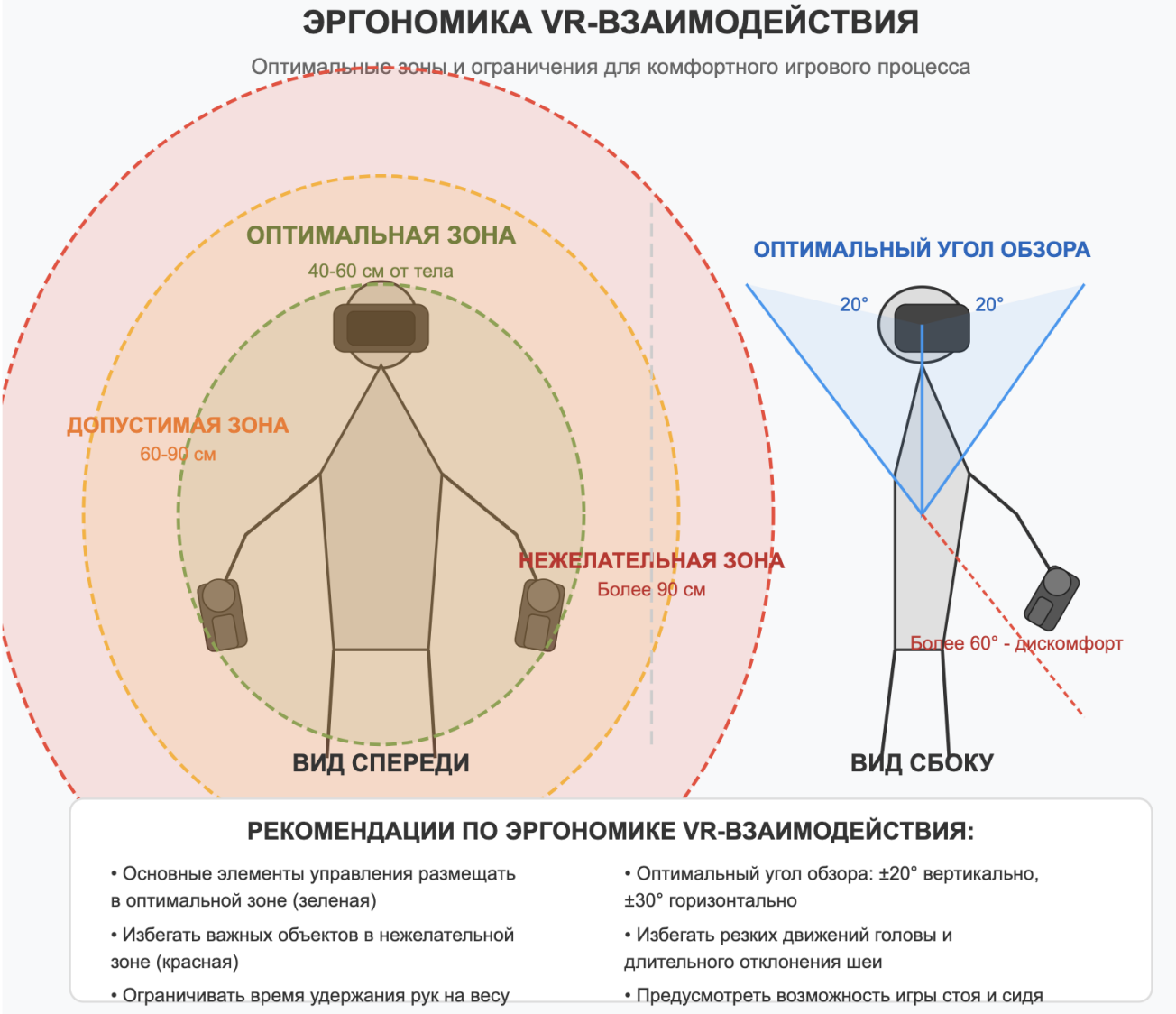



Рис. 1. Схема эргономики VR-взаимодействия

Эргономика VR-взаимодействия тесно связана с общей архитектурой GDD. Для эффективной организации всех компонентов GDD необходимо не только учитывать физические аспекты взаимодействия, но и выстраивать логическую структуру взаимосвязей между различными элементами дизайна. Для наглядного представления предлагаемой структуры GDD и взаимосвязей между её компонентами разработана схема (рис. 2). Она демонстрирует модульный подход к построению документа игрового дизайна, каждая его составляющая является одновременно самостоятельным и взаимосвязанным блоком. Такое структурирование позволяет различным специалистам команды разработки фокусироваться на своих областях ответственности, сохраняя при этом целостное видение проекта.

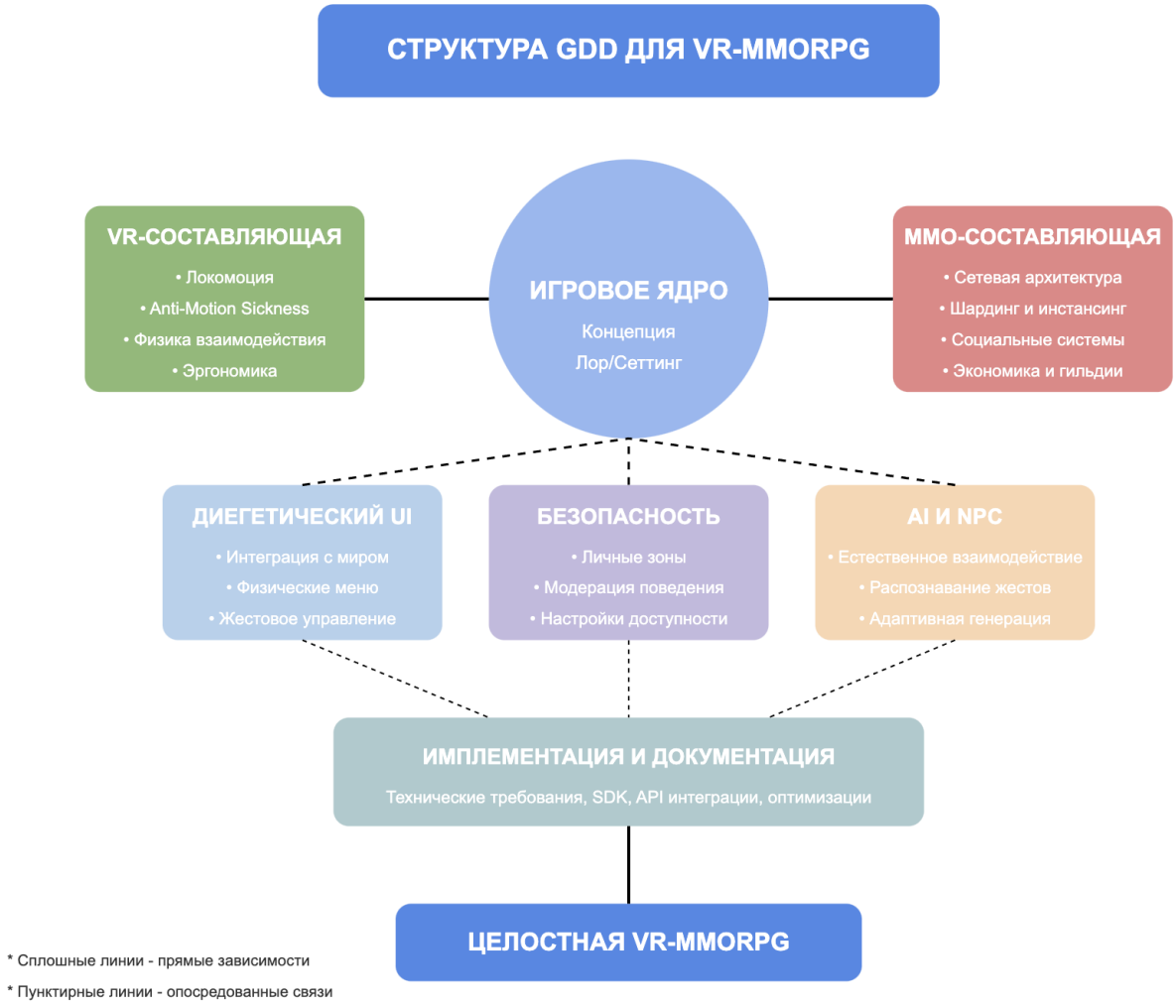



Рис. 2. Структура и взаимосвязи компонентов GDD для VR-MMORPG

При этом важно понимать, что GDD не является статичным документом, а проходит определенную эволюцию от первоначальной концепции до полной реализации игрового проекта. Динамическая природа документа игрового дизайна особенно проявляется в контексте VR-MMORPG, где пересекаются передовые технологии и постоянно развивающиеся подходы к геймплею. На начальных этапах GDD формируется как концептуальный документ, обрисовывающий основные идеи и механики; затем, по мере прототипирования, он обрастает техническими спецификациями и конкретными решениями; в процессе производства документ актуализируется на основе тестирования и обратной связи; а на завершающих этапах трансформируется в комплексное руководство для поддержки и дальнейшего развития проекта.

Каждая стадия трансформации документа сопровождается определенными дополнениями и уточнениями, отражающими текущее состояние разработки.

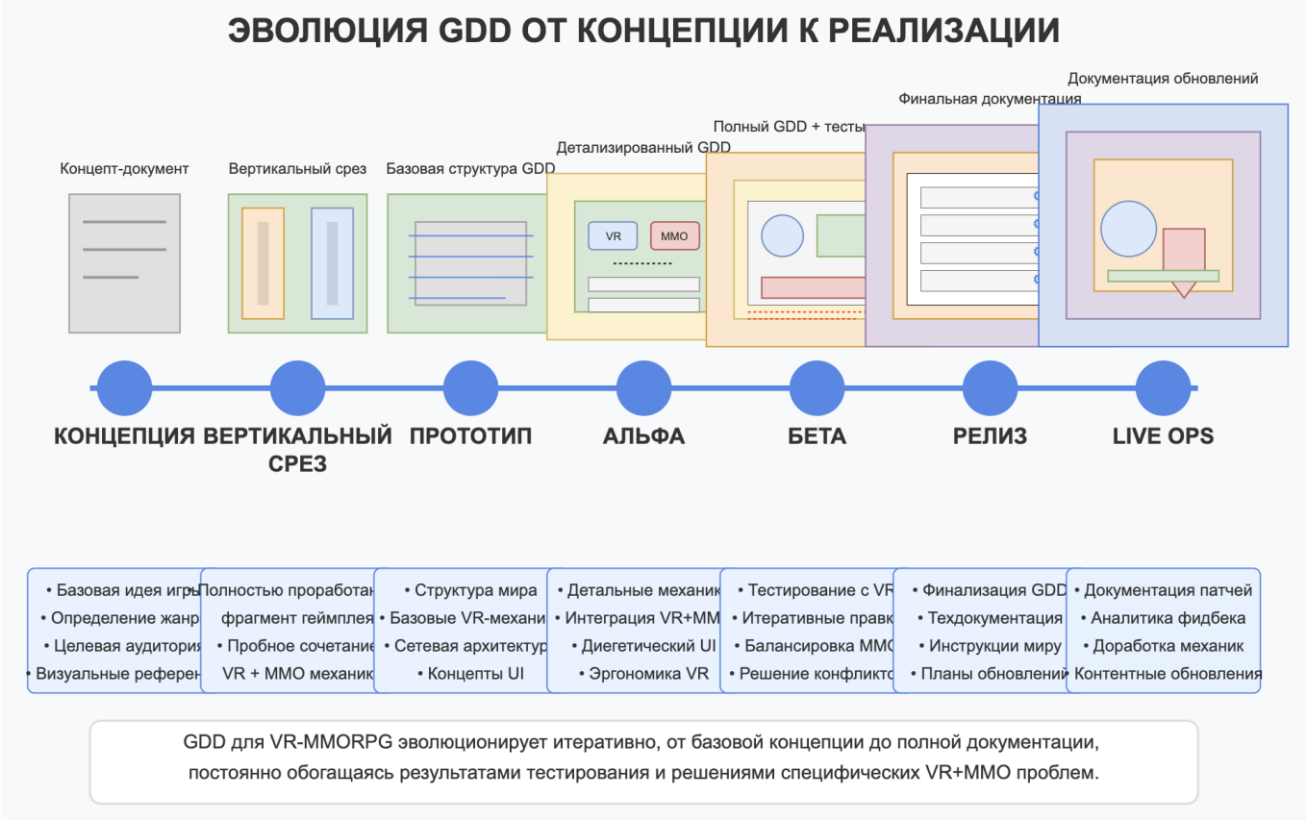



Рис. 3. Схема эргономики VR-взаимодействия

На рис. 3 представлены основные этапы трансформации документа игрового дизайна в процессе разработки VR-MMORPG.

## ОБОБЩЕННЫЕ ПРАВИЛА СОЗДАНИЯ GDD ДЛЯ VR-MMORPG

На основе проведенного исследования можно сформулировать следующие ключевые решения и рекомендации по созданию документа игрового дизайна для VR-MMORPG.

**1. Модульный подход к структуре документа.** Рекомендуется применять модульный подход к структуре GDD, где каждый аспект игры описывается в отдельном разделе, но с чёткими перекрёстными ссылками между взаимозависимыми элементами. Это позволяет различным специалистам команды (программистам, графические дизайнерам, сетевые инженерам, геймдизайнерам и т. д.)

работать с релевантными для них частями документа, сохраняя целостное видение проекта.

**2. Приоритизация комфорта и доступности.** В отличие от традиционных MMO, где имеется геймплей, VR требует противоположного подхода. Рекомендуется ввести систему приоритетов, где психофизиологический комфорт игрока (отсутствие укачивания, эргономика, доступность) должен иметь наивысший приоритет, а только потом геймплей, экономика и другие системы. В GDD должна быть выделена отдельная секция «Комфорт и доступность», предшествующая описанию механик.

**3. Интеграция нативного VR-интерфейса.** Критически важно отказаться от прямого портирования традиционных игровых интерфейсов. Вместо этого следует разработать полностью диегетический интерфейс (встроенный в мир игры), использующий физические метафоры: виртуальные браслеты на руках персонажа, голографические проекции, интерактивные объекты в окружении. Каждый элемент UI должен быть описан с точки зрения

- физического представления в мире,
- механики взаимодействия (жесты, прикосновения),
- ситуаций использования.

**4. Сетевая архитектура с приоритетом локальной синхронизации.** Рекомендуется использовать гибридную клиент-серверную архитектуру с приоритетом локальной синхронизации. В GDD следует прописать, что движения и действия ближайших к игроку объектов должны синхронизироваться с повышенной частотой и приоритетом, в то время как удалённые события могут обновляться реже. Такой подход, известный как «зоны интереса» (Areas of Interest), должен быть детально задокументирован для сетевых программистов.

**5. Социальные механики с акцентом на невербальную коммуникацию.** Необходимо уделять особое внимание невербальной коммуникации при разработке VR-MMORPG, для этого GDD должен описывать:

- систему распознавания и трансляции жестов между игроками,
- механики социального взаимодействия, основанные на физическом присутствии,

- интеграцию голосового общения с пространственным позиционированием,
- эмоциональные выражения аватаров, контролируемые естественными движениями игрока.

**6. Живой документ с инструментами визуализации.** GDD для VR-MMORPG должен быть «живым документом», постоянно обновляемым на основе тестирования и итерации, для этого можно добавлять

- прототипы VR-взаимодействий в виде коротких видео,
- интерактивные схемы ключевых геймплейных циклов,
- тепловые карты вероятности дискомфорта для различных механик,
- обновляемую метрику баланса для различных классов и активностей [18].

**7. Стратегия масштабирования контента и оптимизации.** Учитывая высокие требования VR к производительности, GDD должен включать четкую стратегию технической оптимизации и масштабирования:

- градацию визуальных эффектов для разных уровней производительности,
- правила инстансинга[26] и шардинга игрового мира для поддержки массовости,
- механизмы процедурной генерации и повторного использования контента,
- приоритизацию геймплейных элементов при ограниченных ресурсах.

**8. Интеграция с AI для динамической адаптации.** Современные VR-MMORPG могут существенно выиграть от применения технологий искусственного интеллекта, поэтому в GDD можно добавить:

- адаптацию сложности под индивидуального игрока (используя AI),

[26] Инстансинг (англ. instancing) – технология в многопользовательских играх, при которой создаются изолированные копии определенных игровых локаций или зон для отдельных игроков или групп. Это позволяет разгрузить сервера, обеспечивает приватность прохождения сюжетных миссий и особых событий, а также предотвращает перенаселение важных игровых областей, что особенно актуально для MMORPG с большим количеством одновременно активных пользователей.

- механизмы генеративного контента (диалоги NPC, квесты, события),
- системы анализа поведения игроков для выявления проблемных паттернов,
- автоматическую модерацию для обеспечения безопасной среды.

## НАШ ОПЫТ СОЗДАНИЯ GDD ДЛЯ VR-MMORPG

Предложенные подходы были апробированы при разработке мультиплеерной VR-игры *«Пиратская битва»* (рис. 4).

В ходе разработки игры был пройдён весь путь эволюции GDD от концепции до реализации. На этапе концептирования удалось уточнить важные детали, которые позволили предусмотреть при разработке многие спорные моменты, некоторые из них приведены в Приложении.

Описанные подходы дали положительные результаты со стороны заказчика и пользователей.

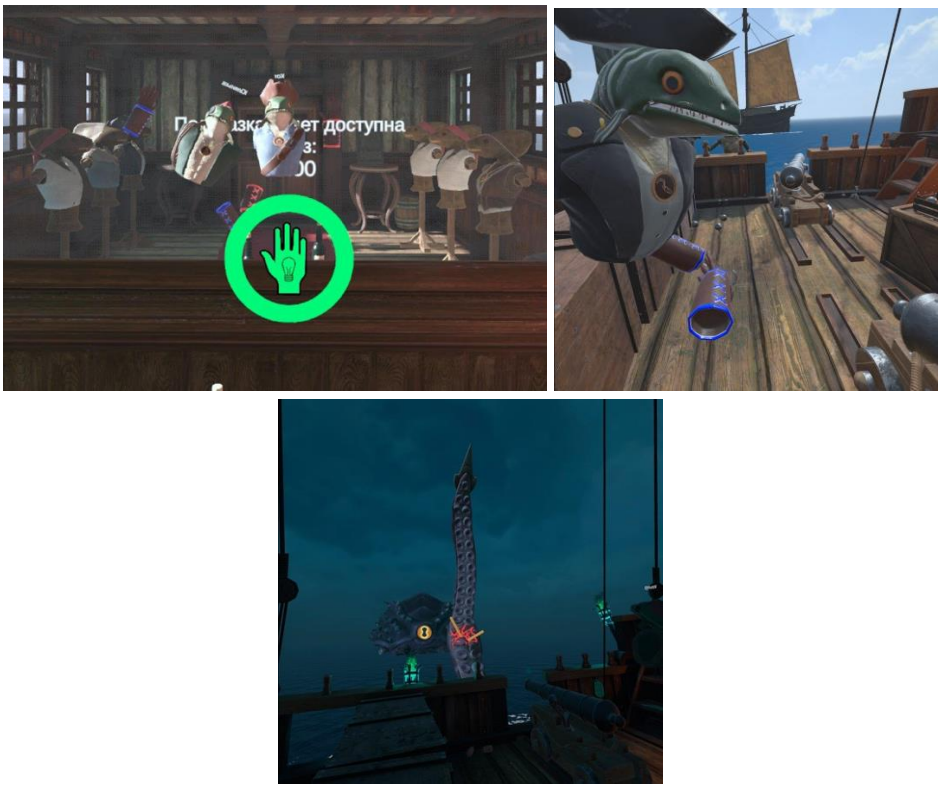


Рис. 4. Игровой процесс многопользовательской VR-игры «Пиратская битва»

## ЗАКЛЮЧЕНИЕ

Разработка документа игрового дизайна для VR-MMORPG представляет собой значительный вызов, требующий синтеза традиционных подходов к ММО-играм с инновационными решениями в области виртуальной реальности. Предложенная методология построения GDD учитывает ключевые особенности как VR-технологий (иммерсивность, физическое взаимодействие, комфорт), так и многопользовательского компонента (сетевая архитектура, социальные взаимодействия, экономика).

Особую ценность для индустрии представляет разработанная структура GDD, позволяющая систематически документировать сложные взаимодействия между различными игровыми системами в контексте VR. Модульный подход обеспечивает возможность итеративного развития документа в процессе разработки, что критически важно для проектов такой сложности.

Перспективные направления исследования включают разработку формализованных языков описания VR-взаимодействий, стандартизацию подходов к документированию сетевых архитектур для VR-игр, а также создание автоматизированных инструментов для генерации и проверки документации игрового дизайна. Особого внимания заслуживает интеграция систем искусственного интеллекта как для повышения качества игрового опыта, так и для упрощения процесса разработки документации.

В заключение отметим, что стандартизация подходов к созданию GDD для VR-MMORPG не только облегчит разработку конкретных проектов, но и будет способствовать развитию индустрии в целом, предоставляя разработчикам общий язык для описания и реализации виртуальных миров нового поколения.

## ЛУДОГРАФИЯ

Into the Radius (2019). CM Games. [VR: HTC Vive, Oculus Rift, Valve Index, Windows Mixed Reality].

Zenith: The Last City (2022). Ramen VR. [VR: Oculus Quest, Oculus Rift, PlayStation VR, HTC Vive, Valve Index].

Half-Life: Alyx (2020). Valve Corporation. [VR: HTC Vive, Oculus Rift, Valve Index, Windows Mixed Reality].

Пиратская битва (2025). MindEdges. [VR: Oculus Quest II].

**СПИСОК ЛИТЕРАТУРЫ**

1. *Hecks E.* Navigating the UI Challenges of Multi-User VR. 2024. URL: https://arinsider.co/2024/06/13/navigating-the-ui-challenges- of-multi-user-vr/

2. *Jagneaux D.* How VR MMO Zenith Is Being 'Built for VR' First and Foremost To Feel Like You're 'Living A Different Life'. 2021. URL: https://www.uploadvr.com/zenith-vr-mmo-built-for-vr-living-a-different-life/

3. How to Create a Game Design Document with the Help of AI Tools. 2023. URL: https://ludo.ai/blog/how-to-create-a-game-design-document-with-the-help-of-ai-tools

4. *Суббота А.* Геймдизайн VR-игр: особенности и рекомендации. 2022. URL: https://dtf.ru/gamedev/1250012-geimdizain-vr-igr-osobennosti-i-rekomendacii

5. How does network latency affect multi-user VR environments? URL: https://zilliz.com/ai-faq/how-does-network-latency-affect-multiuser-vr-environments

6. *Abhinaya S.B., Sabir A., Das A.* Enabling Developers, Protecting Users: Investigating Harassment and Safety in VR // 33rd USENIX Security Symposium (USENIX Security 24). 2024. P. 6561–6578. https://doi.org/10.48550/arXiv.2403.05499

7. Create AI Characters Using 3D Avatars for VR Games and Simulations. 2024. URL: https://convai.com/ blog/create-ai-characters-vr-games-simulations

8. *Christiansen F.R., Hollensberg L.N., Jensen N.B., Julsgaard K., Jespersen K.N., Nikolov I.* Exploring Presence in Interactions with LLM-Driven NPCs: A Comparative Study of Speech Recognition and Dialogue Options // In Proceedings of the 30th ACM Symposium on Virtual Reality Software and Technology (VRST '24). 2024. No. 6, P. 1–11. https://doi.org/https://doi.org/10.1145/3641825.3687716

9. *Woodman H.* The Power of ChatGPT: A Dungeon Master's Guide to Enhancing D&D Games. 2023. URL: https://www.linkedin.com/pulse/power-chatgpt-dungeon-masters-guide-enhancing-dd-games-woodman

10. Tool for creating game design document. 2024. URL: https://gamedev.net/forums/topic/717555-tool-for-creating-game-design-document/

11. *Warning C.* Creating a Game Design Document: The Sooner, the Better. 2014. URL: https://www.gamedeveloper.com/ design/creating-a-game-design-document-the-sooner-the-better

12. *Lang B.* Valve Explains the Deceptively Simple Design Process That Made «Half-Life: Alyx» Excellent. 2020.
URL: https://www.roadtovr.com/valve-half-life-alyx-game-design-interview-robin-walker/

13. *Hung T.D.* Designing a Virtual Reality Psychological Game. 2017. URL: https://www.theseus.fi/bitstream/handle/10024/139166/Tran_Duy_Hung.pdf?sequence=7

14. Unzen Dungeon VR: Game Design Document. 2023. URL: https://portfolio-tools.s3.eu-west-2.amazonaws.com/wp-content/uploads/2023/11/29005500/FMP-Game-Design-Document.pdf

15. *Wajman R.* Mind the Game VR: Immersive Museum Experience GDD. Bachelor's Thesis in Business Information Technology. 2023.
URL: https://static1.squarespace.com/static/635a52bb9bf5dc61aeff7e16/t/650b3d9ca9f672778aae0247/1695235487920/Mind+The+Gap+VR+GDD.pdf

16. *Shigapov M.I., Kugurakova V.V., Zykov E.Y.* Design of Digital Gloves with Feedback for VR // Proceedings of 2018 IEEE East-West Design and Test Symposium, EWDTS 2018. 2018. Art. No. 8524807.

17. *Kugurakova V.V., Elizarov A.M., Khafizov M.R.* Towards the immersive VR: measuring and assessing realism of user experience // ICAROB 2018: Proceedings of the 2018 international conference on artificial life and robotics. 2018. P. 146–152.

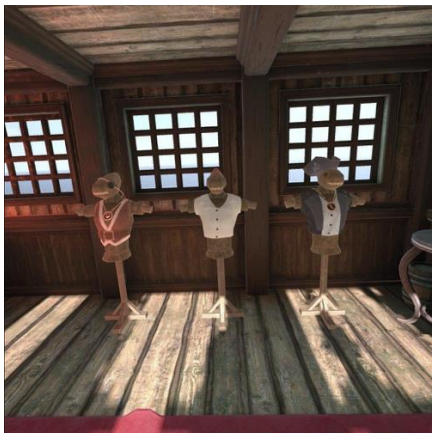
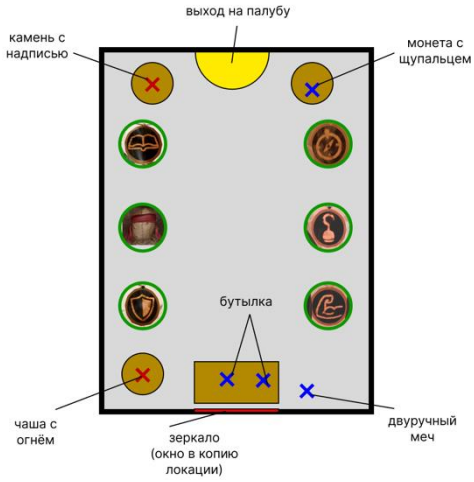
18. *Сахибгареева Г.Ф., Кугуракова В.В.* Практики балансирования компьютерных игр // Программные системы: теория и приложения. 2022. Т. 13. № 3. С. 255–273.

## Приложение. Фрагменты GDD, иллюстрирующие разработанную структуру

| | |
|---|---|
| **1. Базовая концепция и техническая инфраструктура** | |
| Концепция игры, целевая аудитория, уникальное торговое предложение | Разрабатываемая система рассматривается как комплекс виртуальных элементов и механик, формирующих многопользовательский игровой опыт в виртуальной реальности. Целевой аудиторией являются посетители VR-парков. |
| Требования к оборудованию (минимальные спецификации HMD, сетевые требования) | **Требования к программному обеспечению**<br>Система состоит из двух частей: серверная (файл в формате .exe) и пользовательская (приложение в формате .apk для запуска на VR-шлеме).<br>Серверная часть является интерфейсом управления Администратора.<br>Целевые устройства Системы:<br>• ПК на ОС Windows 10 или выше (для сервера),<br>• гарнитура виртуальной реальности Oculus Quest 2 (для приложения).<br>**Требования к аппаратному обеспечению**<br>Минимальные требования к персональному компьютеру для запуска Системы:<br>• Процессор Intel Core i5-10300H;<br>• Оперативная память – 8 Гб DDR4;<br>• Графическая карта: Nvidia GeForce RTX 3050 с 4Гб видеопамяти. |
| **2. VR-специфические элементы** | |
| Система передвижения (локомоции) и телепортации | Для перехода между комнатами игроки могут опускать «рычаги» рядом с воротами к комнате. После использования рычага все игроки при помощи эффекта затемнения появляются в соответствующей комнате. |
| Предотвращение укачивания | Виньетирование при перемещении между сценами |
| Физическое взаимодействие с объектами | Интерактивный предмет – предмет, с которым возможно взаимодействие. В зависимости от назначения для интерактивных предметов доступны следующие действия: удерживание для переноса в руке требует нажатия и удержания Курка, перенос тяжелых предметов требует нажатия и удержания Курка на двух контроллерах, иначе предмет валится к земле, толкание или касание кистями Персонажа Игрока, захват с помощью удерживания Курок, нанесение урона. При взятии интерактивный предмет фиксируется в руке в том же положении, в котором и был подобран (например, Саблю можно схватить за лезвие). Интерактивный предмет можно бросить, отпустив Курок во время движения руки. Скорость движения руки и вес предмета напрямую влияют на дальность его полета. Интерактивный предмет в руке Игрока можно использовать на других объектах. При этом возможны следующие способы использования предмета: нанесение урона при касании оружием, например, Лианы уничтожаются касанием Сабли, Бросаемые Кракеном Бочки и Ящики уничтожаются касанием Сабли/Меча…<br>Выход на палубу на уровне «КАЮТА КАПИТАНА» активируется удерживанием двери выхода на палубу Крюком Игрока с ролью «Капитан»… |

| | |
|---|---|
| Эргономические аспекты и продолжительность игровых сессий | Игрокам доступно 45 минут для прохождения игры. Если Игроки не могут успешно завершить квест, по истечению времени Игроки проигрывают и весь прогресс сбрасывается. Администратору доступна функция изменения времени сессии непосредственно во время игры. |
| **3. Диегетический интерфейс и UX** | |
| Интеграция UI в игровой мир<br><br>**Субтитры** | Игроки могут видеть часть дополнительной информации в виде субтитров (рис. 5). Субтитры всегда находятся в нижней части экрана вне зависимости от местоположения и вращения персонажа Игрока. Текст субтитров Автора истории находится внутри чёрной полупрозрачной рамки. Текст субтитров после выбора роли Персонажа Игрока находится внутри рамки в виде папируса.<br>  <br><br>Рис. 5. Субтитры в игровом процессе |
| **Всплывающие подсказки** | Всплывающие подсказки отображают дополнительный текст о различных предметах окружения. Их видит только тот Игрок, который приблизился к предмету. Всплывающие подсказки зафиксированы на одном месте и поворачиваются лицом к Игроку. |
| **Подсказки** | В каждой комнате уровня «ХРАМ» Игроки могут использовать подсказки. Их количество ограничено и определяется Администратором. Для активации подсказки Игроку необходимо удерживать руку до заполнения шкалы в зоне, где изображена рука с лампочкой (рис. 6).<br>  <br>Рис. 6. Активация подсказки<br><br>Подсказка представляет собой полупрозрачный Силуэт, указывающий на место, где лежит необходимый предмет, и место его применения (рис. 7, 8). |

| | |
|---|---|
| |  <br>Рис. 7. Подсказка «полупрозрачный силуэт» предмета «Сабля»<br>  <br>Рис. 8. Подсказка «полупрозрачный силуэт» предмета «Крюк» |
| Пространственный звук и аудио-навигация | Приближение вражеского корабля на уровне БОЙ С ПИРАТАМИ характеризуется звоном колокола. |
| Тактильная обратная связь | Получение урона Игроком сопровождается вибрированием конкретного контроллера при травме руки либо обоих контроллеров – при травме головы. При наличии травмы экран Игрока обрамляется красной виньеткой. |
| Инвентарь и меню как физические объекты внутри VR | Ролевые приспособления находятся на руках Персонажа Игрока (рис. 9), имеющего конкретную роль: (а) Компас для роли Штурман показывает направление до Монеток; (б) Таймер для роли Капитан показывает оставшееся время сессии для Игроков, при этом таймер не показывает больше 60 минут и имеет 60 делений с метками 0, 15, 30, 45.<br>  <br>Рис. 9. Инвентарь персонажей размещен на их манжетах |

| | |
|---|---|
| Персонажи игроков | Персонажи Игроков представлены в виде Пиратов с рыбьей головой, показывается только верхняя часть туловища и кисти рук в перчатках (рис. 10).<br>  <br>Рис. 10. Персонажи пиратов-«рыб»<br>Персонаж Игрока способен толкать и использовать предметы при приближении. Игроку доступны 6 ролей в локации КАЮТА КАПИТАНА (рис. 11). Смена выбранной роли возможна по нажатию на Курок на Манекене с одеждой.<br>  <br>Рис. 11. Схема размещения персонажей в лобби |

# KEY ASPECTS OF DEVELOPING A GAME DESIGN DOCUMENT FOR MULTIPLAYER VIRTUAL REALITY GAMES

**Y. A. Karpeeva[1] [0009-0005-1632-2914], M. R. Khafizov[2] [0000-0001-7275-9102]**

[1, 2]*Institute of Information Technologies and Intelligent Systems, Kazan Federal University*

[1]leyaleit@gmail.com, [2]murkorp@gmail.com

***Abstract***

Unique aspects of game design documentation development for massively multiplayer online role-playing game in virtual reality (VR) are considered. Relevance and problems of designing such games are discussed, existing approaches to creating and adapting documentation for VR are described, and also such key issues in VR as network interaction, immersion, interaction of players and non-player characters controlled by artificial intelligence are defined. A methodological approach to the construction of a game design document for games of this genre is proposed, using the example of a technical specification of several games with a focus on structuring and standardizing the document. Future directions are discussed, including form-mal descriptions of game logic for scene portability, perspectives on auto-automated VR porting, and tools for auto-generation. Finally, the importance of standardizing technical specifications is emphasized and directions for further research are suggested.

***Keywords:*** *GDD, VR, MMORPG, Game Design, Immersive Experience, Virtual Reality, Automated Development, Network Architecture, User Experience, Documentation Standardization.*

**LUDOGRAPHY**

Into the Radius (2019). CM Games. [VR: HTC Vive, Oculus Rift, Valve Index, Windows Mixed Reality].

Zenith: The Last City (2022). Ramen VR. [VR: Oculus Quest, Oculus Rift, PlayStation VR, HTC Vive, Valve Index].

Half-Life: Alyx (2020). Valve Corporation. [VR: HTC Vive, Oculus Rift, Valve Index, Windows Mixed Reality].

Piratskaya bitva (2025). MindEdges. [VR: Oculus Quest II].

**REFERENCES**

1. *Hecks E.* Navigating the UI Challenges of Multi-User VR. 2024. URL: https://arinsider.co/2024/06/13/navigating-the-ui-challenges- of-multi-user-vr/

2. *Jagneaux D.* How VR MMO Zenith Is Being 'Built for VR' First and Foremost To Feel Like You're 'Living A Different Life'. 2021.
URL: https://www.uploadvr.com/zenith-vr-mmo-built-for-vr-living-a-different-life/

3. How to Create a Game Design Document with the Help of AI Tools. 2023. URL: https://ludo.ai/blog/how-to-create-a-game-design-document-with-the-help-of-ai-tools

4. *Subbota A.* Geimdizain VR-igr: osobennosti I rekomedatsii. [VR game design: features and recommendations.] 2022. URL: https://dtf.ru/gamedev/1250012-geimdizain-vr-igr-osobennosti-i-rekomendacii (In Russian).

5. How does network latency affect multi-user VR environments? URL: https://zilliz.com/ai-faq/how-does-network-latency-affect-multiuser-vr-environments

6. *Abhinaya S.B., Sabir A., Das A.* Enabling Developers, Protecting Users: Investigating Harassment and Safety in VR. // 33rd USENIX Security Symposium (USENIX Security 24). 2024. P. 6561–6578. https://doi.org/10.48550/arXiv.2403.05499

7. Create AI Characters Using 3D Avatars for VR Games and Simulations. 2024. URL: https://convai.com/ blog/create-ai-characters-vr-games-simulations

8. *Christiansen F.R., Hollensberg L.N., Jensen N.B., Julsgaard K., Jespersen K.N., Nikolov I.* Exploring Presence in Interactions with LLM-Driven NPCs: A Comparative Study of Speech Recognition and Dialogue Options // In Proceedings of the 30th ACM Symposium on Virtual Reality Software and Technology (VRST '24). 2024. No. 6, P. 1–11. https://doi.org/10.1145/3641825.3687716

9. *Woodman H.* The Power of ChatGPT: A Dungeon Master's Guide to Enhancing D&D Games. 2023. URL: https://www.linkedin.com/pulse/power-chatgpt-dungeon-masters-guide-enhancing-dd-games-woodman

10. Tool for creating game design document. 2024.
URL: https://gamedev.net/forums/topic/717555-tool-for-creating-game-design-document/

11. *Warning C.* Creating a Game Design Document: The Sooner, the Better. 2014. URL: https://www.gamedeveloper.com/design/creating-a-game-design-document-the-sooner-the-better

12. *Lang B.* Valve Explains the Deceptively Simple Design Process That Made «Half-Life: Alyx» Excellent. 2020. URL: https://www.roadtovr.com/valve-half-life-alyx-game-design-interview-robin-walker/

13. *Hung T.D.* Designing a Virtual Reality Psychological Game. 2017. URL: https://www.theseus.fi/bitstream/handle/10024/139166/Tran_Duy_Hung.pdf?sequence=7

14. Unzen Dungeon VR: Game Design Document. 2023. URL: https://portfolio-tools.s3.eu-west-2.amazonaws.com/wp-content/uploads/2023/11/29005500/FMP-Game-Design-Document.pdf

15. *Wajman R.* Mind the Game VR: Immersive Museum Experience GDD. Bachelor's Thesis in Business Information Technology. 2023.
URL: https://static1.squarespace.com/static/635a52bb9bf5dc61aeff7e16/t/650b3d9ca9f672778aae0247/1695235487920/Mind+The+Gap+VR+GDD.pdf

16. *Shigapov M.I., Kugurakova V.V., Zykov E.Y.* Design of Digital Gloves with Feedback for VR // Proceedings of 2018 IEEE East-West Design and Test Symposium, EWDTS 2018. 2018. Art. No. 8524807.

17. *Kugurakova V.V., Elizarov A.M., Khafizov M.R.* Towards the immersive VR: measuring and assessing realism of user experience // ICAROB 2018: Proceedings of the 2018 international conference on artificial life and robotics. 2018. P. 146–152.

18. *Sahibgareeva G.F., Kugurakova V.V.* Game Balance Practices // Program Systems: Theory and Applications. 2022. Vol. 13. No. 3(54). P. 255–273. (In Russian).

## СВЕДЕНИЯ ОБ АВТОРАХ

***КАРПЕЕВА Юлия Алексеевна*** – студент Института информационных технологий и интеллектуальных систем Казанского федерального университета. Направление исследований: проектный менеджмент в игровой разработке.

***Yuliya Alexeevna KARPEEVA*** – student at the Institute of Information Technologies and Intelligent Systems of Kazan Federal University. Research area: project management in game development.

email: leyaleit@gmail.com
ORCID: 0009-0005-1632-2914

***ХАФИЗОВ Мурад Рустэмович*** – старший преподаватель Института ИТИС КФУ. Сфера научных интересов: разработка видеоигр, виртуальная реальность, синтетические данные.

***Murad Rustemovich KHAFIZOV*** – senior lecturer at the Institute of Information Technologies and Intelligent Systems, Kazan Federal University. Research interests: game development, virtual reality, synth data.

email: murkorp@gmail.com
ORCID: 0000-0001-7275-9102

*Материал поступил в редакцию 28 января 2025 года*

UDC 004.946+616.12-007.2-073.432.19-071

# DEVELOPMENT OF A VIRTUAL REALITY TRAINER FOR THE PRENATAL DETECTION OF CONGENITAL HEART DISEASE

**O. Correa Madrigal[1] [0000-0002-1135-685X], J. E. Perdomo Batista[2] [0009-0008-4194-1268], C. Garcia Guevara[3] [0009-0007-6967-6710]**

[1, 2]*Center for Interactive Technologies, University of Computer Sciences, Cuba;*
[3]*William Soler Cardiocenter, Cuba*

[1]ocorrea@uci.cu, [2]javierb@estudiantes.uci.cu, [3]cardiocentrows@infomed.sld.cu

***Abstract***

The Virtual Reality Trainer for the Prenatal Detection of Congenital Heart Disease and Associated Malformations project addresses the need to improve specialized training in prenatal diagnosis through immersive technologies. Its theoretical module "My Friend the Lung" gamifies learning about fetal cardiovascular anatomy through 3D puzzles, timed challenges, and badges, achieving good results in its pilot test, is noteworthy. The practical module simulates ultrasounds with haptic devices, reducing diagnostic errors and enabling the generation of personalized ultrasounds. A database with 1,200 labeled studies was consolidated as a fundamental source of information. The implementation aims to initially reduce clinical errors, with the potential to decrease perinatal mortality. This project fuses technological innovation, medical rigor, and interactive pedagogy, positioning gamification as an essential tool in medical education and laying the groundwork for its expansion to other specialties.

***Keywords****: virtual reality, congenital heart disease, gamification, prenatal diagnosis, medical training, clinical simulation.*

## INTRODUCTION

Congenital heart defects are one of the leading causes of perinatal and infant mortality worldwide, with an incidence of 6 to 8 cases per 1,000 live births. In Cuba, these anomalies are the second leading cause of death in children under one year of age, highlighting the need to improve the training of specialists in prenatal diagnosis. Fetal echocardiography is the technique of choice for identifying these pathologies; however, its training requires years of experience and access to real-life cases, limiting its reach in regions with limited resources.

Faced with this challenge, the joint Cuba-Russia 2025 project emerged, whose objective is to develop a virtual reality (VR) trainer for the prenatal detection of congenital heart disease and associated malformations. This initiative seeks to integrate emerging technologies, such as VR and gamification, to create an accessible and standardized continuing education system. The project is based on the experience of the William Soler Cardiocenter in Cuba and on collaboration with academic and technological institutions in both countries, promoting innovation in medical education and reducing disparities in healthcare.

**RELATED WORKS**

Ultrasound is the imaging technique of choice for fetal studies. Intrauterine, and echocardiography is irreplaceable for the identification of cardiac structures and the analysis of fetal physiology. The fundamental purpose of prenatal diagnosis is obtain genetic, anatomical, biochemical, and physiological information about the fetus, and analyze whether any abnormalities are expected that will impact both the fetal and postnatal periods. However, there is no non-imaging-based test, whether genetic or biochemical, capable of diagnosing congenital heart disease. To date, it is the only method for identifying the specific type of cardiovascular malformation.

Beginning in 1982, interest in the application of echocardiography to the study of congenital heart disease began in Cuba, and its inclusion in some departments across the country, particularly the William Soler Pediatric Teaching Hospital. With exemplary modesty on the part of its professionals, in the Cardiology Department of this hospital, and initially self-taught, the first diagnoses were made in children with congenital malformations and acquired heart diseases, primarily rheumatic carditis. This was followed by courses and training both within and outside the country, thus ushering in the second great cardiological revolution of the 20th century, and with it, the era of echocardiography.

The progressive decline in infant mortality, which had already been achieved by those years thanks to a series of factors, such as improved living conditions and nutritional status, the drastic reduction in diarrheal and acute respiratory diseases, and the eradication of numerous infectious diseases through immunization, led to a radical change in the epidemiological profile, with other more relevant causes now taking a

significant role in infant mortality. Congenital malformations have since become the second leading cause of death in children under one year of age, half of which are cardiovascular. The incidence of these malformations ranges between 6 and 8 per 1,000 live births, of which 25% are complex, difficult to treat, and have a poor prognosis.

Since its inception, the Cuban program has attracted the attention of other international centers in France, Great Britain, and the United States. Although they all share the inclusion of four-chamber imaging in routine obstetric examinations as a factor in suspecting heart disease, the Cuban program is the only one in which 100% of pregnant women could be screened, thanks to the organization of its National Public Health System. This led to the creation of a large database of cases that were properly diagnosed and digitized in multiple formats. This program addressed the problems arising from the Comprehensive treatment of heart disease, including morphological examination of the heart, treatment of fetal arrhythmias, monitoring of high-risk pregnant women, and genetic counseling by the prenatal diagnostic team, comprised of pediatric cardiologists, obstetricians-gynecologists, geneticists, embryologists, ultrasound technicians, and pediatricians. This phase lasts for a decade, and begins another phase requiring greater use of information and communications technologies (ICTs).

In the last 5 decades, prenatal diagnosis has been perfected and has achieved an impact favorably in sensitive health indicators such as perinatal and infant mortality. The Cuban center is requesting specialized training of greater scope and accessibility from different parts of Latin America. At the same time, progress continues to be made not only in diagnosis, but also in other important areas, such as physiology, the dynamics of placental circulation and its impact on pregnancy outcomes, cardiac rhythm disorders, fetal pharmacology (about which almost nothing is known), and the treatment of fetal heart disease, whether pharmacological, through interventional catheterization, or fetal surgery. In a context where knowledge products are important sources of income for countries, the Cuban program is advancing the intensive use of computer technology to strengthen training and diagnostic systems as high- quality professional services. In this regard, progress is being made in the development of joint projects with the University of Informatics Sciences.

The Interactive Technologies Center (VERTEX) belonging to the University of Computer Sciences has specialized for 12 years in products associated with video

games, virtual reality and augmented reality. In this context, it has participated in the creation of car driving simulators [1], virtual laboratories [2] and serious video games for the rehabilitation of different disabilities [3, 4]. Since 2016, collaboration relations began with the MediaLab center, belonging to the Institute of Information Systems and Intelligent Systems of the Kazan Federal University, Russia. It has extensive experience in the application of virtual reality for the training of surgeons and in platforms for the creation of multi-user simulators with immersion in virtual environments for cooperative learning. In the period 2016–2024, there has been a cooperative contribution to the development of trainers for appendectomy surgeries [5, 6] and more recently to a trainer for the birthing process which uses emerging technologies such as interaction through hand gestures [7].

Echocardiography has different procedures that involve a complex learning curve. Manual dexterity, pattern recognition and image interpretation are necessary skills to achieve an effective diagnosis. The use of echocardiography simulators has evolved from mannequins [8] to the use of virtual reality [9]. Its use has proven to be effective in the training of specialists with demonstrated levels of effectiveness [10]. This technology is already recognized as fundamental in medical training and in the case of echocardiography, recent studies summarize a path of success in its application [11]. However, in relation to prenatal echocardiography, no trainers of this type have been identified, mainly due to the complexities and current state of medical research.

## RESULTS

The Virtual Reality Trainer for Prenatal Detection of Congenital Heart Disease and Associated Malformations project works in several directions, from the creation of innovative technological tools to the validation of their impact on medical training. Below are several results achieved so far, with special emphasis on the theoretical gamification module “My Friend the Pulmonary,” designed to revolutionize the learning of fetal cardiovascular anatomy.

### Integrated Clinical Case Database

A database with more than 1,200 prenatal ultrasound studies was consolidated, Compiled from the historical archive of the “William Soler” Cardiocenter (1982–2025). Each case includes:

- Video classification: Types of heart disease (e.g., tetralogy of Fallot, ventricular septal defect) and associated malformations (e.g., Holt-Oram syndrome) were segmented.
- Clinical metadata: Gestational age, postnatal findings, treatments applied and long-term outcomes.
- Collaborative labeling: This was performed by pediatric cardiologists and prenatal diagnosticians, ensuring accurate classification.

An accessible search engine was configured using a smart search interface that allows users to filter cases by complexity, frequency of occurrence, or clinical relevance, serving as an educational and reference resource for future research.

**Theoretical Module My Friend the Pulmonary**

This module, a pioneer in the gamification of fetal cardiovascular anatomy, focuses on teaching the layout of the pulmonary arteries and cardiac structures through immersive game mechanics. The layout of the pulmonary vessels and arteries is presented as personal, with varying levels of relationship to conditions. A visual-textual and visual-textual association approach seeks to reinforce comparative associative learning. The layout of the pulmonary artery, aorta, and vena cava is analyzed based on size and position. Different combinations of these allow for prediagnosis of most known pathologies.

Gamified Narrative: Users take on the role of “medical explorers” who must identify the correct anatomy of the vascular structures of a virtual heart affected by malformations. The narrative is structured into missions (minigames), with each level corresponding to a specific heart condition.

- 3D Puzzles: Reconstruction of pulmonary arteries and cardiac chambers by dragging anatomical components.
- Time Challenges: Rapid identification of anomalies in rotating 3D models, with scoring based on accuracy and speed.
- Achievements and Badges: Badges awarded for completing challenges (e.g., Valve Master for mastering aortic stenosis identification). Each progression in the minigames unlocks relevant aspects of the Knowledge Catalog, which can be

converted into general-purpose or customized resources for improved system performance.

Complementing the theoretical module is the ultrasound training process. This focuses on simulating ultrasound imaging in a realistic virtual environment with interaction through haptic devices. The main components are:

- Immersive 3D Environment: Virtual patient models with anatomical variations (e.g., fetal position, maternal obesity) that affect image acquisition.
- Haptic Devices: Simulation of resistance when moving the virtual transducer, replicating the sensation of a real exam.
- Calculation of correspondence between real ultrasound and virtual movement: A neural network guides the visualization of a study using the movements of the virtual transducer as input.
- Performance Indicators: Real-time metrics on transducer angle, imaging depth, and accurate diagnosis.

**CONCLUSIONS**

The development of the Virtual Reality Trainer for Prenatal Detection of Congenital Heart Disease represents a significant advance in specialized medical training, integrating immersive technologies and gamification to address critical challenges in prenatal diagnosis. The theoretical module, “My Friend the Lung,” proved transformative, increasing anatomical knowledge retention and reducing the time to identify anomalies in pilot tests with residents. Its gamified design, based on interactive mechanics such as 3D puzzles and timed challenges, not only improves user motivation but also facilitates the intuitive understanding of complex cardiac structures. These results underscore the potential of virtual reality to democratize access to high-quality training, especially in resource-constrained regions, while establishing a replicable model for other medical specialties.

The Cuba–Russia collaboration, based on clinical experience with the Cardiocenter “William Soler” and the technological expertise of the University of Computer Sciences and Kazan Federal University have set a precedent in educational innovation in healthcare. The implementation of the beta version of the trainer in medical centers in Latin America and Russia not only suggests an opportunity to improve birth rates but

also opens the door to commercializing this technology as a global service. Future research should expand the database with international cases and evaluate the long-term impact on perinatal mortality.

**REFERENCES**

1. *Coca Bergolla Y., Guzmán Montoto J.I.* Traffic control in a car driving simulator: Master Thesis // University of Informatics Sciences. 2007. 65 p.
URL: https://repositorio.uci.cu/handle/ident/TD_0960_07 (In Spanish).

2. *Reyes H.L.N., Tellez O.F.* Laboratorio Virtual como herramienta para potenciar las competencias de Física en estudiantes de Bachillerato // Serie Científica de la Universidad de las Ciencias Informáticas. 2022. Vol. 15. No. 5. P. 145–165.

3. *Correa Madrigal O.* Virtual Relax, System of Virtual Reality for Relaxation Therapy // NeuroRehabana. 2010.
URL: https://promociondeeventos.sld.cu/neurorehabana2010ingles/section-2sciences-and-disciplines-involved-in-theneurore-habilitation/

4. *Valdés Fernández D.* VirtualRelax, plataforma para integrar módulos de entornos virtuales, para apoyar a los especialistas en el tratamiento del estrés: Bachelor Thesis //University of Informatics Sciences. 2015. 78 p.
URL: https://repositorio.uci.cu/handle/123456789/8010

5. *Kugurakova V.V., Khafizov M.R., Correa Madrigal O. et al.* Virtual surgery system with realistic visual effects and haptic interaction // Proc. of The International Conference on Artificial Life and Robotics (ICAROB 2017, Japan). 2017. P. 86–89.

6. *Véliz Vega A., Correa Madrigal O., Kugurakova V.V.* Aprendizaje adaptativo basado en Simuladores de Realidad Virtual // Revista Cubana de Ciencias Informáticas. 2021. Vol. 15. No. 2. P. 138–157.

7. *Correa Madrigal O., Muhamethanov I.R., Kugurakova V.V.* A methodological overview about hand tracking and cognitive influences in virtual reality // HCI-COLLAB 2024, Colombia: Springer.

8. *Shakil O., Mahmood F., Matyal R.* Simulation in Echocardiography: An Ever-Expanding Frontier // Journal of Cardiothoracic and Vascular Anesthesia. 2012. Vol. 26. No. 3. P. 476–485. https://doi.org/10.1053/j.jvca.2012.01.019

9. *O'Sullivan D.M., Foley R., Proctor K. et al.* The Use of Virtual Reality Echocardiography in Medical Education // Pediatr Cardiol. 2021. Vol. 42. P. 723–726.

https://doi.org/10.1007/s00246-021-02596-z

10. *Khoche S., Maus T.* The reality of virtual reality in echocardiography education? //Journal of Cardiothoracic and Vascular Anesthesia. 2023. Vol. 37. No. 2. P. 306–307.

11. *Bouraghi H. et al.* Virtual reality and cardiac diseases: A systematic review of applications and effects // Journal of Healthcare Engineering. 2023. Vol. 2023. No. 1. P. 8171057. https://doi.org/10.1155/2023/8171057

# РАЗРАБОТКА ТРЕНАЖЕРА ВИРТУАЛЬНОЙ РЕАЛЬНОСТИ ДЛЯ ПРЕНАТАЛЬНОЙ ДИАГНОСТИКИ ВРОЖДЕННЫХ ПОРОКОВ СЕРДЦА

**О. Корреа Мадригал[1] [0000-0002-1135-685X], Дж. Э. Пердомо Батиста[2] [0009-0008-4194-1268], К. Гарсия Гевара[3] [0009-0007-6967-6710]**

[1, 2]*Центр интерактивных технологий, Университет компьютерных наук, Куба;*

[3]*Кардиоцентр имени У. Солера, Куба*

[1]ocorrea@uci.cu, [2]javierb@estudiantes.uci.cu, [3]cardiocentrows@infomed.sld.cu

***Аннотация***

Проект разработки тренажера виртуальной реальности для пренатального выявления врожденных пороков сердца и сопутствующих заболеваний направлен на улучшение специализированной подготовки в области пренатальной диагностики с помощью иммерсивных технологий. Особенно заслуживает внимания теоретический модуль «Мой друг – легкое», который позволяет в игровой форме изучать анатомию сердечно-сосудистой системы плода с помощью 3D-головоломок, заданий на время и бейджей, добиваясь хороших результатов в ходе пилотного тестирования. Практический модуль имитирует УЗИ с помощью тактильных устройств, снижая количество диагностических ошибок и позволяя создавать пер-

сонализированные УЗИ. В качестве основного источника информации была собрана база данных с 1200 маркированными исследованиями. Реализация проекта направлена на первоначальное снижение количества клинических ошибок, что может привести к снижению перинатальной смертности. Этот проект объединяет технологические инновации, медицинскую строгость и интерактивную педагогику, позиционируя геймификацию как важный инструмент в медицинском образовании и закладывая основу для его распространения на другие специальности.

***Ключевые слова**: виртуальная реальность, врожденные пороки сердца, геймификация, пренатальная диагностика, медицинское обучение, клиническое моделирование.*

**СПИСОК ЛИТЕРАТУРЫ**

1. *Coca Bergolla Y., Guzmán Montoto J.I.* Traffic control in a car driving simulator: Master Thesis // University of Informatics Sciences. 2007. 65 p.
URL: https://repositorio.uci.cu/handle/ident/TD_0960_07 (In Spanish).

2. *Reyes H.L.N., Tellez O.F.* Laboratorio Virtual como herramienta para potenciar las competencias de Física en estudiantes de Bachillerato // Serie Científica de la Universidad de las Ciencias Informáticas. 2022. Vol. 15. No. 5. P. 145–165.

3. *Correa Madrigal O.* Virtual Relax, System of Virtual Reality for Relaxation Therapy // NeuroRehabana. 2010.
URL: https://promociondeeventos.sld.cu/neurorehabana2010ingles/
section-2sciences-and-disciplines-involved-in-theneurore-habilitation/

4. *Valdés Fernández D.* VirtualRelax, plataforma para integrar módulos de entornos virtuales, para apoyar a los especialistas en el tratamiento del estrés: Bachelor Thesis //University of Informatics Sciences. 2015. 78 p.
URL: https://repositorio.uci.cu/handle/123456789/8010

5. *Kugurakova V.V., Khafizov M.R., Correa Madrigal O. et al.* Virtual surgery system with realistic visual effects and haptic interaction // Proc. of The International Conference on Artificial Life and Robotics (ICAROB 2017, Japan). 2017. P. 86–89.

6. *Véliz Vega A., Correa Madrigal O., Kugurakova V.V.* Aprendizaje adaptativo basado en Simuladores de Realidad Virtual // Revista Cubana de Ciencias Informáticas. 2021. Vol. 15. No. 2. P. 138–157.

7. *Correa Madrigal O., Muhamethanov I.R., Kugurakova V.V.* A methodological overview about hand tracking and cognitive influences in virtual reality // HCI-COLLAB 2024, Colombia: Springer.

8. *Shakil O., Mahmood F., Matyal R.* Simulation in Echocardiography: An Ever-Expanding Frontier // Journal of Cardiothoracic and Vascular Anesthesia. 2012. Vol. 26. No. 3. P. 476–485. https://doi.org/10.1053/j.jvca.2012.01.019

9. *O'Sullivan D.M., Foley R., Proctor K. et al.* The Use of Virtual Reality Echocardiography in Medical Education // Pediatr Cardiol. 2021. Vol. 42. P. 723–726. https://doi.org/10.1007/s00246-021-02596-z

10. *Khoche S., Maus T.* The reality of virtual reality in echocardiography education? //Journal of Cardiothoracic and Vascular Anesthesia. 2023. Vol. 37. No. 2. P. 306–307.

11. *Bouraghi H. et al.* Virtual reality and cardiac diseases: A systematic review of applications and effects // Journal of Healthcare Engineering. 2023. Vol. 2023. No. 1. P. 8171057. https://doi.org/10.1155/2023/8171057

## СВЕДЕНИЯ ОБ АВТОРАХ

***КОРРЕА МАДРИГАЛ Омар*** — профессор, директор Центра интерактивных технологий Университета информатики (Куба). Приглашенный профессор кафедры индустрии разрабьотки видеоигр Института информационных технологий и интеллектуальных систем Казанского федерального университета. Сфера научных интересов — различные аспекты разработки компьютерных игр и VR-тренажеров.

***Omar CORREA MADRIGAL*** — Dr., Full Professor, Director of the Center for Interactive Technologies at the University of Informatics (Cuba). He is also a visiting professor of the Department of Video Game Development Industry at the Institute of Information Technologies and Intelligent Systems of Kazan Federal University. His research interests include various aspects of computer games and VR simulators development.

email: ocorrea@uci.cu

ORCID: 0000-0002-1135-685X

***ПЕРДОМО БАТИСТА Джавье Эрнесто*** — студент факультета интерактивных технологий Университета информатики (Куба), руководитель студенческой группы разработки тренажера виртуальной реальности для пренатальной диагностики врожденных пороков сердца. Сфера научных интересов — технологии разработки VR-тренажеров.

***Javier Ernesto PERDOMO BATISTA*** — student of the Faculty of Interactive Technologies at the University of Informatics (Cuba), leader of the student group developing a virtual reality simulator for prenatal diagnosis of congenital heart defects. The sphere of scientific interests - VR simulator development technologies.

email: javierb@estudiantes.uci.cu

ORCID: 0009-0008-4194-1268

***ГАРСИЯ ГЕВАРА Карлос*** — ведущий специалист в области педиатрии и кардиологии Кардиоцентра имени Уильяма Солера (Куба), магистр в области комплексного ухода за ребенком, магистр в области детской эхокардиографии. Сфера научных интересов — детская эхокардиография.

***Carlos GARCIA GUEVARA*** — Leading specialist in Pediatrics and Cardiology at the William Soler Cardiocenter (Cuba), Master in Integrated Child Care, Master in Pediatric Echocardiography, Master in Pediatric Echocardiography. Her research interests include pediatric echocardiography.

email: cardiocentrows@infomed.sld.cu

ORCID: 0009-0007-6967-6710

*Материал поступил в редакцию 28 января 2025 года*

УДК 004.9+005

# ТИПИЗАЦИЯ КООПЕРАТИВНЫХ МЕХАНИК МНОГОПОЛЬЗОВАТЕЛЬСКИХ ВИДЕОИГР

**Д. А. Хаматнуров[1] [0009-0005-7925-1164], А. В. Шубин[2] [0000-0002-6203-3268]**

[1, 2]*Институт информационных технологий и интеллектуальных систем Казанского (Приволжского) федерального университета;*

[1]khamatnurovd@gmail.com, [2]shubin.aleksey.kpfu@gmail.com

***Аннотация***

Проведена типизация кооперативных механик многопользовательских видеоигр. Кооперативные игровые механики представляют собой ключевые элементы геймдизайна, определяющие способы взаимодействия игроков в совместном игровом процессе. Проанализированы существующие исследования в данной области и выделены основные принципы, влияющие на успешность кооперативного взаимодействия. Выделены и классифицированы восемь типов кооперативных механик: совместные, кооперативно-эмергентные, альтруистические, социально-экономические, комплементарные, механики одновременного управления, штрафующие и цепные.

Сделан вывод, что успешные кооперативные механики строятся на принципах комплементарности ролей, сочетании различных способностей и необходимости распределения задач ради достижения общей цели. Предложенная классификация способствует систематизации знаний в области геймдизайна и может быть полезна разработчикам многопользовательских видеоигр. В дальнейшем классификация может быть уточнена и расширена с учетом эволюции индустрии и появления новых форм кооперации.

***Ключевые слова****: игровая механика, кооперативные механики, многопользовательские видеоигры, геймдизайн, классификация, видеоигры.*

## ВВЕДЕНИЕ

Видеоигры — это относительно новая и на данный момент еще плохо формализованная форма медиа. Особенностью видеоигр, отличающей их от других

медиа, является интерактивность – возможность пользователя влиять на виртуальную среду.

Многопользовательские видеоигры являются одними из наиболее популярных проектов и собирают в общем более 4 млн игроков каждый день [1]. Согласно классификации Р. Бартла [2], значительная часть игроков (социальщики и киллеры) получает удовольствие от взаимодействия с другими: дружбы, командной работы или доминирования над соперниками. Исследование Quantic Foundry [3] также подтверждает, что 44% игроков мотивированы именно кооперацией, а 34% ценят конкуренцию.

Одна из форм многопользовательских игр – это кооперативные видеоигры, в которых игроки могут объединяться в коалиции для решения общей задачи. Кооперативные игровые механики — это концепция в дизайне видеоигр, которая подразумевает создание системы взаимодействия между игроками или несколькими виртуальными персонажами, где все члены коалиции получают взаимную выгоду от сотрудничества. Эта концепция может применяться как в многопользовательских, так и в однопользовательских играх и включает в себя разнообразные механики, направленные на углубление игрового процесса и повышение его интересности и комплексности.

Кооперация так или иначе ведёт к распределению ролей даже при равных возможностях игроков – участники коалиции стремятся выбрать наиболее подходящие позицию и стиль игры. При этом вклад каждого члена команды остаётся достаточно весомым, что порождает элемент непредсказуемости (ошибки игроков, иррациональное поведение, различные интересы игроков), усиливая интерес к совместному процессу.

Однако кооперативные механики могут быть реализованы в различных формах и по-разному влиять восприятие игрового процесса. В связи с этим и было решено провести их классификацию и проанализировать по-отдельности.

**СВЯЗАННЫЕ РАБОТЫ**

Классификация игровых механик остается открытым вопросом в рамках изучения видеоигр. В настоящее время глубокая теоретизация разработки видеоигр остро необходима, поскольку игры становятся всё более сложными, а высокая

конкуренция требует точного понимания предпочтений игроков и эффективного использования игровых механик [4].

Как отмечалось выше, классификация игровых механик остается актуальным исследованием и рассматривается во многих работах, например [5–7], но указанные статьи направлены на более высокоуровневую классификацию и не затрагивают кооперативные механики детально, решая другие задачи классификации.

Во многих работах (см., например [8–10]) рассматривается геймификация совместного обучения. В этих исследованиях определялись наиболее существенные практики, поддерживающие командную работу, а также увеличивающие эффективность обучения. В [9, 10] утверждается, что игроки будут действовать в одиночку, если дизайн игры не принуждает их к кооперации. При этом вынужденное сотрудничество позволяет лучше усваивать информацию и работать продуктивнее. Таким образом, при разработке кооперативных игр следует добавлять элементы разделения информации, ролей или ресурсов [8].

В другом исследовании [11] анализировались 14 кооперативных игр с целью определить наиболее вовлекающие аспекты кооперативных видеоигр. Авторы подчеркивают, что все выбранные видеоигры имели следующие общие черты в игровом процессе:

- Комплементарность – игроки могут иметь роли с различными возможностями, дополняющими друг друга (создающими новый эффект) или закрывающими чужие потребности.
- Синергия – игроки могут иметь способности, которые увеличивают эффективность другого игрока при одновременном применении.
- Совместное взаимодействие – игроки могут одновременно совершать одно и то же действие для повышения эффективности данного действия.
- Общие цели – условия победы идентичны для всех участников коалиции, а победа достигается лишь совместными усилиями.
- Ограничение ресурсов – игроки обязаны грамотно распределять ресурсы для поддержания эффективности игроков.
- Принуждение к сотрудничеству – игроки-одиночки не могут достигнуть победы без участия в коалиции.

Подтверждая результаты [11], в [12] выделены схожие паттерны (комплементарность, синергия, общие цели) для игрового процесса кооперативных видеоигр и предложены новые шаблоны в игровом процессе:

- Способности только для других игроков – механики, целью которых могут быть другие игроки, но не сам владелец функционала.
- Синергия целей – концепция разделения задач, выполнение которых одним игроком открывает новые возможности для другого.
- Специальные правила для союзников – механики, делающие взаимодействие в коалиции удобнее.

Таким образом, опираясь на исследования [8–12], можно сказать, что игроки склонны избегать кооперации, если дизайн игры не требует активного взаимодействия. Следовательно, при посредственном дизайне игрового процесса разработчик может потерять все преимущества многопользовательских видеоигр. Различные исследования подтверждают, что успешные кооперативные механики строятся на комплементарности ролей, синергии способностей, разделении задач и распределении ресурсов, что обязывает игроков взаимодействовать и работать в команде для повышения личной эффективности.

## КЛАССИФИКАЦИЯ КООПЕРАТИВНЫХ МЕХАНИК

Ранее отмечалось, что добавление кооперативных механик в видеоигру увеличивает глубину и разнообразие геймплея. В то время как одиночные игры предоставляют игроку подконтрольных компьютеру противников с ограниченным паттерном поведения, кооперативные игры вносят новый уникальный фактор, ведь игрок не может заранее просчитать действия другого игрока. Кроме того, игровой процесс ограничивает личный вклад игрока, создавая необходимость в социальном взаимодействии и распределении обязанностей.

Если говорить о кооперации в видеоиграх, то разделение на четкие роли может быть необязательным условием для создания потребности в объединении усилий. Такие игры и механики можно разделить по симметричности возможностей:

- Общие роли – каждый член коалиции обладает такими же возможностями, как любой другой игрок.

- Специальные роли – каждый игрок более эффективен в определённой области, но в целом не имеет уникальных возможностей относительно других игроков.
- Уникальные роли – каждый игрок может получить свою роль или персонажа, благодаря чему игроку доступны уникальные возможности по сравнению с другими игроками.

Помимо различий в симметрии возможностей, кооперативные механики можно также классифицировать по характеру взаимодействия между игроками (определяя, каким образом игроки влияют друг на друга и на игровой процесс):

1. Совместные механики;
2. Кооперативно-эмергентные механики;
3. Альтруистические механики;
4. Социально-экономические механики;
5. Комплементарные механики;
6. Механики одновременного управления;
7. Штрафующие механики;
8. Цепные механики.

Ниже представлена детальная характеристика каждого типа кооперативных механик с описанием их особенностей, принципов функционирования и примеров реализации в современных играх.

**Совместные механики**

В данную категорию входят механики, которые одновременно доступны всем игрокам, вне зависимости от их роли, но их одиночное использование может быть не выгодно или не способно привести к положительному исходу.

Пример такой механики – механика совместного взаимодействия в Warhammer: Vermintide 2[1]. В этой видеоигре игроки преодолевают локации, наполненные противниками и препятствиями. Некоторые из объектов, такие как тяжелые ворота, требуют времени для открытия, но при совместном взаимодействии с воротами несколькими игроками ворота открываются заметно быстрее.

[1] Список рассматриваемых видеоигр можно найти в разделе «Лудография».

То же самое относится к спасению союзника, если он был критически ранен и не способен двигаться.

**Кооперативно-эмергентные механики**

Разделение ролей – один из возможных вариантов создания потребности в кооперации. Кооперативно-эмергентные механики позволяют реализовать такую возможность в видеоиграх, так как обязывают игроков объединять ролевые способности для повышения эффективности или получения нового эффекта. Такие механики поощряют кооперацию и тактическое взаимодействие, делая игру более глубокой и стратегически насыщенной.

It Takes Two – кооперативная адвенчура[2], в которой игрокам выдают различные игровые инструменты, дополняющие друг друга при совместном использовании, создавая новый эффект.

Рис. 1. Демонстрация разного снаряжения персонажей в It Takes Two

Например, одному игроку выдают оружие, стреляющее липкой взрывоопасной смесью, в то время как другому выдают оружие, поджигающее взрывоопасную смесь (рис. 1). По отдельности они способны решать часть задач, но

[2]От англ. adventure – жанр приключенческих игр.

взрывать препятствия и противников для прохождения можно только при одновременном использовании обоих способностей.

**Альтруистические механики**

Следующая категория механик отличается характером воздействия – игрок не получает прямой выгоды от использования, но может принести пользу другому игроку или всей коалиции сразу. Примером могут являться способности лечения союзников, как, например, в серии видеоигр Left 4 Dead. Эта игра представляет большое число механик, которые обязывают игроков держаться вместе и помогать друг другу. В частности, таковой является механика «выведения из строя» при потери всего здоровья, когда игрок теряет способность двигаться и может быть возвращен в бой только с помощью союзника [13]. Аналогично игрок может быть обездвижен так называемыми «особыми зараженными», от которых игрок не может избавиться самостоятельно и может лишь ожидать помощи союзников (рис. 2).

Рис. 2. Особый зараженный удерживает игрока в Left 4 Dead 2

**Социально-экономические механики**

Данная категория механик затрагивает реализованный функционал видеоигры, не влияющий непосредственно на основной игровой процесс, но расширяющий возможности взаимодействия между игроками. К таким механикам можно

отнести возможности честной торговли и обмена между игроками, а также явного объединения игроков в коалиции (например, группа, гильдия, союз), позволяющие напрямую получить преимущество от объединения (распределение опыта, совместное прохождение подземелий, групповые усиления и т. д.). Все эти механики явно представлены в жанре видеоигр MMORPG, наиболее популярным представителем которой является World of Warcraft. Так, например, в игре имеются экономические механики в виде рынка, позволяющего игрокам продавать и покупать предметы за внутриигровую валюту, существует возможность проводить одновременный бартер при обоюдном согласии. Гильдии и группы, указанные выше, также можно найти в данной видеоигре.

**Комплементарные механики**

World of Warcraft является не только примером социально-экономических механик, но и примером игры с чётким разделением на роли, которые являются комплементарными друг к другу (рис. 3). В игре присутствует более 10 разных классов с уникальными умениями и механиками. При этом каждый класс в различной степени относится к трем основным игровым ролям:

- Танк – игрок, который является «щитом» группы и принимает основной удар на себя. Эта роль ориентирована на максимальную выживаемость, имея максимальные значения здоровья и защиты.
- Боец – роль, направленная на быстрое уничтожение противников. Получение урона может быть критичным для этой роли, потому что имеет мало здоровья и защиты.
- Лекарь – игрок, занимающийся восстановлением здоровья, утраченного другими игроками. Также имеет мало здоровья и не способен быстро справляться с противниками.

Такое разделение на роли позволяет создать зависимость одной роли от других – отсутствие танка не позволит другим ролям выжить при более сильных противниках, так как любой урон может стать смертельным; отсутствие бойца приведёт к затяжным боям, что, в свою очередь, приведёт к раннему истощению ресурсов; отсутствие лекаря не позволит восстанавливаться во время и между боями.


Рис. 3. Коалиция из 10 игроков в World of Warcraft

Конечно, некоторые классы могут относиться одновременно к нескольким ролям, но их эффективность заметно падает по сравнению с классами и чётким пониманием роли.

**Механики одновременного управления**

Следующий класс механик объединяет игроков за счёт выполнения различных действий, направленных на взаимодействие с объектом, управление которым требует наличия более одного игрока.

В видеоигре Sea of Thieves игроки управляют кораблём с большим числом органов управления. В то время как игроки имеют идентичные друг другу возможности, управление большим кораблём в одиночку становится невероятно сложным и, в общем, нерациональным решением. В данной игре выбор персонажа никак не влияет на возможности игрока, но при этом отдельные элементы корабля требуют непрерывного внимания:

- Штурвал – определяет направление движения корабля. Учитывая острова и препятствия, игрок всегда должен следить за тем, в какую сторону движется корабль.

- Паруса – определяют скорость движения корабля. Для достижения максимальной скорости игроки должны следить за ветром и поворачивать паруса на необходимый угол.
- Пушки – боевая мощь во время водных баталий. Без нанесения урона по противнику у него будет больше возможностей для сближения и ответной стрельбы из пушек.
- Корпус корабля – определяет плавучесть корабля. Оставленные от выстрелов пробоины в корпусе рано или поздно потопят корабль.

И это лишь часть обязанностей, которые необходимо выполнять игрокам одновременно. Конечно, игроки могут в любой момент изменить зону ответственности, но в любом случае каждая из этих систем требует одновременного внимания от игроков, с чем не справится одинокий игрок, особенно в условиях большого корабля.

**Штрафующие механики**

Данный тип механик направлен не столько на усиление игроков при кооперации, сколько на ослабление игроков-одиночек. Примером подобной механики является видеоигра Back 4 Blood. Игра по своей сути является идеологическим продолжением серии игр Left 4 Dead, поэтому игровой процесс этих двух игр схож. При этом в Back 4 Blood внедрён искусственный интеллект, который следит за сложностью и периодически направляет орду противников на игроков, если игра становится слишком простой. Кроме того, игра определяет расстояние между игроками и может отправить новую орду на одиночку, если один из игроков решит надолго отделиться от своих союзников, увеличивая вероятность одиночки погибнуть.

**Цепные механики**

Примером данного класса является видеоигра Portal 2 (рис. 4). Эта игра в жанре «головоломка» в кооперативном режиме требует от обоих игроков последовательного использования порталов в правильном порядке для перемещения персонажей и различных объектов по игровой сцене.

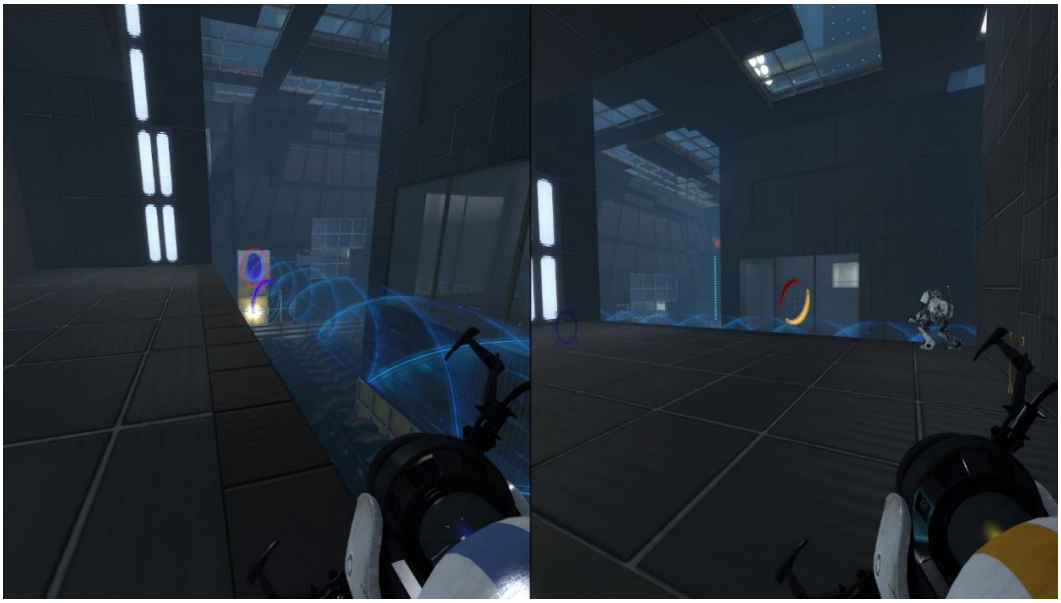


Рис. 4. Кооперативная компания Portal 2 в режиме разделения экрана

Игроки хоть и имеют одинаковые возможности, но должны взаимодействовать друг с другом, так как уровень не может быть пройдён в одиночку. В процессе игры требуется координация действий и общение игроков, поскольку ошибка одного может помешать действиям второго.

**Выводы**

Предложенная классификация кооперативных механик позволяет структурировать основные принципы взаимодействия игроков в многопользовательских играх. Она охватывает 8 ключевых типов кооперации (совместные, эмергентные кооперативные, альтруистические, социально-экономические, комплементарные, штрафующие, цепные механики и механики одновременного управления). Данные категории отражают различные подходы к организации взаимодействия, однако в нынешнем виде классификация остаётся предварительной и может быть расширена в будущем.

Одним из направлений для дальнейшей работы может являться детализация классификации с выделением подклассов внутри категорий. Например, в рамках альтруистических механик можно выделить «жертвенные механики», когда игрок обязан тратить один свой ресурс, чтобы повлиять на члена коалиции (например, потеря здоровья для лечения союзника персонажем Soraka в League of Legends).

Несмотря на свою предварительную форму, приведенная классификация играет важную роль в теоретизации видеоигр. Она не только помогает лучше понимать структуру кооперативного геймплея, но и предоставляет инструментарий для геймдизайнеров, позволяя им осознанно проектировать механики взаимодействия. В дальнейшем эта классификация может быть доработана с учётом новых форм кооперации и развития игровой индустрии.

**ЗАКЛЮЧЕНИЕ**

Кооперация является важнейшей частью игрового процесса во многих современных видеоиграх, оказывая влияние на мотивацию значительного числа игроков, в том числе повышая глубину геймплея. В настоящей работе рассмотрены ключевые аспекты кооперативных механик, на основании которых были выявлены их основные принципы и предложена структурированная классификация, включающая восемь типов взаимодействия.

Классификация кооперативных механик, не только позволяет структурировать существующие игровые механики, но и дает разработчикам инструментарий для более осознанного проектирования многопользовательских игр. В современных видеоиграх часто наблюдается комбинация различных кооперативных механик, что позволяет добиться глубины игрового процесса и сделать взаимодействие игроков более разнообразным и увлекательным. При этом разнообразие методов создания пространства для кооперации игроков достаточно велико, чтобы разработчики могли подобрать необходимый вариант.

Проанализированные исследования подтверждают, что кооперация в видеоиграх часто требует специальных игровых условий. Игроки, имея возможность действовать в одиночку, склонны избегать взаимодействия, если оно не является необходимым для успеха. Это подтверждается анализом механик принуждения к кооперации, которые заставляют игроков объединяться для достижения общей цели, например, при совместном решении головоломок в Portal 2 или прохождения Left 4 Dead, где одиночная игра невозможна.

Предложенная классификация механик может быть полезна как для геймдизайнеров, так и для исследователей игровой индустрии. Она позволяет

лучше понимать структуру кооперативного геймплея и оценивать влияние различных механик на игровой процесс. В будущем классификацию можно расширить за счёт более детального анализа подклассов механик. Кроме того, дальнейшие исследования могут быть направлены на изучение влияния кооперативных механик на поведение игроков, их вовлечённость и удовлетворенность игровым процессом. Важно также учитывать влияние внешних факторов, таких как уровень подготовки игроков, их социальные предпочтения и опыт в многопользовательских играх.

Таким образом, кооперативные механики являются важнейшим инструментом формирования игровых взаимодействий, способствуют усилению социального опыта и делают многопользовательские игры более интересными и захватывающими. Их осознанное проектирование и применение позволят разработчикам создавать увлекательные и глубоко проработанные игровые миры, стимулируя игроков к активному взаимодействию и командной работе.

**ЛУДОГРАФИЯ**

Fatshark (2018). Warhammer: Vermintide 2 [Action RPG] [Windows, PlayStation 4, PlayStation 5, Xbox One, Xbox Series X/S], Fatshark.

Hazelight Studios (2021). It Takes Two [Action-adventure, Puzzle] [Windows, PlayStation 4, PlayStation 5, Xbox One, Xbox Series X/S, Nintendo Switch], Electronic Arts.

Valve (2008). Left 4 Dead [First-person shooter] [Windows, Xbox 360, macOS], Valve Corporation.

Blizzard Entertainment (2004). World of Warcraft [MMORPG] [Windows, macOS], Blizzard Entertainment.

Rare (2018). Sea of Thieves [Action-adventure] [Windows, Xbox One, Xbox Series X/S], Xbox Game Studios.

Turtle Rock Studios (2021). Back 4 Blood [First-person shooter] [Windows, PlayStation 4, PlayStation 5, Xbox One, Xbox Series X/S], Warner Bros. Games.

Valve (2011). Portal 2 [Puzzle-platformer] [Windows, macOS, Linux, PlayStation 3, Xbox 360, Nintendo Switch], Valve Corporation.

Riot Games (2009). League of Legends [MOBA] [Windows, macOS], Riot Games.

**СПИСОК ЛИТЕРАТУРЫ**

1. STEAM Database. Most played Multiplayer games // STEAM Database. 2012. URL: https://steamdb.info/charts/?tagid=3859

2. *Bartle R.* Hearts, Clubs, Diamonds, Spades: Players Who Suit MUDs. 1996. P. 1–27.

3. *Yee N.* The gamer motivation profile: What we learned from 250,000 gamers // Proceedings of the 2016 Annual Symposium on Computer-Human Interaction in Play. 2016. P. 1–2. https://doi.org/10.1145/2967934.2967937

4. *Шубин А.В., Кугуракова В.В.* Проектирование инструментария для создания игрового процесса через систематизацию игровых механик // Электронные библиотеки. 2024. Т. 27. № 5. С. 774–795.

5. *Aarseth E., Smedstad S.M., Sunnana L.A.* Multidimensional typology of games // Proceedings of DiGRA 2003 Conference: Level Up. 2003.

6. *Elverdam C., Aarseth E.* Game Classification and Game Design: Construction Through Critical Analysis // Games and Culture. 2007. Vol. 2. No. 1. P. 3–22. https://doi.org/10.1177/1555412006286892

7. *Ali N., Tajuddin S., Bramantoro A.* Classification of Game Mechanics: A Brief Review // Proceedings of the 4th International Conference on Advances in Computational Science and Engineering. 2024. Vol. 1199. https://doi.org/10.1007/978-981-97-2977-7_19

8. *Oksanen K., Hämäläinen R.* Game Mechanics in the Design of a Collaborative 3D Serious Game // Simulation and Gaming. 2014. Vol. 45. No. 2. P. 255–278. https://doi.org/10.1177/1046878114530799

9. *Toups Dugas P.O., Kerne A., Hamilton W.* Game design principles for engaging cooperative play: Core mechanics and interfaces for non-mimetic simulation of fire emergency response // Proceedings of the 2009 ACM SIGGRAPH Symposium on Video Games. 2009. P. 71–78. https://doi.org/10.1145/1581073.1581085

10. *Nebel S., Schneider S., Beege M., Kolda F., Mackiewicz V., Rey G.D.* You cannot do this alone! Increasing task interdependence in cooperative educational videogames to encourage collaboration // Educational Technology Research and Development. 2017. Vol. 65. No. 4. P. 993–1014. https://doi.org/10.1007/s11423-017-9511-8

11. *Seif El-Nasr M. et al.* Understanding and evaluating cooperative games // Proceedings of the SIGCHI conference on human factors in computing systems. 2010. P. 253–262. https://doi.org/10.1145/1753326.1753363

12. *Rocha J.B., Mascarenhas S., Prada R.* Game Mechanics for Cooperative Games // ZON Digital Games. 2008. P. 1–10.

13. *Dubbelman T.* Narrative game mechanics // Interactive Storytelling: 9th International Conference on Interactive Digital Storytelling (ICIDS 2016). Springer International Publishing. 2016. P. 39–50. https://doi.org/10.1007/978-3-319-48279-8_4

## THE TYPOLOGY OF COOPERATIVE MECHANICS IN MULTIPLAYER VIDEOGAMES

**D. A. Khamatnurov[1] [0009-0005-7925-1164], A. V. Shubin[2] [0000-0002-6203-3268]**

[1, 2]*Institute of Information Technologies and Intelligent Systems, Kazan Federal University*

[1]khamatnurovd@gmail.com, [2]shubin.aleksey.kpfu@gmail.com

***Abstract***

Cooperative mechanics of multiplayer video games are typified. Cooperative game mechanics are the key elements of gameplay design that determine the ways of interaction between players in cooperative gameplay. Existing research in this area is analyzed and the main principles influencing the success of cooperative interaction are identified. Eight types of cooperative mechanics were identified and classified: cooperative, cooperative-emergent, altruistic, socio-economic, complementary, simultaneous control, penalty, and chain mechanics.

The authors concluded that successful cooperative mechanics are built on the principles of complementary roles, the combination of different abilities, and the need to distribute tasks for the sake of achieving a common goal. The proposed classification contributes to the systematization of knowledge in the field of game design and can be useful for developers of multiplayer video games. In the future, the classification can be refined and expanded taking into account the evolution of the industry and the emergence of new forms of cooperation.

***Keywords**: cooperative mechanics, multiplayer video games, game design, classification.*

**LUDOGRAPHY**

Fatshark (2018). Warhammer: Vermintide 2 [Action RPG] [Windows, PlayStation 4, PlayStation 5, Xbox One, Xbox Series X/S], Fatshark.

Hazelight Studios (2021). It Takes Two [Action-adventure, Puzzle] [Windows, PlayStation 4, PlayStation 5, Xbox One, Xbox Series X/S, Nintendo Switch], Electronic Arts.

Valve (2008). Left 4 Dead [First-person shooter] [Windows, Xbox 360, macOS], Valve Corporation.

Blizzard Entertainment (2004). World of Warcraft [MMORPG] [Windows, macOS], Blizzard Entertainment.

Rare (2018). Sea of Thieves [Action-adventure] [Windows, Xbox One, Xbox Series X/S], Xbox Game Studios.

Turtle Rock Studios (2021). Back 4 Blood [First-person shooter] [Windows, PlayStation 4, PlayStation 5, Xbox One, Xbox Series X/S], Warner Bros. Games.

Valve (2011). Portal 2 [Puzzle-platformer] [Windows, macOS, Linux, PlayStation 3, Xbox 360, Nintendo Switch], Valve Corporation.

Riot Games (2009). League of Legends [MOBA] [Windows, macOS], Riot Games.

**REFERENCES**

1. STEAM Database. Most played Multiplayer games // STEAM Database. 2012. URL: https://steamdb.info/charts/?tagid=3859

2. *Bartle R.* Hearts, Clubs, Diamonds, Spades: Players Who Suit MUDs. 1996. P. 1–27.

3. *Yee N.* The gamer motivation profile: What we learned from 250,000 gamers // Proceedings of the 2016 Annual Symposium on Computer-Human Interaction in Play. 2016. P. 1–2. https://doi.org/10.1145/2967934.2967937

4. *Shubin A.V., Kugurakova V.V.* Designing a tool for creating gameplay through thesystematization of game mechanics // Russian Digital Libraries. 2024. Vol. 27. No. 5. P. 774–795 (In Russian).

5. *Aarseth E., Smedstad S.M., Sunnana L.A.* Multidimensional typology of games // Proceedings of DiGRA 2003 Conference: Level Up. 2003.

6. *Elverdam C., Aarseth E.* Game Classification and Game Design: Construction Through Critical Analysis // Games and Culture. 2007. Vol. 2. No. 1. P. 3–22. https://doi.org/10.1177/1555412006286892

7. *Ali N., Tajuddin S., Bramantoro A.* Classification of Game Mechanics: A Brief Review // Proceedings of the 4th International Conference on Advances in Computational Science and Engineering. 2024. Vol. 1199. https://doi.org/10.1007/978-981-97-2977-7_19

8. *Oksanen K., Hämäläinen R.* Game Mechanics in the Design of a Collaborative 3D Serious Game // Simulation and Gaming. 2014. Vol. 45. No. 2. P. 255–278. https://doi.org/10.1177/1046878114530799

9. *Toups Dugas P.O., Kerne A., Hamilton W.* Game design principles for engaging cooperative play: Core mechanics and interfaces for non-mimetic simulation of fire emergency response // Proceedings of the 2009 ACM SIGGRAPH Symposium on Video Games. 2009. P. 71–78. https://doi.org/10.1145/1581073.1581085

10. *Nebel S., Schneider S., Beege M., Kolda F., Mackiewicz V., Rey G.D.* You cannot do this alone! Increasing task interdependence in cooperative educational videogames to encourage collaboration // Educational Technology Research and Development. 2017. Vol. 65. No. 4. P. 993–1014. https://doi.org/10.1007/s11423-017-9511-8

11. *Seif El-Nasr M. et al.* Understanding and evaluating cooperative games //Proceedings of the SIGCHI conference on human factors in computing systems. 2010. P. 253–262. https://doi.org/10.1145/1753326.1753363

12. *Rocha J.B., Mascarenhas S., Prada R.* Game Mechanics for Cooperative Games // ZON Digital Games. 2008. P. 1–10.

13. *Dubbelman T.* Narrative game mechanics // Interactive Storytelling: 9th International Conference on Interactive Digital Storytelling (ICIDS 2016). Springer International Publishing. 2016. P. 39–50. https://doi.org/10.1007/978-3-319-48279-8_4

**СВЕДЕНИЯ ОБ АВТОРАХ**

***ХАМАТНУРОВ Данил Азатович*** – магистрант кафедры индустрии разработки видеоигр Института информационных технологий

и интеллектуальных систем Казанского федерального университета. Сфера научных интересов: разработка видеоигр, игровой дизайн.

***Danil Azatovich KHAMATNUROV*** – Master's student of the Department of Video Game Development Industry at the Institute of Information Technologies and Intelligent Systems of Kazan Federal University. Research interests: video game development, game design.

email: khamatnurovd@gmail.com

ORCID: 0009-0005-7925-1164

***ШУБИН Алексей Витальевич*** – ассистент кафедры индустрии разработки видеоигр Института информационных технологий и интеллектуальных систем Казанского федерального университета, аспирант. Сфера научных интересов: разработка видеоигр, игровой дизайн, игровые механики.

***Aleksey Vitalevich SHUBIN*** – Assistant of the Department of Video Game Development Industry at the Institute of Information Technologies and Intelligent Systems of Kazan Federal University, postgraduate student. Research interests: video game development, game design, game mechanics.

email: shubin.aleksey.kpfu@gmail.com

ORCID: 0000-0002-6203-3268

*Материал поступил в редакцию 2 февраля 2025 года*

УДК 004.93

# ОНТОЛОГИЧЕСКАЯ МОДЕЛЬ ПОСТРОЕНИЯ КОНТУРОВ ОБЪЕКТОВ НА ИЗОБРАЖЕНИИ

**М. В. Бобырь[1] [0000-0002-5400-6817], В. П. Добрица[2] [0000-0001-7533-3684], А. С. Сизов[3] [0000-0001-9658-0318], А. А. Дородных[4] [0000-0003-0292-3127]**

[1, 2, 3]*Юго-Западный государственный университет, Курская обл., Курск, 305040 Россия*

[4]*Научно-исследовательский институт органических полупродуктов и красителей, Московская обл., Долгопрудный, 141701, Россия*

[1]maxbobyr@gmail.com, [2]dobritsa@mail.ru, [3]sizov@mail.ru, [4]alex.dorodnych@mail.ru

***Аннотация***

В настоящее время разработка онтологических моделей построения границ и их контуров по движущимся объектам в реальном времени или близком к нему является актуальной задачей. В связи с этим в статье представлена онтологическая модель реализации такого процесса. Рассмотрены основные алгоритмы детекции границ объектов на изображении, а также представлены программные коды для их реализации. Отмечено, что для распознавания контуров лучшим является алгоритм Канни. Вместе с этим определён и его серьезный недостаток, заключающий в том, что при незначительном движении объектов более 50% информации о контурах теряется.

***Ключевые слова****: границы объектов, контура объектов, Канни, Собель, Прюитт, Робертс, Лапласиан*

## ВВЕДЕНИЕ

Детектирование границ и контуров объектов на изображении в режиме реального времени и/или близком к нему является одной из ключевых задач компьютерного зрения и обработки изображений. Алгоритмы построения контуров применяются в различных областях, таких как: медицинская визуализация с целью анализа медицинских изображений, например рентгенография, МРТ и ис-

пользование в виртуальных биологических лабораториях [1]; робототехника с целью навигации и взаимодействия роботов с окружающей средой [2]; геоинформационные системы для анализа и визуализации географических данных [3].

В целом технология построения контуров включает две вычислительные процедуры: детектирование границ и построение контуров по найденным границам. Наиболее часто используемыми алгоритмами детекции контуров являются следующие модели. Детектор Канни наиболее часто используется в библиотеке OpenCV для обнаружения границ и включает несколько шагов: сглаживание изображения с помощью гауссова фильтра, вычисление градиентов, подавление немаксимальных значений и двойной порог для выделения потенциальных границ [4, 5]. Детектор Собеля применяет сверточные матрицы для вычисления градиентов интенсивности изображения в горизонтальном и вертикальном направлениях. Полученные градиенты применяются для вычисления угла градиента и построения границ объектов по ним [6, 7]. Детектор Прюитта аналогичен алгоритму Собеля, но использует другие коэффициенты в сверточных матрицах для вычисления градиентов и их углов [8, 9]. Детектор Робертса основан на двухмерных диагональных сверточных масках, за счёт чего является наиболее быстродействующим фильтром [10, 11]. Детектор Лапласиана использует вторые производные для вычисления изменения интенсивности пикселей на изображении с целью определения точек, в которых изменение градиентов очень резкое [12, 13].

К алгоритмам построения контуров относятся следующие модели. Алгоритм Сузуки–Абе является одним из наиболее распространённых алгоритмов выделения контуров, он реализован в OpenCV [14, 15]. Алгоритм Дугласа–Пекера упрощает процесс построения полилиний, так как запоминает только координаты пикселей, в которых изгибается контур, тем самым уменьшая количество вершин в распознанном контуре объекта [16, 17]. Алгоритм Чана–Весе используется для сегментации изображений и выделения контуров путем их минимизации [18, 19].

Следует отметить, что методы обнаружения границ и выделения контуров играют важную роль в задачах компьютерной обработки изображений. Выбор конкретного метода зависит от задачи и характеристик изображений, таких как уровень шума и сложность объектов.

## МЕТОДЫ И МАТЕРИАЛЫ

Онтологическая модель построения границ и контуров объектов (ОМПГКО) на изображении в реальном времени или близком к нему представляет собой кортеж, состоящий из набора понятий, их атрибутов и связей между ними:

$$O = \left\langle \underset{n=1}{\overset{5}{O_к}}, \underset{a=1}{\overset{5}{O_a}}, \underset{c=1}{\overset{4}{O_c}} \right\rangle, \quad (1)$$

где $O_к$ – онтологическая модель классов объектов; $O_a$ – онтологическая модель свойств атрибутов; $O_c$ – связей между понятиями и атрибутами.

ОМПГКО в виде семантической сети представлена на рис. 1.

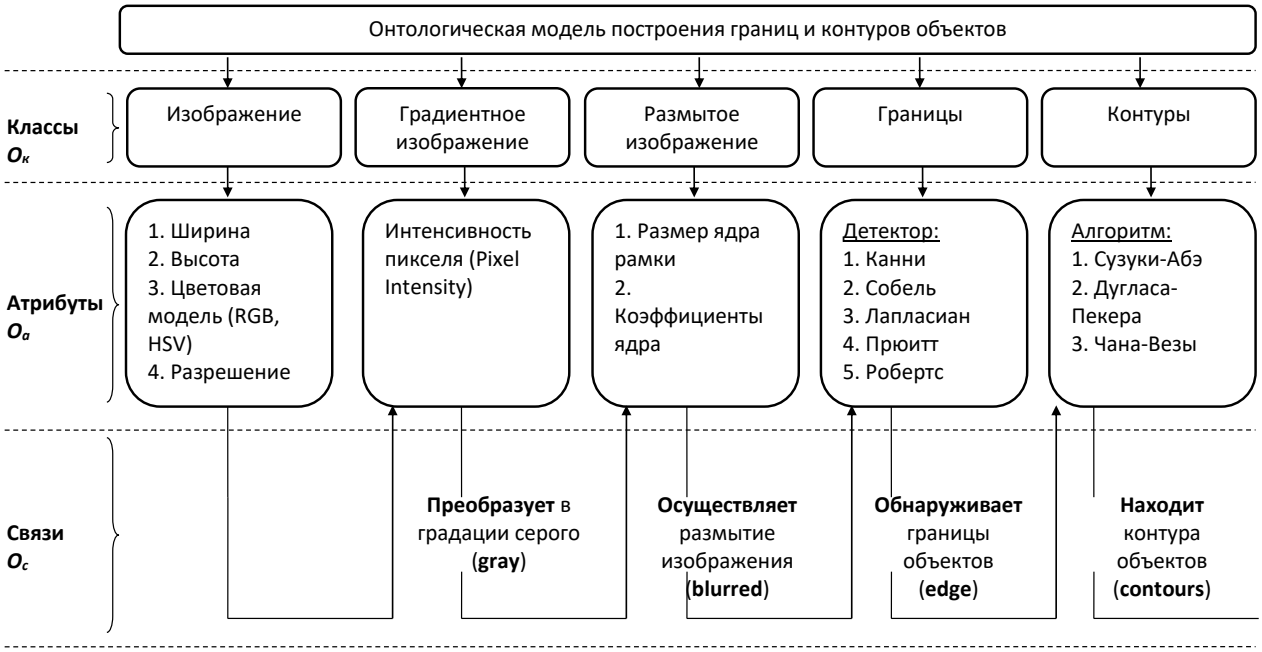



Рис. 1. Онтологическая модель построения границ объектов

Процесс обработки изображения с помощью ОМПГКО осуществляют в следующей последовательности. Сначала исходное изображение преобразуют в градиенты серого с целью уменьшения количества каналов с четырех (RGBA) до одного. Затем осуществляют размытие изображения с помощью одного из фильтров (линейного, гауссова, билатерального) с целью уменьшения шума на изображении и более четкого выделения границ на нём. Далее с помощью одного из методов нахождения границ (например, Канни, Собеля, Приютта, Робертса или

Лапласа) выделяют границы на изображении. После этого, используя один из алгоритмов обнаружения контуров (например, Сузуки–Абе, Дугласа–Пекера, Чана–Весы), строят набор контуров, представляющий замкнутые линии, которые описывают формы объектов на изображении.

Одним из эффективных алгоритмов построения контуров является модель Сузуки–Абе, содержащая следующую последовательность вычислительных операций.

1. **Бинаризация изображения** – преобразование изображения в бинарный формат, где пиксели объекта помечают как 1 (белые), а фон – как 0 (черные).

2. **Инициализация**. Алгоритм начинает работу поиском первого белого пикселя (запись координат стартового пиксель-объекта), причём с целью удаления ложных границ рекомендуется пользоваться логическим фильтром. Алгоритм начинает работу с поиска первого белого пикселя (записи его координат как стартовой точки контура). Для исключения ложных границ рекомендуется использовать фильтр связности: если сумма белых пикселей (значение 1) восьми соседних пикселей относительного центрального пикселя в окне 3×3 меньше или равна 1, этот (центральный) пиксель считается шумом, пропускается, и осуществляется поиск следующего белого пикселя. С целью повышения быстродействия на данном шаге рекомендуется пользоваться моделью оптимизации подсчёта суммы значений в рамке 3×3 [20].

3. **Поиск контуров.** Алгоритм обходит пиксели объекта по часовой стрелке, сохраняя координаты точек, которые образуют замкнутую линию вокруг объекта с выделенными границами. Для реализации этого способа рекомендуется пользоваться рекурсивным алгоритмом закрашивания выделенных областей [21].

4. **Сохранение контура.** Как только замкнутый контур найден, алгоритм дошёл до стартового пиксель-объекта (Шаг 2), найденные координаты пикселей сохраняются, тем самым формируя распознанный контур.

5. **Поиск всех объектов**. Алгоритм продолжает выполнять шаги 2–4, пока все контуры распознанных объектов не будут найдены на изображении.

Таким образом, ОМПГКО возвращает список всех найденных контуров, каждый из которых представляет собой замкнутую линию вокруг распознанных объектов на изображении.

## РЕЗУЛЬТАТЫ ЭКСПЕРИМЕНТАЛЬНЫХ ИССЛЕДОВАНИЙ

Для тестирования ОМПГКО было разработано специализированное программное обеспечение на языке высокого уровня Python в среде Anaconda 3. Программный код алгоритмов построения контуров с помощью различных операторов детектирования границ представлен на листингах 1–5.

```
import cv2
import numpy as np

cap = cv2.VideoCapture(0)// захват видео с веб-камеры
cv2.namedWindow("frame")

while True:
   ret, frame = cap.read()
   gray = cv2.cvtColor(frame,cv2.COLOR_BGR2GRAY) // преобразование в серое
   blurred = cv2.GaussianBlur(gray, (5, 5), 0)             // размытие
   edge = cv2.Canny(blurred,100,200)// детектор Канни
contours,h = cv2.findContours(edge,cv2.RETR_TREE,cv2.CHAIN_APPROX_SIMPLE)
   cv2.drawContours(frame,contours,–1,(0,0,255),3)   // нахождение контуров
   cv2.imshow("frame", frame)                                  // вывод на экран
   cv2.imshow("edge", edge)

   k = cv2.waitKey(1) & 0xFF
   if k == 27:
      break
cap.release()
cv2.destroyAllWindows()
```

Листинг 1. Детектирование границ с помощью оператора Канни

```
sobelx = cv2.Sobel(blurred, cv2.CV_64F, 1, 0, ksize=3)

   sobely = cv2.Sobel(blurred, cv2.CV_64F, 0, 1, ksize=3)

   edge = np.sqrt(sobelx**2 + sobely**2)

edge = np.uint8(edges)
```

Листинг 2. Фрагмент программного кода для детектирования границ с помощью оператора Собеля

```
laplacian = cv2.Laplacian(gray, cv2.CV_64F, ksize=3)
    edge = np.uint8(np.absolute(laplacian))
```

Листинг 3. Фрагмент программного кода для детектирования границ с помощью Лапласиана

```
kernel_x = np.array([[1, 0, –1], [1, 0, –1], [1, 0, –1]])
    kernel_y = np.array([[1, 1, 1], [0, 0, 0], [–1, –1, –1]])
    prewitt_x = cv2.filter2D(blurred, –1, kernel_x)
    prewitt_y = cv2.filter2D(blurred, –1, kernel_y)
edge = (prewitt_x + prewitt_y)
```

Листинг 4. Фрагмент программного кода для детектирования границ с помощью оператора Прюитта

```
kernel_x = np.array([[1, 0], [0, –1]])
    kernel_y = np.array([[0, 1], [–1, 0]])
    roberts_x = cv2.filter2D(gray, –1, kernel_x)
    roberts_y = cv2.filter2D(gray, –1, kernel_y)
edge = (roberts_x + roberts_y)
```

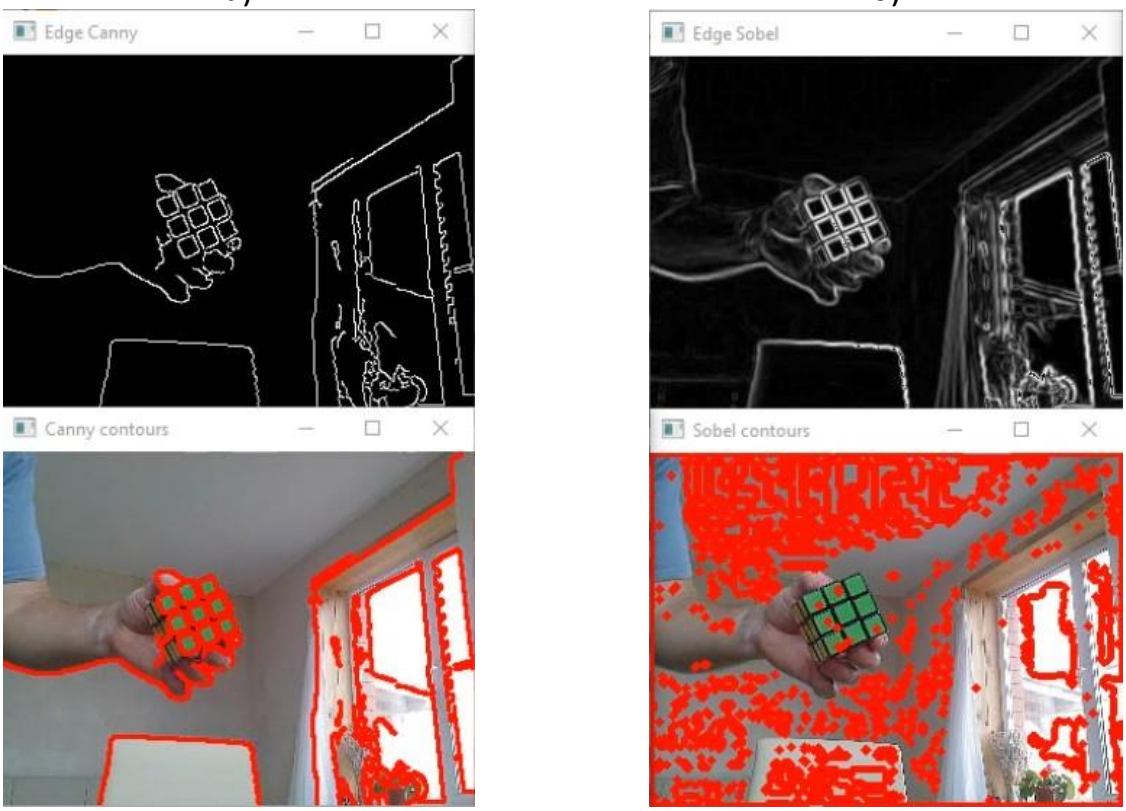
Листинг 5. Фрагмент программного кода для детектирования границ с помощью оператора Робертса

В первом эксперименте осуществлялся визуальный контроль качества детектирования границ и построения по ним контуров. В ходе эксперимента вращался кубик-рубик и программы должны были построить по девять контуров на каждой из его граней. Результат работы алгоритмов представлен на рис. 2.

В ходе первого эксперимента использовался видеоряд, захваченный с веб-камеры, причем для каждого из алгоритмов (листинги 1–5) строились два изображения. На верхнем изображении представлены результаты получения границ объектов с помощью одного из операторов: Канни, Собеля, Лапласиан, Прюитта, Робертса. На нижнем изображении показаны выделенные контуры, найденные

по границам объектов. Анализ результатов работы перечисленных алгоритмов, представленных на рис. 2, показал, что лучшее построение контуров осуществляется при применении алгоритма Канни. При использовании остальных фильтров на исходном изображении появляется много шума. Укажем наиболее предпочтительные детекторы границ в порядке их убывания: Канни → Собель → Лапласиан → Прюитт → Робертс.

а) б)



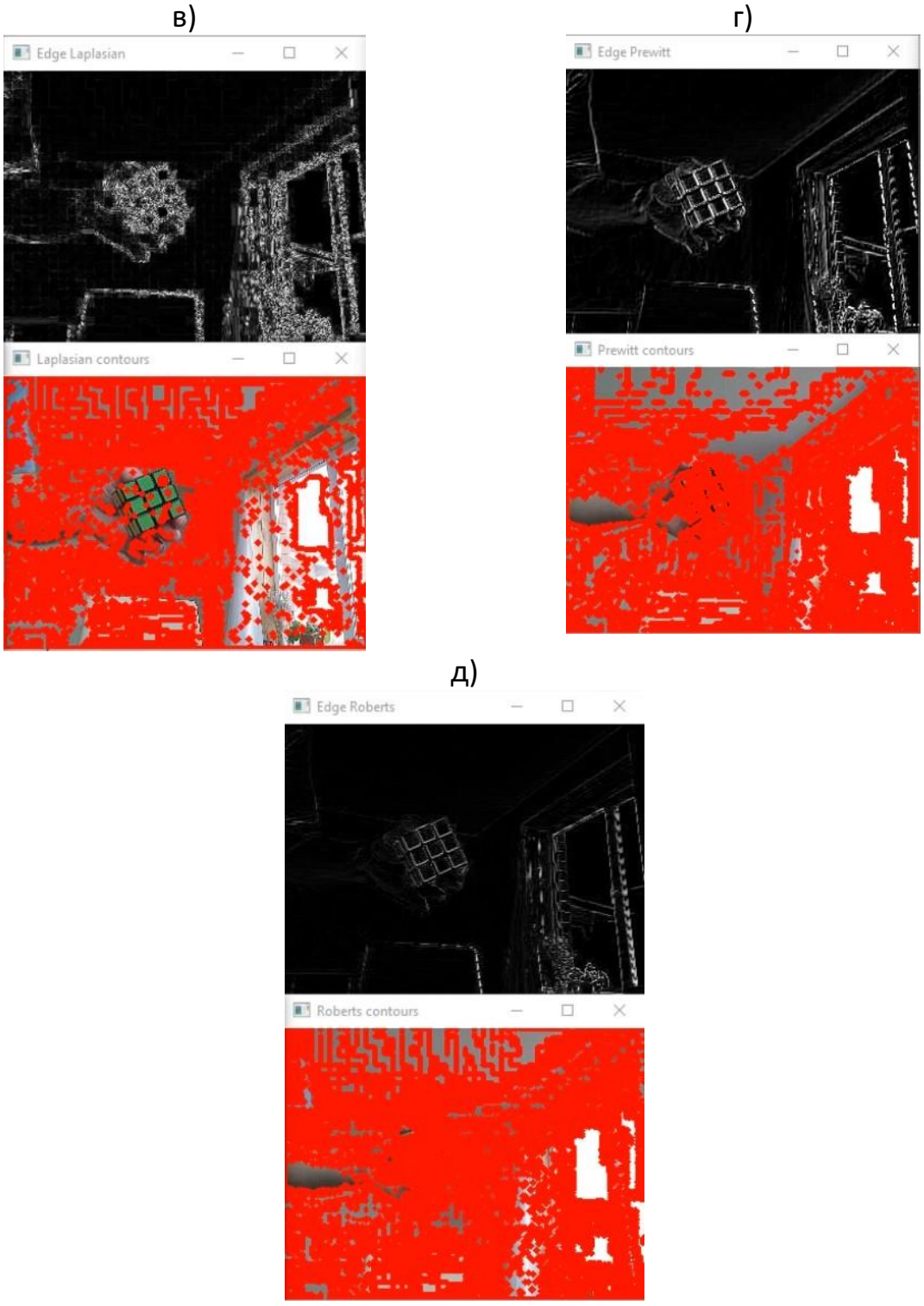
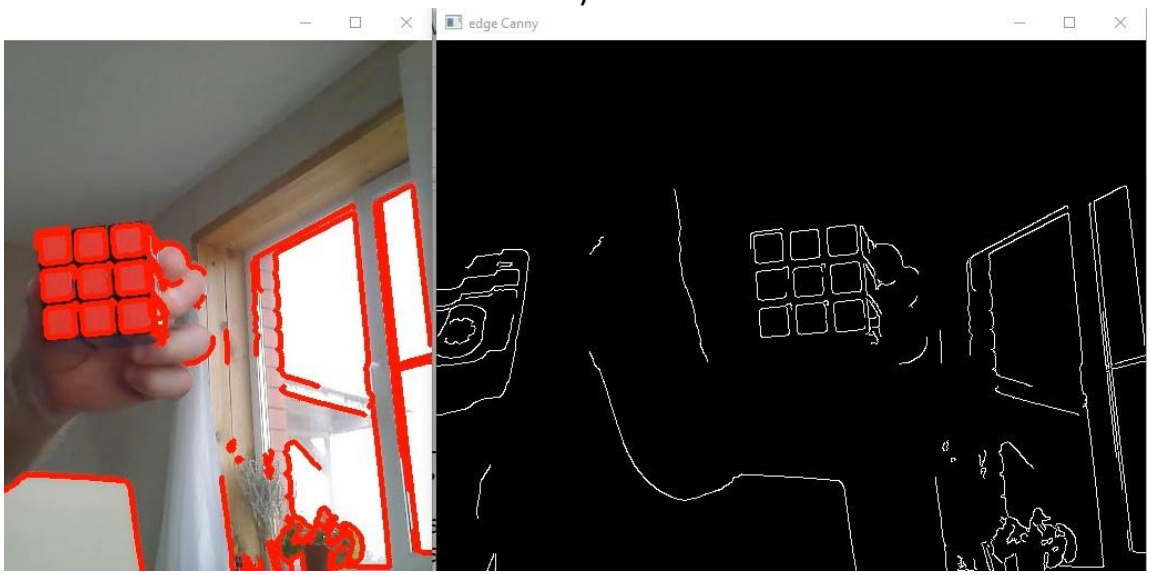



Рис. 2. Детектирование контуров на основе ОМПГКО, причем на верхнем изображении представлен один из детекторов, а на нижнем – выделенные контуры на исходном изображении: а – детектор Канни; б – детектор Собеля; в – детектор Лапласа; г – детектор Прюитта; а – детектор Робертса

а)




б)

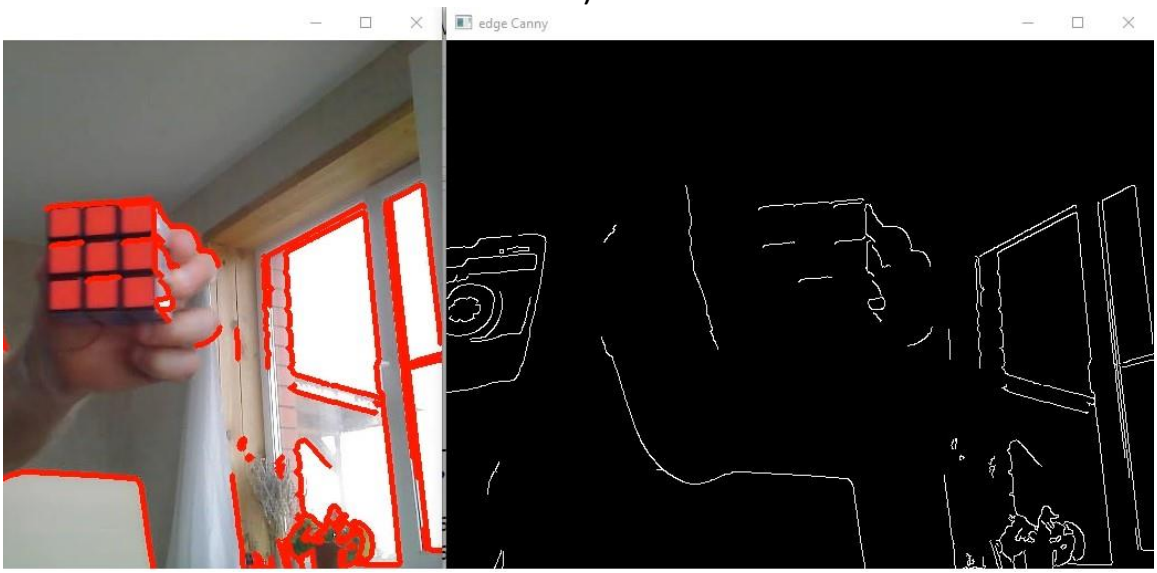



Рис. 3. Детектирование границ объектов с помощью детектора Канни:

а – без движения; б – в движении

Следует отметить, что при определении границ на всех изображениях (кроме детектора Канни) границы представлены градациями серого и значения их интенсивности находятся в диапазоне от 0 до 255, в то время как на выходе алгоритма Канни — бинарное изображение.

В ходе второго эксперимента также использовался видеоряд, захваченный с веб-камеры, и для каждого из алгоритмов (листинги 1–5) анализировалось качество выделения границ при движении объектов, причём использовались два наилучших фильтра: Канни и Собеля. Результаты представлены на рис. 3 и 4.

В ходе второго эксперимента было установлено, что без движения детектор Канни при распознавании границ кубика-рубика достаточно точно их определяет (рис. 3 а), однако при относительно небольшом движении теряет более 50% контуров (рис. 3 б).

а)


б)


Рис. 4. Детектирование границ объектов с помощью детектора Собеля:
а – без движения; б – в движении

В то же время оператор Собеля недостаточно корректно выделяет контуры как на статичном (рис. 4 а), так и на подвижном (рис. 5 а) изображениях, при этом также происходит смазывание границ объекта. Для повышения качества распознавания границ объектов рекомендуется использовать нечеткую логику [22, 23], включающую новый метод детекции краёв [24].

**ЗАКЛЮЧЕНИЕ**

Таким образом, рассмотрены пять методов детектирования границ объектов на изображении и построение по ним контуров на основе детекторов: Канни, Собеля, Лапласиана, Прюитта и Робертса. Результаты экспериментального исследования, в ходе которого проводилось детектирование границ и контуров на изображениях, показали превосходство детектора Канни по отношению к другим алгоритмам. Однако детектор Канни имеет серьезный недостаток, заключающий в том, что при движении более 50% контуров, видимых на статичном изображении, теряются. Результаты эксперимента также показали, что точность распознавания границ и контуров объектов с использованием алгоритма Канни напрямую зависит от скорости движения объектов – при очень высокой скорости потери при детектировании контуров доходят до 90%.

**Благодарности**

Работа выполнена при финансовой поддержке Государственного задания (проект № 0851-2020-0032).

**СПИСОК ЛИТЕРАТУРЫ**

1. *Абрамов В.Д., Кугуракова В.В., Ризванов А.А. [и др.]* Виртуальные лаборатории как средство обучения биомедицинским технологиям // Электронные библиотеки. 2016. Т. 19, № 3. С. 129–148.

2. *Sanz P.* Robotics: Modeling, Planning, and Control. IEEE Robotics and Automation Magazine. 2009. https://doi.org/10.1109/MRA.2009.934833

3. *Reddy G.P.O.* Geographic Information System: Principles and Applications. 2018. P. 45–62. https://doi.org/10.1007/978-3-319-78711-4_3

4. *Canny J.* A Computational Approach to Edge Detection. IEEE Transactions on Pattern Analysis and Machine Intelligence. 1986. PAMI-8(6). P. 679–698. https://doi.org/10.1109/TPAMI.1986.4767851

5. *Qin X*. A modified Canny edge detector based on weighted least squares. Computational Statistics. (2021). Vol. 36. No. 1. P. 641–659. https://doi.org/10.1007/s00180-020-01017-8.

6. *Бо Я., Неусыпин К.А.* Совершенствование метода предварительной обработки звездной карты с использованием оператора Собеля // Автоматизация. Современные технологии. 2023. Т. 77. № 7. С. 308–314. https://doi.org/10.36652/0869-4931-2023-77-7-308-314.

7. *Tian R., Sun G., Liu X., Zheng B.* Sobel edge detection based on weighted nuclear norm minimization image denoising // Electronics. 2021. Vol. 10. No. 6. P. 1–15. https://doi.org/10.3390/electronics10060655

8. *Adlakha D., Tanwar R.* Analytical Comparison between Sobel and Prewitt Edge Detection Techniques. International Journal of Scientific & Engineering Research. 2016. Vol. 7. No. 1. P. 1482–1485.

9. *Karthick C.N., Nirmala P.* Smart Edge Detection Technique in X-ray Images for Improving PSNR using Prewitt Edge Detection Algorithm with Gaussian Filter in Comparison with Laplacian Algorithm // Cardiometry. 2023. №25. P.1758–1762. https://doi.org/10.18137/cardiometry.2022.25.17581762

10. *Gong H.X., Hao L.* Roberts edge detection algorithm based on GPU. Journal of Chemical and Pharmaceutical Research. 2014. Vol. 6. No. 7. P. 1308–1314.

11. *Utama K.M.R.A., Umar R., Yudhana A.* Edge detection comparative analysis using Roberts, Sobel, Prewitt, and Canny methods// Jurnal Teknologi Dan Sistem Komputer. 2022. Vol. 10. No. 2. P. 67–71. https://doi.org/10.14710/jtsiskom.2021.14209

12. *Qu Y., Tang W., Su, Cheng S.* Web Inspection Algorithm of Low Contrast Paper Defects Using Gabor Filter // In Lecture Notes in Electrical Engineering. 2022. Vol. 920. P. 371–378. Springer Science and Business Media Deutschland GmbH. https://doi.org/10.1007/978-981-19-3927-3_36

13. *Darwis D., Fernando Y., Trisnawati F., Marzuki D.H., Setiawansyah, S.* Comparison of edge detection methods using Roberts and Laplacian operators on mango leaf objects. Barekeng // Jurnal Ilmu Matematika Dan Terapan. 2023. Vol. 17. No. 3. P. 1815–1824. https://doi.org/10.30598/barekengvol17iss3pp1815-1824

14. *Suzuki S., Abe K.* Topological structural analysis of digitized binary images by border following // Computer Vision, Graphics and Image Processing. 1985. Vol. 30. P. 32–46. https://doi.org/10.1016/0734-189X(85)90016-7

15. *Mei L., Wang C., Zhao Y., Wei W., Li Y.* Real-time detection method of landmark in UAV autonomous landing // Systems Engineering and Electronics. 2019. Vol. 41. No. 10. P. 2157–2162. https://doi.org/10.3969/j.issn.1001-506X.2019.10.01

16. *Zhou Z., Zhang Y., Yuan X., Wang H.* Compressing AIS Trajectory Data Based on the Multi-Objective Peak Douglas-Peucker Algorithm // IEEE Access. 2023. No. 11. P. 6802–6821. https://doi.org/10.1109/ACCESS.2023.3234121

17. *Liu J., Li H., Yang Z., Wu K., Liu Y., Liu R.W.* Adaptive Douglas–Peucker Algorithm with Automatic Thresholding for AIS-Based Vessel Trajectory Compression // IEEE Access. 2019. No. 7. P. 150677–150692.
https://doi.org/10.1109/ACCESS.2019.2947111

18. *Gunasekara S.R., Kaldera K., Dissanayake M.B.* A Systematic Approach for MRI Brain Tumor Localization and Segmentation Using Deep Learning and Active Contouring // Journal of Healthcare Engineering. 2021.
https://doi.org/10.1155/2021/6695108

19. *Çataloluk H., Çelebi F.V.* A novel hybrid model for two-phase image segmentation: GSA based Chan–Vese algorithm // Engineering Applications of Artificial Intelligence. 2018. No. 73. P. 22–30. https://doi.org/10.1016/j.engappai.2018.04.027

20. *Бобырь М.В., Храпова Н.И., Супрунова О.Г., Дородных А.А.* Рекурсивный алгоритм закрашивания областей распознанных объектов // Известия Юго-Западного государственного университета. 2023. Т. 27. № 1. С. 126–139.

21. *Бобырь М.В., Емельянов С.Г., Милостная Н.А.* Оптимизация числа проходов в задаче логической фильтрации изображений // Искусственный интеллект и принятие решений. 2023. № 2. С. 98–107.
https://doi.org/10.14357/20718594230208.

22. *Бобырь М.В., Кулабухов С.А.* Моделирование процесса управления температурным режимом в зоне резания на основе нечеткой логики // Проблемы машиностроения и надежности машин. 2017. № 3. С. 76–82.
https://doi.org/10.3103/S1052618817030049.

23. *Бобырь М.В., Храпова Н.И.* Двухуровневая информационно-аналитическая система управления интеллектуальным светофором// Электронные библиотеки. 2024. №5. С. 696–717.

24. *Бобырь М.В., Архипов А.Е., Горбачев С.В.* Нечетко-логические методы в задаче детектирования границ объектов// Информатика и автоматизация. 2022. Т. 21, № 2.С. 376–404.

## ONTOLOGICAL MODEL FOR CREATING OBJECT CONTOURS IN AN IMAGE

**M. V. Bobyr[1] [0000-0002-5400-6817], V. P. Dobritsa[2] [0000-0001-7533-3684],**
**A. S. Sizov[3] [0000-0001-9658-0318], A. A. Dorodnykh[4] [0000-0003-0292-3127]**

[1, 2, 3]Southwest State University, Kursk, Russia
[4]Research Institute of Organic Semi-Products and Dyes, Dolgoprudny, Russia

[1]maxbobyr@gmail.com, [2]dobritsa@mail.ru, [3]sizov@mail.ru, [4]alex.dorodnych@mail.ru

***Abstract***

Now days, the development of ontological models for creating edges and their contours for moving objects in real time or close to it is an urgent task. An ontological model for implementing this process is shown in the article. The main algorithms for detecting object edges and constructing contours in an image and program codes for their implementation are considered in the article. It is noted that the Canny algorithm is the best for recognizing edges. At the same time, its serious drawback is determined, which consists in the fact that with insignificant movement of objects, more than 50% of information about the contours is lost.

***Keywords****: object edges, object contours, Canny, Sobel, Prewitt, Roberts, Laplacian.*

**REFERENCES**

1. *Abramov V.D., Kugurakova V.V., Rizvanov A.A. [et al.]* Virtual laboratories as a means of teaching biomedical technologies // Electronic libraries. 2016. Vol. 19. No. 3. P. 129–148.

2. *Sanz P.* Robotics: Modeling, Planning, and Control. IEEE Robotics and Automation Magazine. 2009. https://doi.org/10.1109/MRA.2009.934833

3. *Reddy G.P.O.* Geographic Information System: Principles and Applications. 2018. P. 45–62. https://doi.org/10.1007/978-3-319-78711-4_3

4. *Canny J.* A Computational Approach to Edge Detection. IEEE Transactions on Pattern Analysis and Machine Intelligence. 1986. PAMI-8(6). P. 679–698. https://doi.org/10.1109/TPAMI.1986.4767851

5. *Qin X*. A modified Canny edge detector based on weighted least squares. Computational Statistics. (2021). Vol. 36. No. 1. P. 641–659. https://doi.org/10.1007/s00180-020-01017-8.

6. *Bo Ya., Neusypin K.A.* Improving the method of preprocessing a star map using the Sobel operator // Automation. Modern technologies. 2023. Vol. 77. No. 7. P. 308–314. https://doi.org/10.36652/0869-4931-2023-77-7-308-314.

7. *Tian R., Sun G., Liu X., Zheng B.* Sobel edge detection based on weighted nuclear norm minimization image denoising // Electronics. 2021. Vol. 10. No. 6. P. 1–15. https://doi.org/10.3390/electronics10060655

8. *Adlakha D., Tanwar R.* Analytical Comparison between Sobel and Prewitt Edge Detection Techniques. International Journal of Scientific & Engineering Research. 2016. Vol. 7. No. 1. P. 1482–1485.

9. *Karthick C.N., Nirmala P.* Smart Edge Detection Technique in X-ray Images for Improving PSNR using Prewitt Edge Detection Algorithm with Gaussian Filter in Comparison with Laplacian Algorithm // Cardiometry. 2023. No. 25. P. 1758–1762. https://doi.org/10.18137/cardiometry.2022.25.17581762

10. *Gong H.X., Hao L.* Roberts edge detection algorithm based on GPU. Journal of Chemical and Pharmaceutical Research. 2014. Vol. 6. No. 7. P. 1308–1314.

11. *Utama K.M.R.A., Umar R., Yudhana A.* Edge detection comparative analysis using Roberts, Sobel, Prewitt, and Canny methods// Jurnal Teknologi Dan Sistem Komputer. 2022. Vol. 10. No. 2. P. 67–71. https://doi.org/10.14710/jtsiskom.2021.14209

12. *Qu Y., Tang W., Su, Cheng S.* Web Inspection Algorithm of Low Contrast Paper Defects Using Gabor Filter// In Lecture Notes in Electrical Engineering. 2022. Vol. 920. P. 371–378. Springer Science and Business Media Deutschland GmbH. https://doi.org/10.1007/978-981-19-3927-3_36

13. *Darwis D., Fernando Y., Trisnawati F., Marzuki D.H., Setiawansyah, S.* Comparison of edge detection methods using Roberts and Laplacian operators on mango leaf objects. Barekeng // Jurnal Ilmu Matematika Dan Terapan. 2023. Vol. 17. No. 3. P. 1815–1824. https://doi.org/10.30598/barekengvol17iss3pp1815-1824

14. *Suzuki S., Abe K.* Topological structural analysis of digitized binary images by border following // Computer Vision, Graphics and Image Processing. 1985. Vol. 30. P. 32–46. https://doi.org/10.1016/0734-189X(85)90016-7

15. *Mei L., Wang C., Zhao Y., Wei W., Li Y.* Real-time detection method of landmark in UAV autonomous landing // Systems Engineering and Electronics. 2019. Vol. 41. No. 10. P. 2157–2162. https://doi.org/10.3969/j.issn.1001-506X.2019.10.01

16. *Zhou Z., Zhang Y., Yuan X., Wang H.* Compressing AIS Trajectory Data Based on the Multi-Objective Peak Douglas-Peucker Algorithm // IEEE Access. 2023. No. 11. P. 6802–6821. https://doi.org/10.1109/ACCESS.2023.3234121

17. *Liu J., Li H., Yang Z., Wu K., Liu Y., Liu R.W.* Adaptive Douglas–Peucker Algorithm with Automatic Thresholding for AIS-Based Vessel Trajectory Compression // IEEE Access. 2019. No. 7. P. 150677–150692.
https://doi.org/10.1109/ACCESS.2019.2947111

18. *Gunasekara S.R., Kaldera K., Dissanayake M.B.* A Systematic Approach for MRI Brain Tumor Localization and Segmentation Using Deep Learning and Active Contouring // Journal of Healthcare Engineering. 2021.
https://doi.org/10.1155/2021/6695108

19. *Çataloluk H., Çelebi F.V.* A novel hybrid model for two-phase image segmentation: GSA based Chan–Vese algorithm // Engineering Applications of Artificial Intelligence. 2018. No. 73. P. 22–30. https://doi.org/10.1016/j.engappai.2018.04.027

20. *Bobyr M.V., Khrapova N.I., Suprunova O.G., Dorodnykh A.A.* The recursive algorithm for filling areas of recognized objects // Izvestiya Yugo-Zapadnogo gosudarstvennogo universiteta. 2023. T. 27, № 1. S. 126-139.

21. *Bobyr M.V., Emelyanov S.G., Milostnaya N.A.* Optimization of the number of passes in the problem of logical image filtering // Artificial Intelligence and Decision Making. 2023. № 2. S. 98–107. https://doi.org/10.14357/20718594230208.

22. *Bobyr M.V., Kulabukhov S.A.* Simulation of control of temperature mode in cutting area on the basis of fuzzy logic // Journal of Machinery Manufacture and Reliability. 2017. Vol. 46. P. 288–295. https://doi.org/10.3103/S1052618817030049.

23. *Bobyr M.V., Khrapova N.I.* Two-level information and analytical control system for intelligent traffic lights // Electroniclibraries. 2024. №5. S. 696–717.

24. *Bobyr M.V., Arkhipov A.E., Gorbachev S.V.* Fuzzy logic methods in the problem of detecting object boundaries // Informatics and Automation. 2022. Vol. 21, No. 2. P. 376–404.

## СВЕДЕНИЯ ОБ АВТОРАХ

***БОБЫРЬ Максим Владимирович***. Доктор технических наук, профессор. Профессор кафедры «Программная инженерия», ФГБОУ ВО «Юго-Западный государственный университет». Области исследований: интеллектуальные системы управления, адаптивные нейро-нечеткие системы вывода, распознавание и обработка изображения, робототехнические системы.

***Maxim V. BOBYR.*** Doctor of technical sciences, professor. Professor of the Department of Software Engineering, Southwest State University. Research areas: intelligent control systems, adaptive neuro-fuzzy inference systems, image recognition and processing, robotic systems.

email: maxbobyr@gmail.com

ORCID: 0000-0002-5400-6817

***ДОБРИЦА Вячеслав Порфирьевич***. Доктор технических наук, профессор. Профессор кафедры «Информационная безопасность», ФГБОУ ВО «Юго-Западный государственный университет». Области исследований: системы искусственного интеллекта, нейронные сети для обнаружения и локализации кибератак.

***Vyacheslav P. DOBRITSA.*** Doctor of technical sciences, professor. Professor of the Department of Software Engineering, Southwest State University. Research areas: artificial intelligence systems, neural networks for detection and localization of cyber attacks.

email: dobritsa@mail.ru

ORCID: 0000-0001-7533-3684

***СИЗОВ Александр Семенович***. Доктор технических наук, профессор. Профессор кафедры «Программная инженерия», ФГБОУ ВО «Юго-Западный государственный университет». Области исследований: нейронные сети и нечёткие системы, интеллектуальные системы управления сложными техническими объектами.

***Alexander S. SIZOV.*** Doctor of technical sciences, professor. Professor of the Department of Software Engineering, Southwest State University. Research areas: neural networks and fuzzy systems, intelligent control systems for complex technical objects.

email: sizov@mail.ru

ORCID: 0000-0001-9658-0318

***ДОРОДНЫХ Александр Алексеевич***. АО «НИОПИК (Научно-исследовательский институт органических полупродуктов и красителей)». Область исследования: вычислительное моделирование для интерфейсов мозг–компьютер, глаз–компьютер и глаз–мозг–компьютер, решение задач в режиме, близком к реальному времени. Системы технического зрения для распознавания объектов с использованием нечеткой логики, нейронных сетей.

***Alexander A. DORODNYKH.*** JSC "NIOPIK (Research Institute of Organic Semi-finished Products and Dyes)". Research area: Research area: near-real-time problem solving and computational modeling for brain–computer, eye–computer and eye–brain–computer interfaces. Machine vision systems for object recognition using fuzzy logic, neural networks.

email: alex.dorodnych@mail.ru

ORCID: 0000-0003-0292-3127

*Материал поступил в редакцию 25 ноября 2024 года*

УДК 624.04:531.39.3

# ПРОГИБ И ПЕРВЫЕ СОБСТВЕННЫЕ ЧАСТОТЫ КОЛЕБАНИЙ РЕГУЛЯРНОЙ АРОЧНОЙ ФЕРМЫ

**М. Н. Кирсанов**[0000-0002-8588-3871]

*Национальный исследовательский университет «МЭИ», Россия, г. Москва*

*c216@ya.ru*

***Аннотация***

Численно получены зависимости первых четырёх частот собственных колебаний плоской регулярной фермы распорного типа. Использована модель, в которой масса фермы концентрируется в её узлах. Для расчёта жёсткости фермы использована формула Максвелла–Мора. Для первой частоты методом индукции с использованием упрощённого варианта метода Донкерлея в системе компьютерной математики Maple построена аналитическая зависимость от числа панелей. Показано хорошее совпадение с численным результатом. Получена формула для статического прогиба фермы как функция числа панелей, размеров и нагрузки.

***Ключевые слова****: плоская ферма, собственные колебания, первые частоты колебаний, прогиб, метод Донкерлея, аналитическое решение, Maple, основная частота, формула Максвелла–Мора, регулярная ферма.*

## ВВЕДЕНИЕ

В практических расчётах частоты собственных колебаний конструкций, как правило, используются специализированные численные пакеты на основе метода конечных элементов [1–3]. Альтернативный метод расчёта — аналитический, применим для статически определимых регулярных ферм. Известны два простых метода, дающие оценки первой частоты: метод Донкерлея (оценка снизу) и метод Рэлея (оценка сверху) [4]. Здесь получены формулы для двухсторонней оценки частоты колебаний плоской консольной фермы с раскосной решёткой. В [5] приведён упрощенный вариант метода Донкерлея с более точным аналитическим решением. Аналитическое решение в виде конечной формулы может быть использовано для оценки численного решения, тем более, что точность такого метода не связана с числом стержней в конструкции, в то время как

метод конечных элементов для крупномасштабных систем склонен к накоплению погрешностей. В [6] получена аналитическая оценка основной частоты собственных колебаний регулярной решетчатой фермы и проанализирован спектр всех частот. Оценка основной частоты колебаний пространственной регулярной фермы с горизонтальным ригелем в виде компактной формулы дана в [7]. В [8, 9] исследована зависимость области резонансно безопасных частот спектра собственных колебаний плоской регулярной фермы с произвольным числом панелей от параметров задачи. Аналитическое решение задачи о частоте колебаний пространственной консольной фермы построено в [10]. Формула для собственной частоты колебаний плоской фермы регулярного типа получена в [11]. Справочник [12] содержит схемы плоских регулярных ферм и формулы для расчёта их прогибов, усилий в характерных стержнях и смещений опор.

**СХЕМА ФЕРМЫ**

Рассмотрим схему статически определимой фермы распорного типа с параллельными поясами (рис. 1). Ферма имеет крестообразную решётку и две неподвижные опоры. Средняя часть, приподнятая на высоту *h,* содержит 2*n* панелей длиной 2*a* и высотой 2*h*.

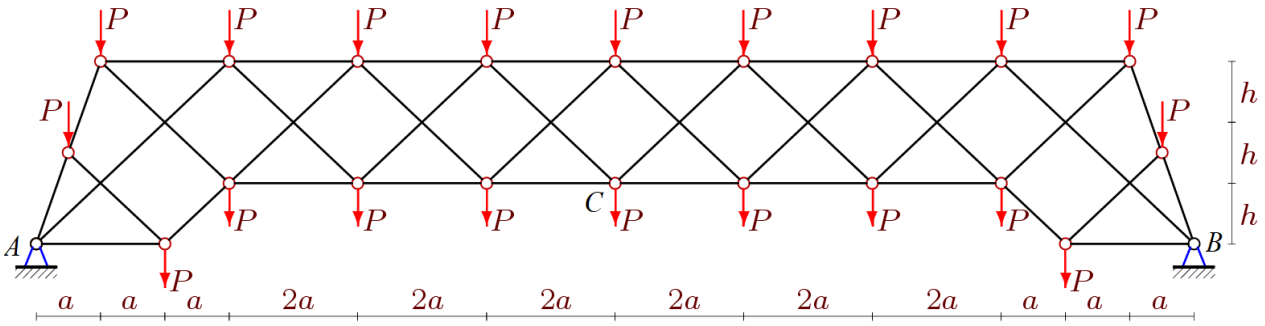



Рис. 1. Ферма под действием нагрузки, *n* = 3

Ферма состоит из $\eta = 8n + 16$ стержней и $4n + 10$ узлов. В число стержней не входят четыре стержня, моделирующие две неподвижные опоры.

Для расчёта жёсткости конструкции по формуле Максвелла–Мора при определении частот колебаний методом вырезания узлов находят усилия в стержнях. Схема конструкции задаётся координатами узлов и порядком соединения стерж-

ней в узлы. Начало координат размещается в левой опоре *A* (рис. 1 и 2). Координаты имеют вид:

$$x_1 = 0, y_1 = 0, x_2 = 2a, y_2 = 0,$$
$$x_{i+2} = (2i+1)a,\ y_{i+2} = h,\ i = 1,..,2n+1,$$
$$x_{i+2n+6} = (2i-1)a,\ y_{i+2n+6} = 3h,\ i = 1,..,2n+3,$$
$$x_{2n+4} = L_0 - 2a,\ y_{2n+4} = 0,$$
$$x_{2n+5} = L_0,\ y_{2n+5} = 0,$$
$$x_{2n+6} = a/2,\ y_{2n+6} = 3h/2,$$
$$x_{2n+7} = a,\ y_{2n+7} = y_{4n+10} = 3h/2,$$
$$x_{4n+10} = L_0 - a/2.$$

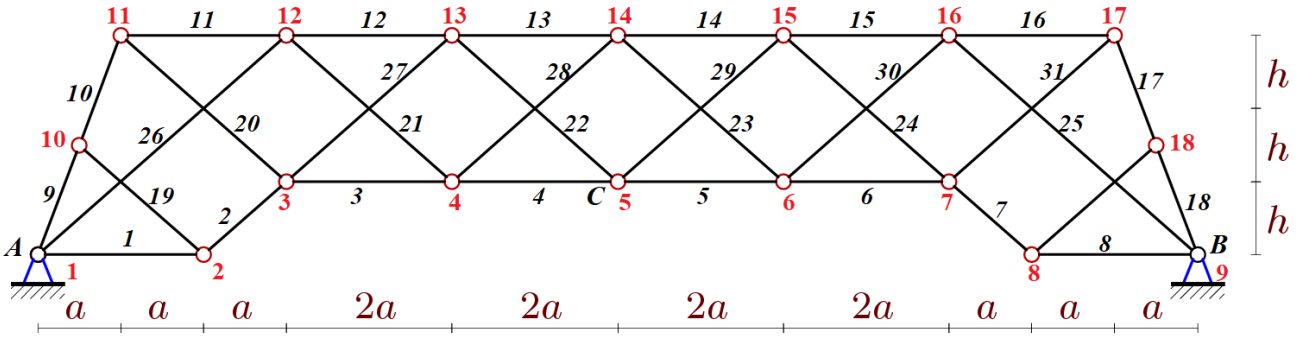



Рис. 2. Нумерация узлов и стержней фермы, *n* = 2

Порядок соединения стержней в узлы решётки фермы определяется специальными списками номеров узлов по концам отдельных стержней конструкции. Стержни поясов, например, задаются неориентированными списками:

$$\Phi_i = [i, i+1],\ \Phi_{i+2n+5} = [i+2n+5, i+2n+6],\ i = 1,..,2n+4.$$

Условие равновесия узлов записывается в виде уравнений в проекции на оси координат. Коэффициентами этих уравнений являются направляющие косинусы усилий:

$$l_{x,i} = (x_{\Phi_{i,1}} - x_{\Phi_{i,2}})/l_i,\ l_{y,i} = (y_{\Phi_{i,1}} - y_{\Phi_{i,2}})/l_i,\ i = 1,...,\eta,$$

где $l_i = \sqrt{l_{x,i}^2 + l_{y,i}^2}$. Матрица $G$ системы линейных уравнений равновесия узлов формируется следующим образом:

$$G_{2\Phi_{i,1}-1,i} = l_{x,i} / l_i,\ G_{2\Phi_{i,1},i} = l_{y,i} / l_i,$$
$$G_{2\Phi_{i,2}-1,i} = -l_{x,i} / l_i, G_{2\Phi_{i,2},i} = -l_{y,i} / l_i.$$

Система уравнений равновесия узлов записывается в матричной форме

$$\mathbf{GS} = \mathbf{T}, \quad (1)$$

где $\mathbf{T}$ — вектор внешних узловых нагрузок, $\mathbf{S}$ — вектор усилий в стержнях. Усилия находятся из решения системы уравнений методом обратной матрицы в системе компьютерной математики Maple.

**ПРОГИБ**

Прогиб фермы с *n* панелями в половине пролета рассчитывается по вертикальному смещению среднего узла нижнего пояса с использованием формулы Максвелла–Мора:

$$\Delta_n = \sum_{i=1}^{\eta} S_i^{(P)} S_i^{(1)} l_i / (EF),$$

где $S_i^{(P)}$ — усилие в стержне с номером *i* от действия нагрузки *P*, распределенной по всем узлам фермы, s — усилие от единичной силы, приложенной к узлу *C* с номером *n*+3, смещение которого рассчитывается, *EF* — жёсткость стержней фермы на продольные усилия. Аналитическая зависимость прогиба от числа панелей определяется методом индукции обобщением последовательности решений для ферм различного порядка. Решение системы (1) в системе Maple даёт следующую последовательность:

$$\Delta_1 = P(51a^3 + 11c^3 + 12h^3) / (2h^2EF),$$
$$\Delta_2 = P(12644a^3 + 528c^3 + 41d^3 + 576h^3) / (72h^2EF),$$
$$\Delta_3 = P(2273a^3 + 117c^3 + 60h^3) / (6h^2EF),$$
$$\Delta_4 = P(27276a^3 + 660c^3 + 19d^3 + 288h^3) / (24h^2EF),$$
$$\Delta_5 = P(11537a^3 + 249c^3 + 84h^3) / (6h^2EF),...$$

Для определения общего члена этой последовательности потребовалось продолжения её до 18 членов. Общий вид полученного решения имеет вид:

$$\Delta_n = P(C_1a^3 + C_2c^3 + C_3d^3 + C_4h^3) / (h^2EF). \quad (2)$$

Операторы системы Maple из решения рекуррентных уравнений дают следующие коэффициенты:

$$C_1 = (60n^4 + 8(4(-1)^n + 25)n^3 + 2(32(-1)^n + 215)n^2 + \\ + 6(27(-1)^n + 79)n + 123(-1)^n + 135) / 36,$$

$$C_2 = (2n^2 + (5 - (-1)^n)n + 2(-1)^n + 5) / 2,$$

$$C_3 = (25 + 8n)((-1)^n + 1) / 144,$$

$$C_4 = 2n + 4.$$

Зависимость (2) имеет асимптотику, кубическую по числу панелей:

$$\lim_{n\to\infty} \Delta_n / n^3 = 5P_0 a^3 / (12h^2 EF),$$

где $P_0 = 4(n+2)P$ – суммарная нагрузка на ферму.

## ЧИСЛЕННОЕ ОПРЕДЕЛЕНИЕ ЧАСТОТ СОБСТВЕННЫХ КОЛЕБАНИЙ

Модель фермы предполагает, что масса фермы равномерно распределена по узлам сосредоточенными массами $m$. Колебания происходят по оси *y*. Горизонтальные смещения масс не учитываются, число степеней свободы системы масс равно числу узлов $K = 4n + 10$. Уравнения движения масс в узлах фермы записываются в матричной форме:

$$\mu I_K \ddot{Y} + D_K Y = 0. \qquad (3)$$

Здесь $Y$ – вектор вертикальных смещений узлов фермы, $\ddot{Y}$ – вектор ускорений, $I_K$ — единичная матрица, $D_K$ – матрица жёсткости. В предположении, что колебания гармонические с частотой $\omega$, справедлива замена $\ddot{Y} = -\omega^2 Y$. При умножении уравнения (3) слева на матрицу податливости $B_K$ задача сводится к проблеме собственных значений матрицы $B_K$: $B_K Y = \lambda Y$, где $\lambda = 1/(\omega^2 \mu)$ — собственные числа. Матрица податливости является обратной матрице жёсткости: $B_K = 1/D_K$. Значения элементов этой матрицы вычисляются по формуле Максвелла–Мора:

$$b_{i,j} = \sum_{\alpha=1}^{\eta} S_\alpha^{(i)} S_\alpha^{(j)} l_\alpha / (EF), \qquad (4)$$

где $S_\alpha^{(i)}$ — усилие в стержне с номером $\alpha = 1,..,\eta$ от действия единичной вертикальной силы, приложенной к узлу *i*. В число стержней включены четыре стержня, моделирующие неподвижные шарнирные опоры. Длина вертикальных опорных

стержней принята равной *h*, горизонтальных — *a*. Эти длины определяют жёсткость опор. Жёсткость *EF* всех стержней фермы одинаковая. Вычислить собственные числа матрицы для расчёта спектра частот можно численно в системе Maple.

На рис. 3 представлены результаты расчёта первых трёх собственных частот в зависимости от числа панелей в ферме. Приняты размеры фермы: *h* = 2м, *a* = 3 м, массы в узлах $\mu = 200$ кг, модуль упругости $E = 2.1 \cdot 10^5$ МПа, площадь поперечного сечения стержней $F = 9\ \text{см}^2$. Плотность распределения различных частот существенно зависит от числа панелей. При *n* = 6, например, две верхние частоты $\omega_3$ и $\omega_4$ совпадают, а для фермы с одной панелью в половине пролета (*n* = 1) совпадают частоты $\omega_1$ и $\omega_2$. Для первой частоты можно найти приближенное аналитическое выражение.

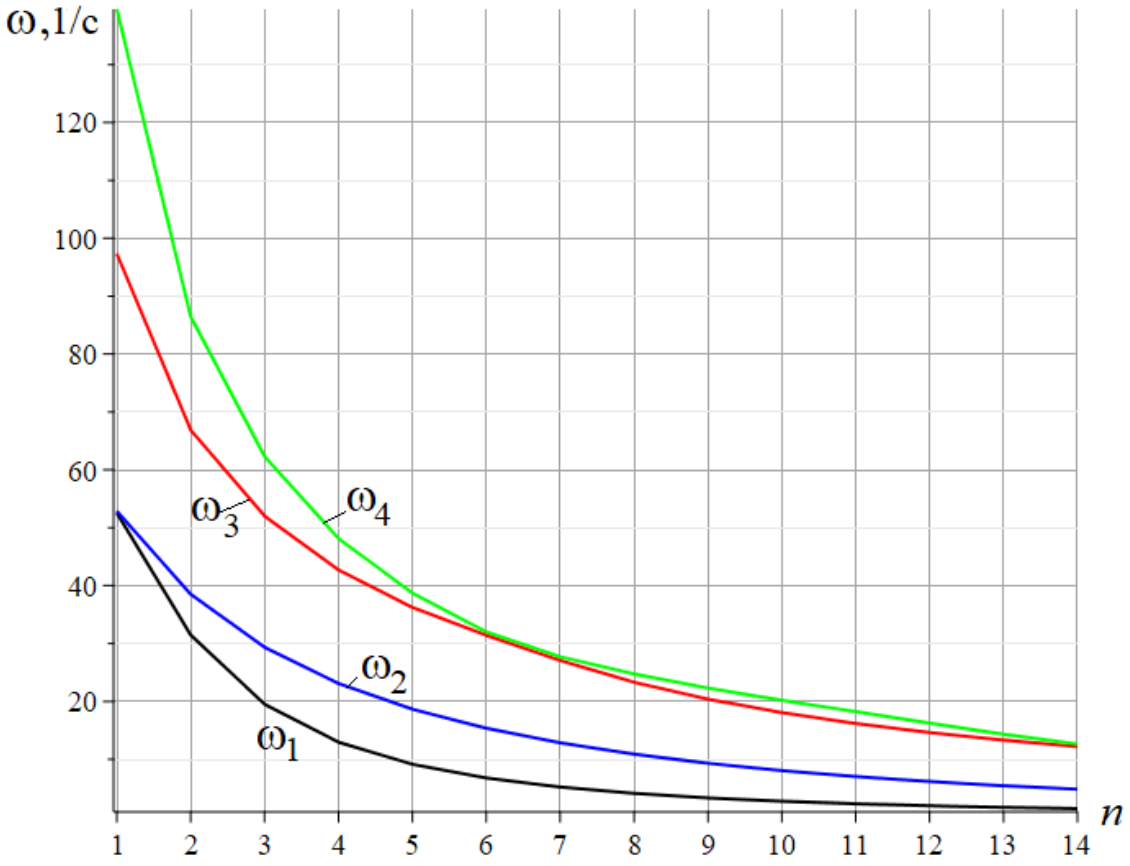



Рис. 3. *Частоты колебаний в зависимости от числа панелей*

**ФОРМУЛА ДЛЯ ОСНОВНОЙ ЧАСТОТЫ**

Для приближенной оценки нижней границы основной частоты известна формула Донкерлея

$$\omega_D^{-2} = \sum_{j=1}^{K} \omega_j^{-2}, \tag{5}$$

где $\omega_j$ — парциальная частота груза в узле *j*, вычисленная из уравнения его движения:

$$\mu \ddot{y}_j + D_j y_j = 0, \quad j = 1, 2, ..., K.$$

Коэффициент жёсткости $D_j$ есть величина, обратная податливости, которая вычисляется по формуле Максвелла–Мора:

$$\delta_j = 1/D_j = \sum_{\alpha=1}^{\eta} \left( S_\alpha^{(j)} \right)^2 l_\alpha / (EF). \tag{6}$$

Из (5) и (6) следует формула для нижней границы первой собственной частоты по Донкерлею:

$$\omega_D^{-2} = \mu \sum_{j=1}^{K} \delta_j = \mu \Delta_n. \tag{7}$$

Метод Донкерлея имеет два недостатка: заниженное значение частоты и сложность вычисления суммы в символьной форме. Этих недостатков лишен упрощенный вариант метода Донкерлея [5], в котором предложено заменить сумму приближенным её значением, рассчитанным по теореме о среднем. Сумма ординат в (7) ассоциируется с площадью криволинейной фигуры, для вычисления которой используется формула площади треугольника:

$$\omega_D^{-2} = \mu \sum_{j=1}^{K} \delta_j = \mu \delta_{\max} K / 2 = \mu \Delta_{\max},$$

где $\delta_{\max}$ — максимальное значение $\delta_j$. Точка, смещение которой под действием силы, приложенной к ней, имеет наибольшее значение, выбирается опытным путем. В данной задаче, очевидно, это средний шарнир *C* в нижнем поясе с номером *n*+3. Рассчитав значение максимального прогиба от единичной силы для ферм разного порядка, получим последовательность

$$\Delta_{\max 1} = K(9a^3 + 5c^3 + h^3) / (4h^2 EF),$$
$$\Delta_{\max 2} = K(409a^3 + 54c^3 + d^3 + 9h^3) / (36h^2 EF),$$
$$\Delta_{\max 3} = K(65a^3 + 9c^3 + h^3) / (4h^2 EF),$$
$$\Delta_{\max 4} = K(1697a^3 + 90c^3 + d^3 + 9h^3) / (36h^2 EF),$$
$$\Delta_{\max 5} = K(233a^3 + 13c^3 + h^3) / (4h^2 EF), \ldots$$

Обобщение этого ряда на произвольное число панелей даёт формулу

$$\Delta_{\max} = (4n + 10)(C_1 a^3 + C_2 c^3 + C_3 d^3 + h^3 / 4) / (h^2 EF),$$

где

$$C_1 = (12n^3 + 6(2(-1)^n + 5)n^2 + \\ + 8(2(-1)^n + 5)n + 3(-1)^n + 30) / 36,$$
$$C_2 = (4n - (-1)^n + 5) / 8,$$
$$C_3 = ((-1)^n + 1) / 72.$$

Отсюда следует формула для расчёта первой частоты свободных колебаний фермы:

$$\omega_* = h\sqrt{\frac{EF}{\mu(4n + 10)(C_1 a^3 + C_2 c^3 + C_3 d^3 + h^3 / 4)}}. \quad (8)$$

**СРАВНЕНИЕ РЕШЕНИЙ. ЧИСЛЕННЫЙ РАСЧЁТ**

Для оценки приближённого аналитического решения (8) надо найти первую частоту численно. Приняты те же параметры фермы, что и в решении задачи о первых четырёх частотах на рис. 3. На рис. 4 приведено сравнение аналитической зависимости (8) частоты $\omega_*$ от числа панелей и частоты $\omega_1$, полученной численным путём. Рассмотрены два варианта высоты *h*.

При увеличении числа панелей собственная частота монотонно уменьшается, а результаты аналитического расчёта приближаются к численному. Для уточнения погрешности аналитического метода введём относительную величину $\varepsilon_* = |\omega_1 - \omega_*| / \omega_1$. Из рис. 5 видно, что точность формулы (8) с увеличением числа панелей растет. Для ферм с меньшей высотой погрешность незначительно меньше. Существенно влияет на точность и чётность числа панелей. Например, при *n*=5 точность в три раза больше, чем при *n*=6.

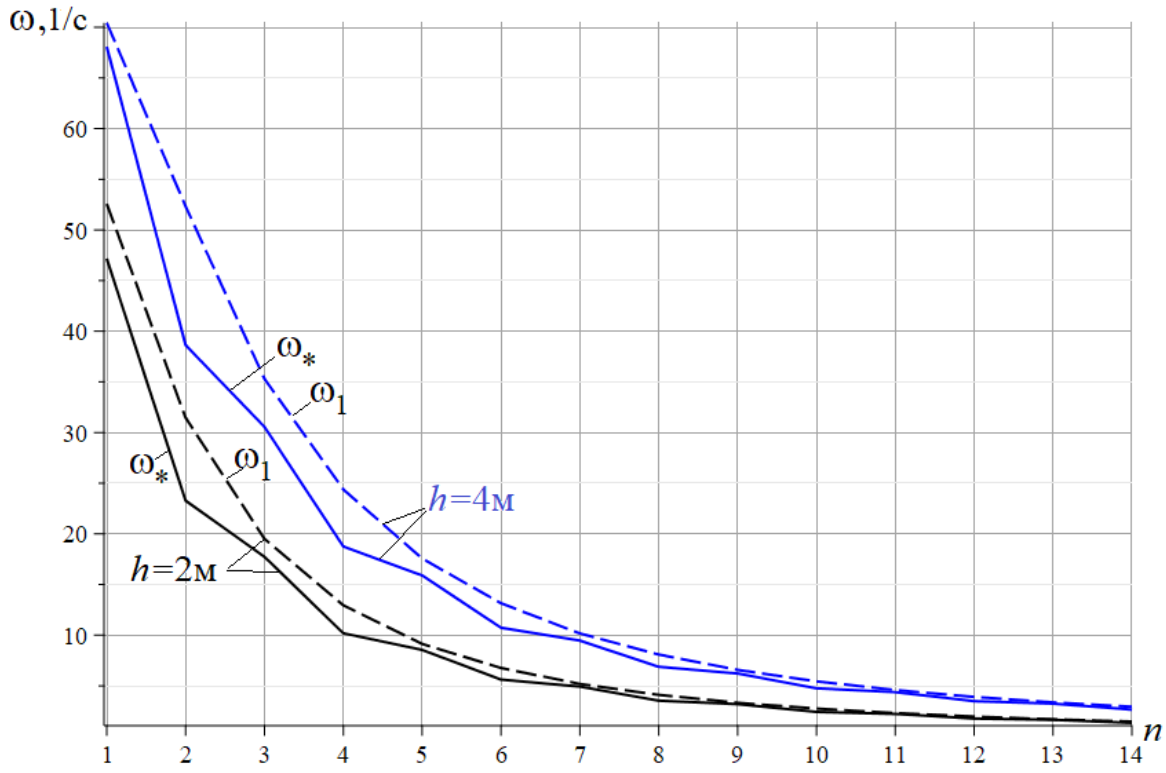



Рис. 4. Зависимость первой частоты собственных колебаний фермы от числа панелей при $h$ = 2 м и $h$ = 4 м.

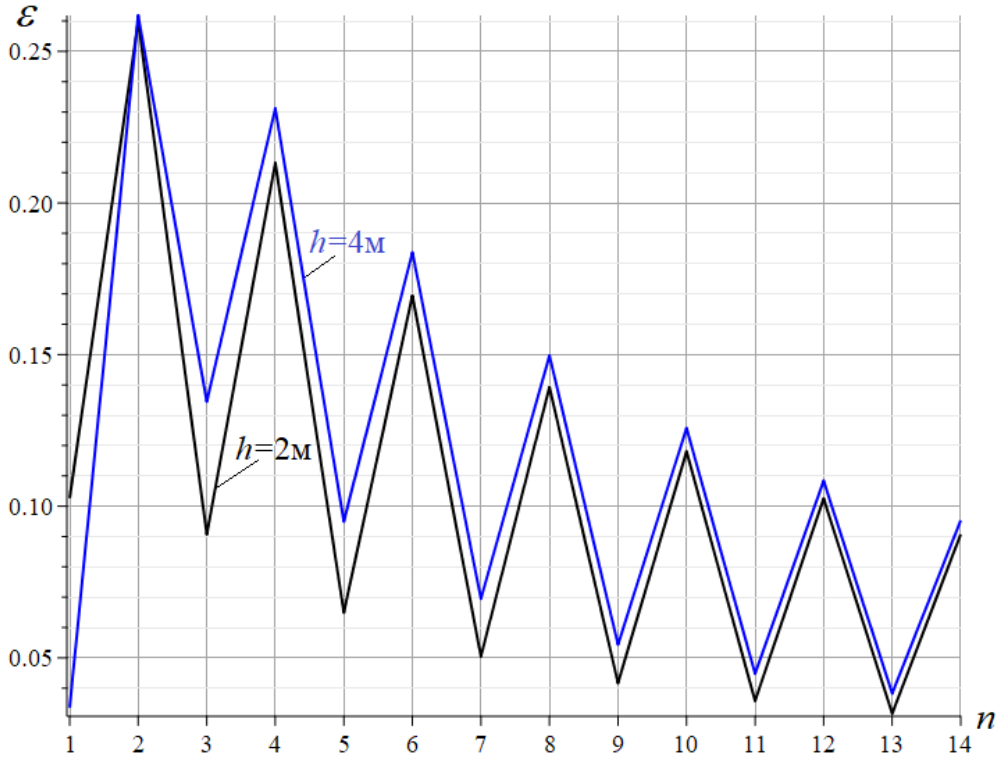



Рис. 5. Зависимости погрешности аналитического решения от числа панелей

## ЗАКЛЮЧЕНИЕ

Рассмотрена схема плоской распорной фермы арочного типа. Построена формула зависимости прогиба фермы под действием распределенной узловой нагрузки и дан ее анализ. Численно рассчитаны первые четыре собственные частоты колебаний в зависимости от числа панелей. Для первой частоты методом индукции получено приближённое аналитическое выражение. Показано, что точность этого решения растет с увеличением числа панелей. Предложенный алгоритм построения аналитического решения может быть использован для расчётов основной частоты регулярных статически определимых ферм. Одно из преимуществ аналитического решения, помимо его очевидной простоты, состоит в том, что если погрешность численного решения закономерно растет с увеличением числа панелей, то для аналитического решения она падает. Полученная формула может служить простой оценкой численного решения, полученного для более точной модели этой же фермы, например, с учётом масс стержней.

## СПИСОК ЛИТЕРАТУРЫ

1. *Саиян С.Г., Шитикова М.В.* Сравнительный анализ динамического отклика зданий и сооружений различной высотности на ветровые и сейсмические воздействия // Строительная механика и конструкции. 2025. № 1(44). С. 16–30. https://doi.org/10.36622/2219-1038.2025.44.1.002. EDN QCCKXD.

2. *Han Q.H., Xu Y., Lu Y., Xu J., Zhao Q.H.* Failure mechanism of steel arch trusses: Shaking table testing and FEM analysis // Engineering Structures. 2015. Vol. 82. P. 186–198. https://doi.org/10.1016/j.engstruct. 2014.10.013.

3. *Plevris V., Ahmad A.* Deriving Analytical Solutions Using Symbolic Matrix Structural Analysis: Part 2 – Plane Trusses // Heliyon. 2025. Vol. 11. No. 4. https://doi.org/10.1016/j.heliyon.2025.e42372

4. *Vorobev O*. Bilateral analytical estimation of first frequency of a plane truss // Construction of Unique Buildings and Structures. 2020. Vol. 92. Article No. 9204. https://doi.org/10.18720/CUBS.92.4

5. *Kirsanov M.* Simplified Dunkerley Method for Estimating the First Oscillation Frequency of a Regular Truss // Construction of Unique Buildings and Structures. 2023. Vol. 108. https://doi.org/10.4123/CUBS.108.1.

6. *Комерзан Е.В., Маслов А.Н.* Аналитическая оценка основной частоты собственных колебаний регулярной фермы // Строительная механика и конструкции. 2023. №2(37). С. 17–26. https://doi.org/10.36622/ VSTU.2023.37.2.002

7. *Комерзан Е.В., Маслов А.Н.* Оценка основной частоты колебаний Г-образной пространственной фермы // Строительная механика и конструкции. 2023. №2(37). С. 35–45. https://doi.org/10.36622/ VSTU.2023.37.2.004

8. *Лыонг Конг Л.* Зависимость области резонансно безопасных частот от размеров статически определимой плоской фермы // Строительная механика и конструкции. 2024. №2(41). С. 16–26.
https://doi.org/10.36622/2219-1038.2024.41.2.002

9. *Luong C.L.* Resonance safety zones of a truss structure with an arbitrary number of panels // Construction of Unique Buildings and Structures. 2024. Vol. 113. Article No. 11304. https://doi.org/10.4123/CUBS.113.4

10. *Sviridenko O., Komerzan E.* The dependence of the natural oscillation frequency of the console truss on the number of panels // Construction of Unique Buildings and Structures. 2022. Vol. 101 Article No. 10101.
https://doi.org/10.4123/CUBS.101.1

11. *Maslov A.* The first natural frequency of a planar regular truss. Analytical solution // Construction of Unique Buildings and Structures. 2023. Vol. 109. Article No. 10912. https://doi.org/10.4123/CUBS.109.12

12. *Kirsanov M.N.* Planar Trusses: Schemes and Formulas. Cambridge Scholars Publishing. 2019. 198 p. Lady Stephenson Library, Newcastle upon Tyne, NE6 2PA, UK ISBN (13): 978-1-5275-3531-2

# DEFLECTION AND FIRST NATURAL FREQUENCIES OF A REGULAR ARCHED TRUSS OSCILLATION

**M. N. Kirsanov**[0000-0002-8588-3871]

*National Research University MPEI, Moscow, Russia*

*c216@ya.ru*

***Abstract***

The dependences of the first four frequencies of natural oscillations of a planar regular truss of the thrust type are obtained numerically. A model is used in which the mass of the truss is concentrated in its nodes. The Maxwell–Mohr formula is used to calculate the rigidity of the truss. For the first frequency, an analytical dependence on the number of panels is derived by the induction method using a simplified version of the Dunkerley method in the Maple computer mathematics system. Good agreement with the numerical result is shown. An analytical dependence of the static deflection of the truss on its dimensions and load is obtained.

***Keywords****: Planar truss, natural oscillations, first oscillation frequencies, deflection, Dunkerley method, analytical solution, Maple, fundamental frequency, Maxwell–Mohr formula, regular truss.*

**REFERENCES**

1. *Saiyan S.G., Shitikova M.V.* Comparative analysis of the dynamic response of buildings and structures of different heights to wind and seismic loading // Structural mechanics and structures. 2025. Vol. 1 (44). P. 16–30. https://doi.org/10.36622/2219-1038.2025.44.1.002. EDN QCCKXD.

2. *Han Q.H., Xu Y., Lu Y., Xu J., Zhao Q.H.* Failure mechanism of steel arch trusses: Shaking table testing and FEM analysis // Engineering Structures. 2015. Vol. 82. P. 186–198. https://doi.org/10.1016/j.engstruct.2014.10.013.

3. *Plevris V., Ahmad A.* Deriving Analytical Solutions Using Symbolic Matrix Structural Analysis: Part 2 – Plane Trusses // Heliyon. 2025. Vol. 11. No. 4. https://doi.org/10.1016/j.heliyon.2025.e42372

4. *Vorobev O.* Bilateral analytical estimation of first frequency of a plane

truss // Construction of Unique Buildings and Structures. 2020. Vol. 92. Article No. 9204. https://doi.org/10.18720/CUBS.92.4

5. *Kirsanov M.* Simplified Dunkerley Method for Estimating the First Oscillation Frequency of a Regular Truss // Construction of Unique Buildings and Structures. 2023. Vol. 108. https://doi.org/10.4123/CUBS.108.1

6. *Komerzan E.V., Maslov A.N.* Analytical assessment of the fundamental frequency of natural vibrations of a regular truss // Structural mechanics and structures. 2023. Vol. 2 (37). P. 17–26. https://doi.org/10.36622/ VSTU.2023.37.2.002

7. *Komerzan E.V., Maslov A.N.* Estimation of the fundamental vibration frequency of an L-shaped spatial truss // Structural mechanics and structures. 2023. Vol. 2 (37). P. 35–45. https://doi.org/10.36622/VSTU.2023.37.2.004

8. *Luong Kong L.* Dependence of the region of resonantly safe frequencies on the dimensions of a statically determinate flat truss // Structural Mechanics and Structures. 2024. No. 2 (41). P. 16–26.
https://doi.org/10.36622/2219-1038.2024.41.2.002

9. *Luong C.L.* Resonance safety zones of a truss structure with an arbitrary number of panels // Construction of Unique Buildings and Structures. 2024. Vol. 113. Article No. 11304. https://doi.org/10.4123/CUBS.113.4

10. *Sviridenko O., Komerzan E.* The dependence of the natural oscillation frequency of the console truss on the number of panels // Construction of Unique Buildings and Structures. 2022. Vol. 101 Article No. 10101.
https://doi.org/ 10.4123/CUBS.101.1

11. *Maslov A.* The first natural frequency of a planar regular truss. Analytical solution // Construction of Unique Buildings and Structures. 2023. Vol. 109. Article No. 10912. https://doi.org/10.4123/CUBS.109.12

12. *Kirsanov M.N.* Planar Trusses: Schemes and Formulas. Cambridge Scholars Publishing. 2019. 198 p. Lady Stephenson Library, Newcastle upon Tyne, NE6 2PA, UK ISBN (13): 978-1-5275-3531-2

## СВЕДЕНИЯ ОБ АВТОРЕ


***КИРСАНОВ Михаил Николаевич*.** Д-р физ.-мат. наук, профессор кафедры робототехники, мехатроники, динамики и прочности машин, Национальный исследовательский университет «МЭИ». Россия, г. Москва

***Mikhail Nikolaevich KIRSANOV*.** Doctor of Physical and Mathematical Sciences, Professor of the Department of Robotics, Mechatronics, Dynamics, and Strength of Machines, National Research University "MPEI. Moscow, Russia.

e-mail: c216@ya.ru

ORCID: 0000-0002-8588-3871

*Материал поступил в редакцию 5 апреля 2025 года*

УДК 551.5+ 556+ 504.3+ 504.4+ 51-73+ 51-74+ 51-76+ 517.5

# НОВЫЕ ВОЗМОЖНОСТИ ПРЕОБРАЗОВАНИЯ ФУРЬЕ: КАК ОПИСАТЬ ПРОИЗВОЛЬНЫЙ ЧАСТОТНО-ФАЗОВЫЙ МОДУЛИРОВАННЫЙ СИГНАЛ?

**Р. Р. Нигматуллин[1] [0000-0003-2931-4428], А. А. Литвинов[2] [0009-0000-3901-3704],**

**С. И. Осокин[3] [0000-0002-0699-5390]**

*[1, 2]Казанский национальный исследовательский технический университет имени А.Н. Туполева, 420111 Казань, Россия*
*[3]Казанский федеральный университет, Институт информационных технологий и интеллектуальных систем, 420008 Казань, Россия*

[1]renigmat@gmail.com, [2]litvinov85@gmail.com, [3]s.osokin@it.kfu.ru

**Аннотация**

В работе построено преобразование любого произвольного сигнала в строго периодическую форму, которое позволяет применять обычное преобразование Фурье для аппроксимации уже преобразованного сигнала. Наиболее интересным приложением (по мнению авторов) является аппроксимация сигналов с частотно-фазовой модуляцией, которые фактически находятся внутри найденного преобразования. Это новое преобразование будет полезным для описания откликов различных сложных систем, когда отсутствует обычная модель описания. В качестве доступных данных мы рассматриваем метеоданные, соответствующие измерениям концентрации метана ($CH_4$) в атмосфере в течение 4 недель наблюдений. Было важно рассмотреть интегральные (кумулятивные) данные и найти их амплитудно-частотную характеристику (АЧХ). Если рассматривать каждый столбец как сигнал с частотно-фазовой модуляцией, то АЧХ можно оценить с помощью преобразования Фурье, период которого равен $2\pi$, что справедливо для любого анализируемого случайного сигнала. Такое «универсальное» преобразование Фурье позволяет описать широкий набор случайных сигналов и сравнить их между собой по АЧХ. Эти новые возможности традиционного Фурье-анализа позволяют преобразованию Фурье стать еще более востребованным инструментом в арсенале методов, используемых исследователями в области обработки данных.

***Ключевые слова**: преобразование Фурье, случайный сигнал, частотно-фазовый модулированный сигнал, амплитудно-частотная характеристика, сложные системы, метеорологические данные, вихревые ковариации.*

***Список основных аббревиатур:** АЧХ – амплитудно-частотная характеристика; НОКАФСС – Не Ортогональный Комбинированный Анализ Фурье для Сглаженных Сигналов); NOCFASS – Non-Orthogonal Combined Fourier Analysis of the Smoothed Signals; ПРА – последовательность ранжированных амплитуд*

## 1. ВВЕДЕНИЕ И ПОСТАНОВКА ЗАДАЧИ

Всем известен основной недостаток традиционного преобразования Фурье. Он заключается в предположении, что любой цифровой случайный сигнал является периодическим, т. е.

$$Sg(t+T)=Sg(t). \quad (1)$$

Здесь Sg(t) – произвольный случайный сигнал, T – период, совпадающий с Range(t). Range(t)=max t−min t определяет длину анализируемого сигнала Sg(t). Во многих случаях это предположение о периодичности сигнала оказывается недоказуемым, но при этом используется.

Если мы хотим заменить чисто периодический сигнал его апериодической копией, то возникает проблема невозможности аппроксимации дискретного апериодического сигнала. Эта проблема не может быть решена интегральным преобразованием Фурье апериодических сигналов, следовательно, оно не может быть использовано для предсказания апериодического сигнала за пределами заданного интервала временного окна.

Дискретные представления многих аналоговых сигналов играют важную роль при их обработке. Они содержат необходимую информацию, связанную со свойствами сигналов, и допускают их дальнейшую обработку [1]. В традиционной схеме сигналы могут быть представлены в виде рядов Тейлора–Маклорена, Дирихле, Лорана, Лежандра, Паде, Прони и Фурье. Подчеркнём, что, по нашему мнению, эти ряды разложения применяются для описания данных без какого-либо математического обоснования. В области обработки сигналов классический ряд Фурье – простой и часто используемый инструмент. Однако он не позволяет

выделить как субгармонические, так и интергармонические компоненты заданного сигнала, и во многих ситуациях имеет серьёзные недостатки [2–5]. Предлагаемый метод позволяет преодолеть ограничения Фурье-анализа. Фурье-анализ основан на частотно-временных методах [6], которые использовались в последние десятилетия, а именно: дискретное [7–9], быстрое [10–12], оконное преобразование Фурье [13, 14], преобразование Габора [15–17], вейвлет [18–20], преобразование Гильберта–Хуанга [21–23], преобразование Фурье–Бесселя [24, 25] и даже разложение по эмпирическим модам [26, 27]. В указанных работах показано, что существует множество различных приближений для преодоления основного недостатка (1), связанного с предположением о периодичности. Однако внимательный анализ этих приближений позволяет сформулировать следующую задачу: определить, существует ли универсальное преобразование исходного сигнала, позволяющее преобразовать его в другой цифровой сигнал, имеющий строго период $2\pi$. Действительно, если записать следующее соотношение:

$$Sg(t)=a\cdot\cos(F(t))+b \tag{2}$$

и начать анализировать вместо исходной функции Sg(t) аргумент F(t) косинуса в (2), то мы получим желаемый результат. Функция F(t) представляет собой объединённый сигнал с частотно-фазовой модуляцией и обеспечивает желаемый интервал [–1, 1] для cos(F(t)), а аргумент F(t) попадает в интервал $[0,\pi]$. Поэтому конечный результат разложения любого сигнала $\cos(\Omega_k t)$ сохраняет прежний вид (2), если к нему добавить разложение F(t) по аппроксимирующей функции Yft(t,K) в виде конечного отрезка ряда Фурье

$$\begin{aligned} &F(t)\cong Yft(t,K)=Ph_0+\sum_{k=1}^{K}\left[Ac_k\cos\left(\Omega_k t\right)+As_k\sin\left(\Omega_k t\right)\right],\\ &\Omega_k=2,3,\ldots,K. \end{aligned} \tag{3}$$

Здесь мы учитываем свойство $F(t)=F(t\pm\pi)$, определяющее полупериодическую функцию. Таким образом, эти два простых выражения (2) и (3) достаточно эффективно решают задачу разложения любой случайной функции Sg(t) в ряд Фурье, поскольку в данном случае решена проблема периодичности разлагаемого сигнала.

## 2. ОПИСАНИЕ МЕТЕОДАННЫХ

Скажем несколько слов о реальных данных и их особенностях. В качестве реальных данных мы взяли вихревые ковариационные экологические данные, связанные с содержанием метана $CH_4$ в атмосфере. В настоящей работе мы рассматриваем баланс метана, т. е. произведение соответствующей концентрации на величину вертикальной скорости. Были взяты данные с приборов на вышке, расположенной возле Обсерватории Казанского университета, с измерениями вихревых ковариаций, связанных с содержанием метана в атмосфере, и скорости воздушных потоков. Измерения концентрации метана $CH_4$ (umol/mol) и вертикальной скорости воздуха W (m/s) проводились с 1 по 7 января 2024 года. Значения данных $CH_4$ были умножены на соответствующие измеренные значения скорости потока воздуха W. Частота измерения составляла 10 раз в секунду. Данные за одну секунду усредняли, а уже секундные (усредненные) значения собирали в часовые группы/столбцы. Получилось 168 часовых столбцов в неделю по 3600 секунд в столбце.

## 3. ОПИСАНИЕ ПРОЦЕДУРЫ ОБРАБОТКИ

Каждая прямоугольная матрица $N \times M$ содержит $N = 3600$ строк (каждая строка соответствует одной секунде измерений и, следовательно, один столбец – одному часу измерений), а количество столбцов $M = 24 \times 7 = 168$ соответствует одной неделе измерений. Демонстрируя новую модификацию преобразования Фурье, мы рассматриваем только три базовые кривые, соответствующие максимальным, средним и минимальным значениям. Эти кривые показаны на рис. 1 Здесь независимая переменная $x_j = j/N$ ($N = 3600$). Сделали нормировку независимой переменной на общее количество измерений за один час.

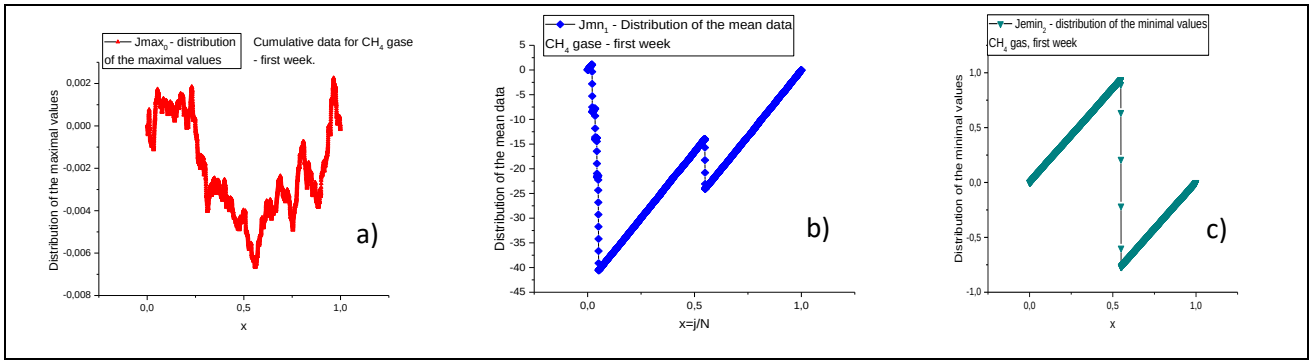



Рис. 1. Слева показана кривая с максимальными значениями, в центре – средние данные (усредненные по всему периоду измерений), а справа – кривая с минимальными значениями

Вычисление аргументов Fq(t) (q = 0, 1, 2) из (2) для этих кривых, расположенных в интервале (0, $\pi$), показано на рис. 2.

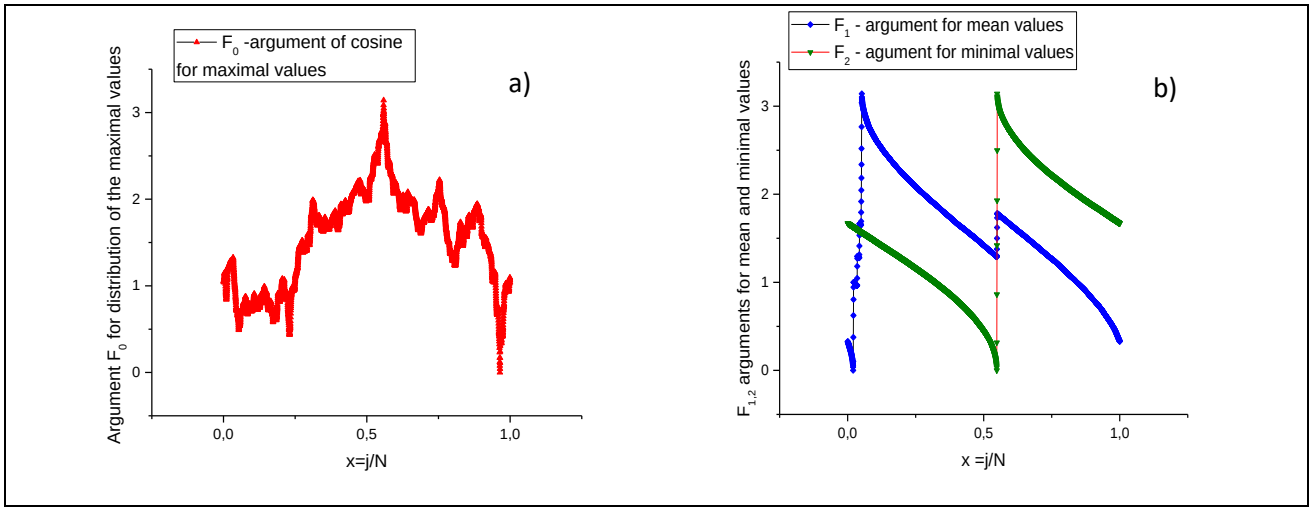



Рис. 2. Слева показан аргумент $F_0(t)$ функции косинуса в (2) для максимальной кривой, справа – соответствующие аргументы для средней кривой $F_1(t)$ (синие точки) и минимальной кривой $F_2(t)$ (зеленые точки)

Для нахождения АЧХ для этих трёх кривых удобно использовать метод НОКАФСС (Не Ортогональный Комбинированный Анализ Фурье для Сглаженных Сигналов) / NOCFASS (Non-Orthogonal Combined Fourier Analysis of the Smoothed Signals) [28], предложенный одним из авторов (RRN) настоящей работы. Основная идея подхода НОКАФСС заключается в смещении экстремальной кривой к центру преобразования Фурье с помощью угла $\pi$. Она поясняется рисунками 3.

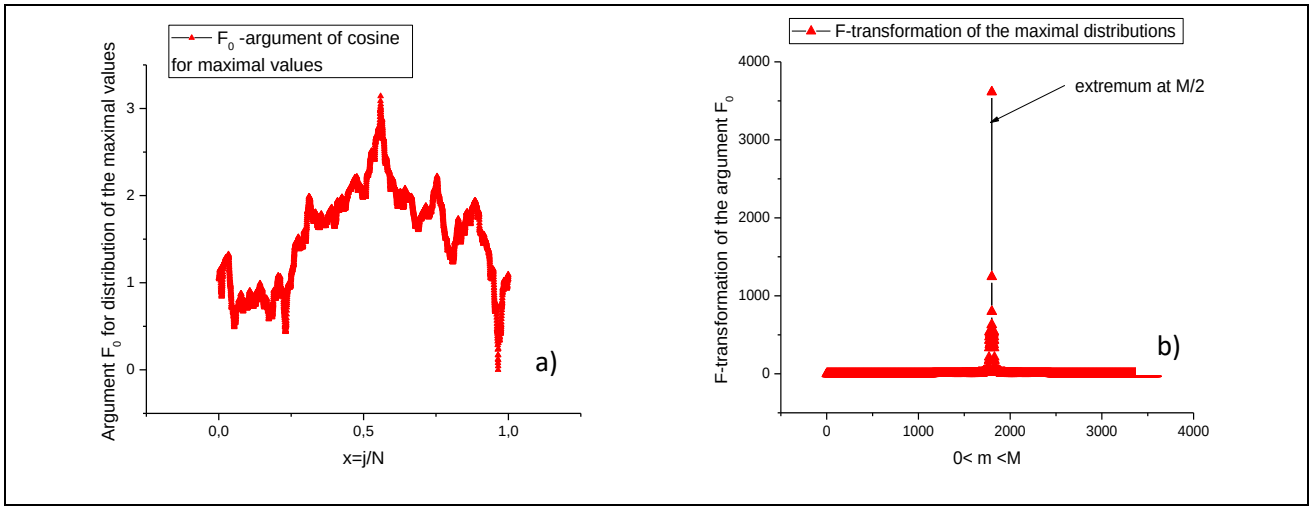



Рис. 3. Слева показан аргумент $F_0(t)$, соответствующий кривой с максимальными значениями. Справа – преобразование Фурье, смещенное к центру N/2 = 1800 с резонансным значением, равным 4032.668

Анализ этого преобразования Фурье показал, что достаточно взять небольшое количество частот, расположенных в окрестности резонансного пика, показанного справа. Поэтому мы рассматриваем безразмерную полосу частот в интервале [1800, 2000].

Такой модифицированный подход позволил уменьшить количество мод, которое существенно зависит от значения конечной моды K. Это значение определяется величиной относительной погрешности, которая, в свою очередь, вычисляется как

$$\mathrm{RelErr}(K) = \left[ \frac{\mathrm{stdev}(F(t) - Yft(t,K))}{\mathrm{mean}\left|F(t)\right|} \right] \cdot 100\% . \qquad (4)$$

Для аппроксимации достаточно взять небольшое количество частот, охватывающих интервал [Vect, Vect+200]. Итоговый результат показан на рис. 4 и 5.

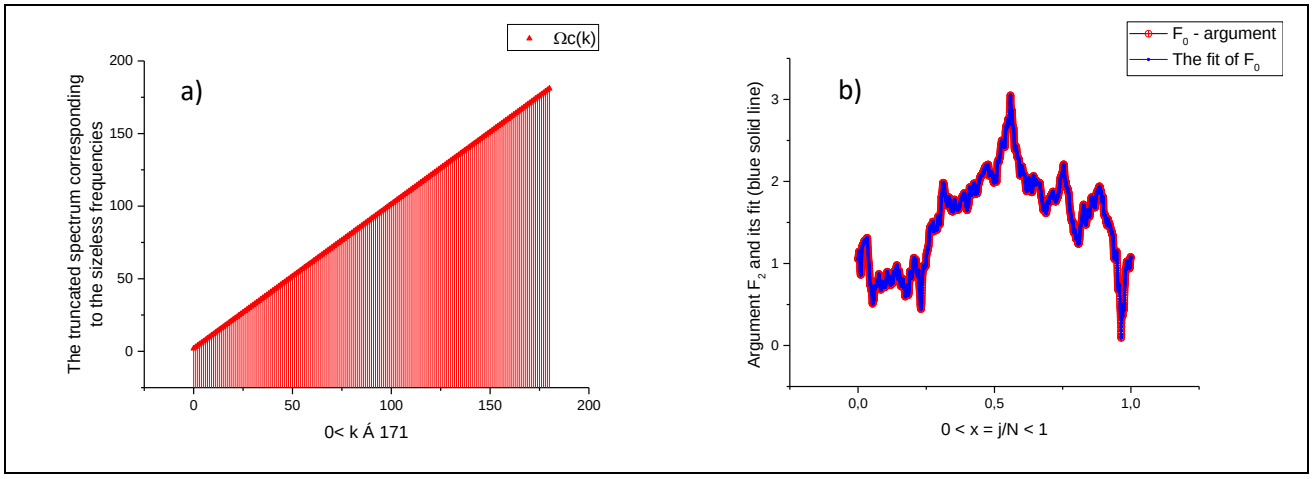



Рис. 4. Слева показан усеченный спектр, расположенный в интервале [2, 171]. Справа аппроксимирующая функция Yft(t) для функции $F_0(t)$

Усеченный спектр, показанный на рис. 4 (a), достаточен для обеспечения аппроксимации с величиной относительной ошибки (определяемой выражением (4)) менее 1%. Точные значения относительной погрешности собраны в таблице 1.

АЧХ для усеченного спектра приведена на рис. 5.

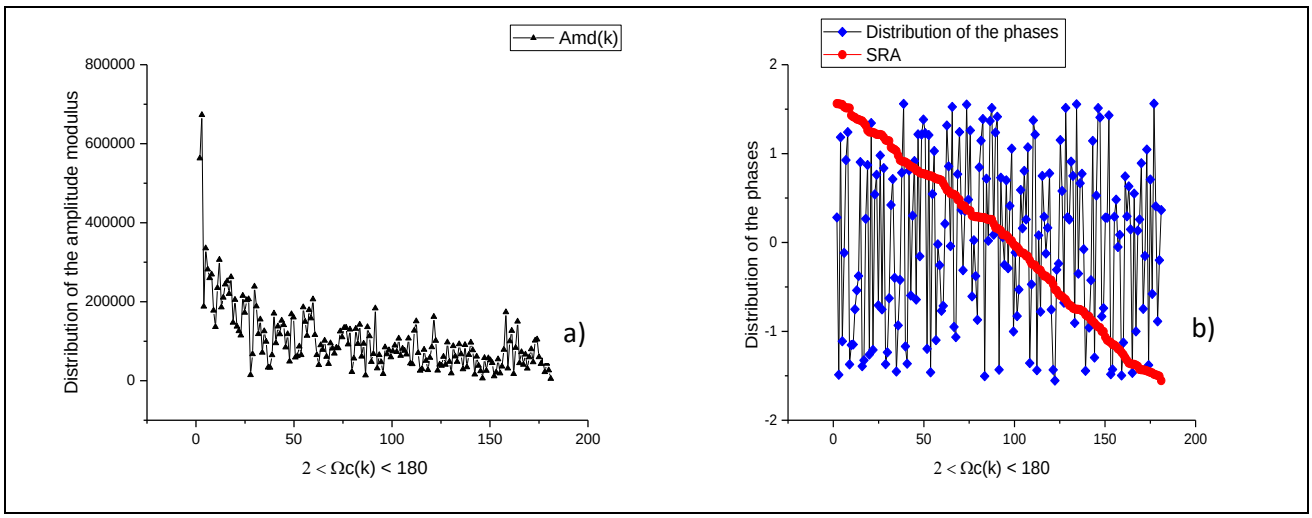



Рис. 5. Слева показан модуль амплитуд $\mathrm{Amd}_k = \sqrt{\mathrm{Ac}_k^2 + \mathrm{As}_k^2}$ ,справа – распределение фаз $\mathrm{Phase}_k = \tan^{-1}\left(\mathrm{As}_k / \mathrm{Ac}_k\right)$

Сплошная прямая линия, показанная на правом рисунке, соответствует последовательности ранжированных амплитуд (ПРА) упорядоченных фаз. Легко заметить, что ПРА близка по виду к отрезку прямой линии. Это свидетельствует

о том, что распределение фаз практически однородно. Чтобы завершить процедуру аппроксимации, продемонстрируем переход от аппроксимации аргумента $F_0(t)$ к исходному сигналу $Mex(t)_0$ (см. рис. 6).

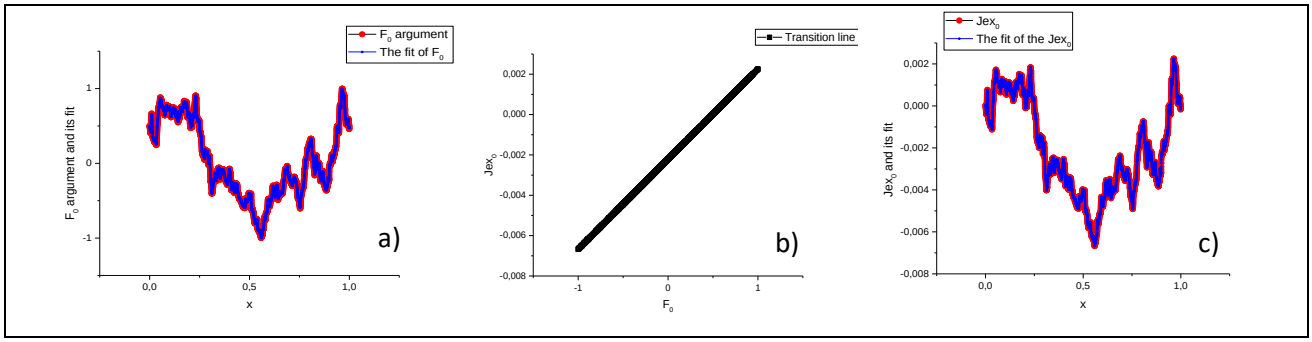



Рис. 6. Слева показаны масштабированная функция $mex_0(t) = \cos(F_0(t))$ и ее аппроксимация $Jex_0(Yft(t))$. Центральный рисунок помогает определить значения наклона a и пересечения b. Справа синей сплошной линией показана окончательная аппроксимирующая функция

На рис. 6 показано восстановление исходного сигнала. Слева можно видеть масштабированную функцию $mex_0(t) = \cos(F_0(t))$ (красные точки) и ее аппроксимацию $Jex_0(Yft(t))$, показанную синей сплошной линией. Чтобы найти параметры масштабирования, можно использовать центральный рисунок, который помогает определить значения наклона ($a = 0.00446$) и пересечения ($b = -0.00221$). Окончательная аппроксимирующая функция показана на правом рисунке синей сплошной линией.

Приближенная аппроксимирующая функция исходного сигнала определяется выражением

$$Ex(t) = a\cos\left(Yft(t)\right) + b,$$

где параметры масштабирования a и b находятся как значения наклона и пересечения соответственно из центрального рис. 6 (b). Таким же образом можно аппроксимировать случайные кривые, изображенные на рисунке 2 (b). Приведём основные из них (см. рис. 7–10).

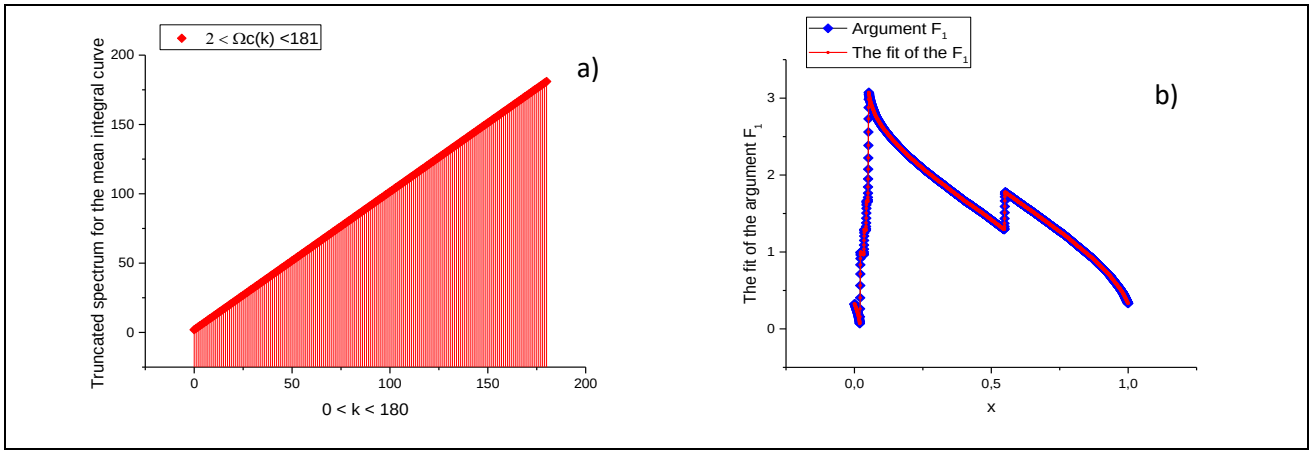



Рис. 7. Слева показан усеченный спектр, расположенный в интервале [2, 181]. Справа сплошной синей линией представлена аппроксимирующая функция Yft(t) для функции $F_1(t)$

Усеченный спектр на рис. 7 (a) достаточен для обеспечения аппроксимации с величиной относительной ошибки (определяемой выражением (4)) менее 1%. Точные значения относительной погрешности представлены в таблице 1.

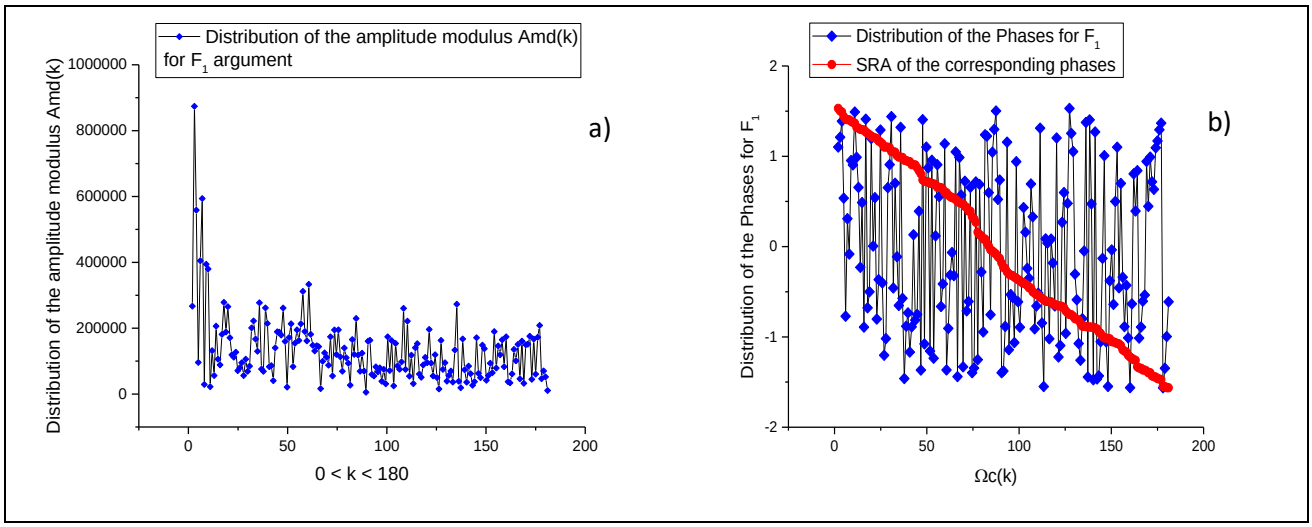



Рис. 8. Слева показаны модули амплитуд $\mathrm{Amd}_k = \sqrt{\mathrm{Ac}_k^2 + \mathrm{As}_k^2}$, справа – распределение фаз $\mathrm{Phase}_k = \tan^{-1}\left(\mathrm{As}_k / \mathrm{Ac}_k\right)$ для аргумента $F_1(t)$

Сплошная прямая линия, показанная на рис. 8 (b), соответствует последовательности ранжированных амплитуд (ПРА) для упорядоченных фаз.

Легко заметить, что она близка по виду к отрезку прямой линии. Это ещё раз свидетельствует о том, что распределение фаз практически однородно.

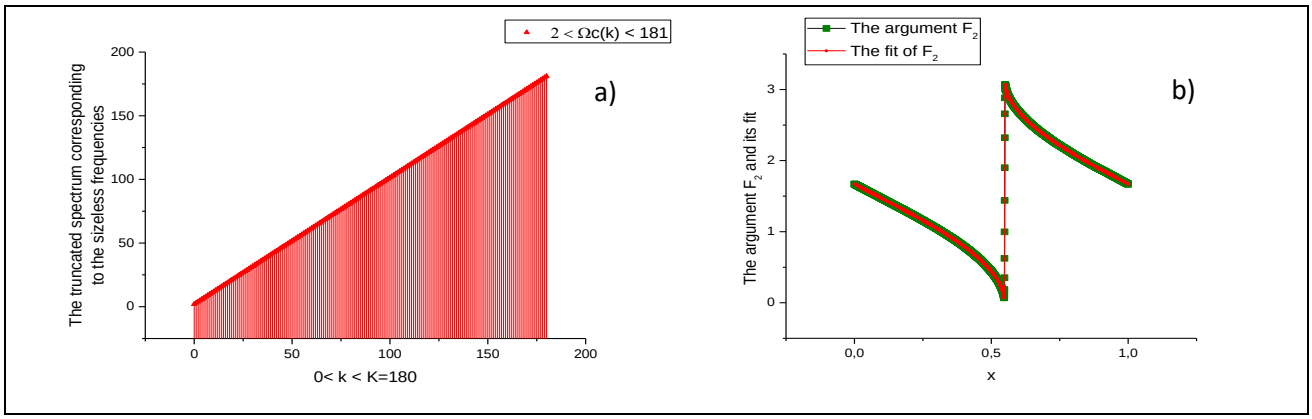



Рис. 9. Слева показан усеченный спектр, расположенный в интервале [2, 181]. Справа сплошной красной линией показана аппроксимирующая функция Yft(t) для функции $F_2(t)$

Усеченный спектр на рис. 9 (a) достаточен для обеспечения аппроксимации с величиной относительной ошибки (определяемой выражением (4)) менее 1%. Точные значения относительной погрешности представлены в таблице 1.

Сплошная прямая линия, показанная на рис. 10 (b), соответствует последовательности ранжированных амплитуд (ПРА) для упорядоченных фаз. Видно, что она близка по виду к отрезку прямой линии. Это подтверждает также, что распределение фаз практически однородно.

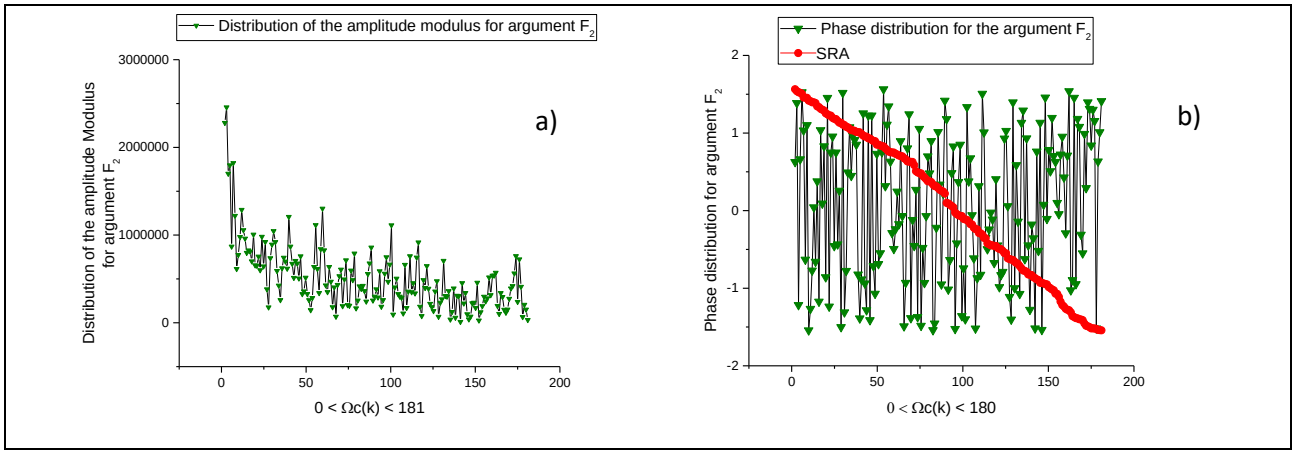



Рис. 10. Слева показан модуль амплитуд $\mathrm{Amd}_k = \sqrt{\mathrm{Ac}_k^2 + \mathrm{As}_k^2}$, справа – распределение фаз $\mathrm{Phase}_k = \tan^{-1}\left(\mathrm{As}_k / \mathrm{Ac}_k\right)$ для аргумента $F_2(t)$.

Таблица 1. Основные параметры, которые использовались для аппроксимации функций $F_p(t)$ (p = 0,1,2), соответствующих первой неделе измерений

| Функция | Fres | Ph0 | RelErr(%) | K | a | b |
|---|---|---|---|---|---|---|
| F0(t) | 4032.6694 | 0.0947 | 0.107 | 180 | 0.00446 | -0.00221 |
| F1(t) | 3901.1394 | 0.0825 | 0.079 | 180 | 3.3585 | 3.8223 |
| F2(t) | 3571.141 | 0.9045 | 0.144 | 180 | 0.8517 | 0.0839 |

Мы считаем, что полученные значения достаточно информативны для подтверждения эффективности модифицированного преобразования Фурье.

## 4. РЕЗУЛЬТАТЫ И ОБСУЖДЕНИЕ

Предложенная модификация традиционного преобразования Фурье предоставляет новые возможности для аппроксимации широкого набора случайных функций. Можно утверждать, что любая случайная функция имеет свою АЧХ, котрая находится внутри аргумента косинуса (см. выражения (2) и (3)) и даёт фактически истинный спектр для сигналов с частотно-фазовой модуляцией и второй спектр для амплитудно-модулированных сигналов. Действительно, если случайный сигнал имеет следующую структуру:

$$Sg(t) = A(t)\cos(Fr(t)) \rightarrow a\cos\big(F(t)\big) + b,$$

то правая часть случайного сигнала Sg(t) включает в себя все три типа возможных модуляций (амплитудную, частотную и фазовую). Преобразование, использованное в настоящей работе, в полной мере применимо для двух видов модуляции (частотной и фазовой) как точное, а для амплитудной модуляции может рассматриваться как приближенное. АЧХ, рассчитанная для функции F(t), может рассматриваться как полезный вторичный спектр при анализе и аппроксимации различных случайных сигналов с амплитудной модуляцией. Если амплитудная модуляция отсутствует, то зависимость (2) становится единственной и точной. Формула

(2) допускает следующее обобщение (если Sg(t) представляется в виде одной моды)

$$Sg(t) = A\cos\left(F(t)\right) + B\sin\left(F(t)\right),$$

где постоянные A и B предполагаются известными. Отсюда можно найти аргумент F(t)

$$F(t) = \cos^{-1}\left(\frac{Sg(t)}{\sqrt{A^2 + B^2}}\right) + \tan^{-1}\left(\frac{A}{B}\right). \quad (5)$$

Таким образом, выражение (5) обобщает предыдущие выражения и расширяет границы применения предложенного модифицированного преобразования Фурье.

Выделим основные полученные результаты.

1. Для случайной функции Sg(t) найдена чисто периодическая функция F(t).

2. Эта функция F(t) связана нелинейным образом с исходным случайным сигналом.

3. Предложенный подход решает задачу аппроксимации широкого класса сигналов с частотно-фазовой модуляцией.

4. Благодаря этому, широкое применение этой модификации преобразования Фурье будет полезно, в частности, для описания откликов различных сложных систем, таких как технические, финансовые, медицинские и т. д., когда простая модель отсутствует.

**ФИНАНСИРОВАНИЕ**

Работа выполнена за счет средств субсидии, выделенной Казанскому федеральному университету для выполнения государственного задания в сфере научной деятельности, проект № FZSM-2024-0004.

**СПИСОК ЛИТЕРАТУРЫ**

1. *Mertins A.* Signal Analysis: Wavelets, Filter Banks, Time-Frequency Transforms and Applications. Wiley: Chichester, UK, 1999.

2. *Arecchi F., Meucci R., Puccioni G., Tredicce J.* Experimental evidence of subharmonic bifurcations, multistability, and turbulence in a Q-switched gas laser // Phys. Rev. Lett. 1982. Vol. 49. P. 1217–1220.

3. *Chen J., Chau K., Chan C., Jiang Q.* Subharmonics and chaos in switched reluctance motor drives // *IEEE Trans. Energy Convers*. 2002. Vol. 17. P. 73–78.

4. *Lauterborn W., Cramer E.* Subharmonic route to chaos observed in acoustics // Phys. Rev. Lett. 1981. Vol. 47. P. 1445.

5. *Wilden I., Herzel H., Peters G., Tembrock G.* Subharmonics, biphonation, and deterministic chaos in mammal vocalization // Bioacoustics. 1998. Vol. 9. P. 171–196.

6. *Cohen L.* Time-Frequency Analysis. Prentice Hall Press: NJ, USA, 1995; Vol. 778.

7. *Almeida L.B.* The fractional Fourier transform and time-frequency representations // IEEE Trans. Signal Process. 1994. Vol. 42. P. 3084–3091.

8. *Sejdić E., Djurović I., Stanković L.* Fractional Fourier transform as a signal processing tool: An overview of recent developments // Signal Process. 2011. Vol. 91. P. 1351–1369.

9. *Su X., Tao R., Kang X.* Analysis and comparison of discrete fractional Fourier transforms // Signal Process. 2019. Vol.160. P. 284–298.

10. *Portnoff M.* Time-frequency representation of digital signals and systems based on short-time Fourier analysis // IEEE Trans. Acoust. Speech Signal Process. 1980. Vol. 28. P. 55–69.

11. *Qian S., Chen D.* Joint time-frequency analysis // IEEE Signal Process. Mag. 1999. Vol. 16. P. 52–67.

12. *Li M., Liu Y., Zhi S., Wang T., Chu F.* Short-time Fourier transform using odd symmetric window function // J. Dyn. Monit. Diagn. 2022. Vol. 1. P. 37–45.

13. *Hlubina P., Luňáček J., Ciprian D., Chlebus R.* Windowed Fourier transform applied in the wavelength domain to process the spectral interference signals // Opt. Commun. 2008. Vol. 281. P. 2349–2354.

14. *Kemao Q.* Windowed Fourier transform for fringe pattern analysis // Appl. Opt. 2004. Vol. 43. P. 2695–2702.

15. *Qian S., Chen D.* Discrete Gabor transform // IEEE Trans. Signal Process. 1993. Vol. 41. P. 2429–2438.

16. *Yao J., Krolak P., Steele C.* The generalized Gabor transform // IEEE Trans. Image Process. 1995. Vol. 4. P. 978–988.

17. *Zhao Z., Tao R., Li G., Wang Y.* Clustered fractional Gabor transform // Signal Process. 2020. Vol. 166. Article No. 107240.
https://doi.org/10.1016/j.sigpro.2019.107240Get rights and content

18. *Mallat S.* A Wavelet Tour of Signal Processing, 3rd ed. Academic Press: Burlington, MA, USA, 1999.

19. *Yan R., Gao R.X., Chen X.* Wavelets for fault diagnosis of rotary machines: A review with applications // Signal Process. 2014. Vol. 96. P. 1–15.

20. *Kumar A.* Wavelet signal processing: A review for recent applications // Int. J. Eng. Tech. 2020. Vol. 6. No. 6. P. 1–7.

21. *Peng Z.K., Peter W.T., Chu F.L.* A comparison study of improved Hilbert–Huang transform and wavelet transform: Application to fault diagnosis for rolling bearing // Mech. Syst. Signal Process. 2005. Vol. 19. P. 974–988.

22. *Zayed A.I.* Hilbert transform associated with the fractional Fourier transform // IEEE Signal Process. Lett. 1998. Vol. 5. P. 206–208.

23. *De Souza U.B., Escola J.P.L., da Cunha Brito L.* A survey on Hilbert–Huang transform: Evolution, challenges and solutions // Digit. Signal Process. 2021. Vol. 120. No. 4. 103292. https://doi.org/10.1016/j.dsp.2021.103292

24. *Bhattacharyya A., Singh L., Pachori R.B.* Fourier–Bessel series expansion based empirical wavelet transform for analysis of non-stationary signals // Digit. Signal Process. 2018. Vol. 78. P. 185–196.

25. *Chaudhary P.K., Gupta V., Pachori R.B.* Fourier–Bessel representation for signal processing: A review // Digit. Signal Process. 2023. Vol. 135. 103938.
https://doi.org/10.1016/j.dsp.2023.103938Get rights and content

26. *Huang N.E., Shen Z., Long S.R., Wu M.C., Shih H.H., Zheng Q., Yen N.C., Tung C.C., Liu H.H.* The empirical mode decomposition and the Hilbert spectrum for nonlinear and non-stationary time series analysis // Proc. R. Soc. Lond. A Math. Phys. Eng. Sci. 1998. Vol. 454. P. 903–995.

27. *Wu Z., Huang N.E.* Ensemble empirical mode decomposition: A noise-assisted data analysis method // Adv. Adapt. Data Anal. 2009. Vol. 1. P. 1–41.

28. *Nigmatullin R.R., Alexandrov V., Agarwal P., Jain S., Ozdemir N.* Description of Multi-Periodic Signals Generated By Complex Systems: NOCFASS – New Possibilities of the Fourier Analysis // Numerical Algebra, Control and Optimization. March 2024. Vol. 14. No. 1. P. 1–19. https://doi.org/10.3934/naco.2022008.

## NEW POSSIBILITIES OF THE FOURIER TRANSFORMATION: HOW TO DESCRIBE AN ARBITRARY FREQUENCY-PHASE MODULATED SIGNAL?

**R. R. Nigmatullin[1] [0000-0003-2931-4428], A. A. Litvinov[2] [0009-0000-3901-3704],**

**S. I. Osokin[3] [0000-0002-0699-5390]**

*[1, 2]Kazan National Research Technical University named after A.N. Tupolev, Radioelectronics and Informative Measurements Technics department, 420111 Kazan, Russia.*

*[3]Kazan Federal University, Institute of Information Technologies and Intellectual Systems, 420008 Kazan, Russia.*

[1]renigmat@gmail.com, [2]litvinov85@gmail.com, [3]s.osokin@it.kfu.ru

**Abstract**

In this paper, the authors found a transformation that is valid for any *arbitrary* signal. This transformation is strictly periodical and therefore it allows to apply the ordinary F-transformation for the fitting of the transformed signal. The most interesting application (in accordance with the author's opinion) is the fitting of the frequency-phase modulated signals that actually located inside the found transformation. This new transformation will be useful for application of the responses of different complex systems when an ordinary model is absent.

As an available data we consider meteo-data corresponding to measurements of methane concentration ($CH_4$) in atmosphere during 4 weeks of its observation. For us it is important to consider the integral (cumulative) data and find their amplitude-frequency response (AFR). If one considers each column as frequency-phase modulated signal, then AFR can be evaluated with the help of F-transformation that has the period

equals $2\pi$ that is valid for any analyzed random signal. This "universal" F-transformation allows to fit a wide set of random signals and compare them with each other in terms of their AFRs. Concluding the abstract one can say that these new possibilities of the traditional F-analysis will serve as a common tool in the armory of the methods used by researchers in data processing area.

***Keywords**: Fourier transform, random signal, frequency-phase modulated signal, amplitude-frequency response, complex systems, meteorological data, eddy covariance.*

***The list of the main abbreviations:** AFR – amplitude-frequency response, NOCFASS – Non-Orthogonal Combined Fourier Analysis of the Smoothed Signals, SRA – sequence of the range amplitudes*

**REFERENCES**

1. *Mertins A.* Signal Analysis: Wavelets, Filter Banks, Time-Frequency Transforms and Applications. Wiley: Chichester, UK, 1999.

2. *Arecchi F., Meucci R., Puccioni G., Tredicce J.* Experimental evidence of subharmonic bifurcations, multistability, and turbulence in a Q-switched gas laser // Phys. Rev. Lett. 1982. Vol. 49. P. 1217–1220.

3. *Chen J., Chau K., Chan C., Jiang Q.* Subharmonics and chaos in switched reluctance motor drives // *IEEE Trans. Energy Convers*. 2002. Vol. 17. P. 73–78.

4. *Lauterborn W., Cramer E.* Subharmonic route to chaos observed in acoustics // Phys. Rev. Lett. 1981. Vol. 47. P. 1445.

5. *Wilden I., Herzel H., Peters G., Tembrock G.* Subharmonics, biphonation, and deterministic chaos in mammal vocalization // Bioacoustics. 1998. Vol. 9. P. 171–196.

6. *Cohen L.* Time-Frequency Analysis. Prentice Hall Press: NJ, USA, 1995; Vol. 778.

7. *Almeida L.B.* The fractional Fourier transform and time-frequency representations // IEEE Trans. Signal Process. 1994. Vol. 42. P. 3084–3091.

8. *Sejdić E., Djurović I., Stanković L.* Fractional Fourier transform as a signal processing tool: An overview of recent developments // Signal Process. 2011. Vol. 91. P. 1351–1369.

9. *Su X., Tao R., Kang X.* Analysis and comparison of discrete fractional Fourier transforms // Signal Process. 2019. Vol.160. P. 284–298.

10. *Portnoff M.* Time-frequency representation of digital signals and systems based on short-time Fourier analysis // IEEE Trans. Acoust. Speech Signal Process. 1980. Vol. 28. P. 55–69.

11. *Qian S., Chen D.* Joint time-frequency analysis // IEEE Signal Process. Mag. 1999. Vol. 16. P. 52–67.

12. *Li M., Liu Y., Zhi S., Wang T., Chu F.* Short-time Fourier transform using odd symmetric window function // J. Dyn. Monit. Diagn. 2022. Vol. 1. P. 37–45.

13. *Hlubina P., Luňáček J., Ciprian D., Chlebus R.* Windowed Fourier transform applied in the wavelength domain to process the spectral interference signals // Opt. Commun. 2008. Vol. 281. P. 2349–2354.

14. *Kemao Q.* Windowed Fourier transform for fringe pattern analysis // Appl. Opt. 2004. Vol. 43. P. 2695–2702.

15. *Qian S., Chen D.* Discrete Gabor transform // IEEE Trans. Signal Process. 1993. Vol. 41. P. 2429–2438.

16. *Yao J., Krolak P., Steele C.* The generalized Gabor transform // IEEE Trans. Image Process. 1995. Vol. 4. P. 978–988.

17. *Zhao Z., Tao R., Li G., Wang Y.* Clustered fractional Gabor transform // Signal Process. 2020. Vol. 166. Article No. 107240.
https://doi.org/10.1016/j.sigpro.2019.107240Get rights and content

18. *Mallat S.* A Wavelet Tour of Signal Processing, 3rd ed. Academic Press: Burlington, MA, USA, 1999.

19. *Yan R., Gao R.X., Chen X.* Wavelets for fault diagnosis of rotary machines: A review with applications // Signal Process. 2014. Vol. 96. P. 1–15.

20. *Kumar A.* Wavelet signal processing: A review for recent applications // Int. J. Eng. Tech. 2020. Vol. 6. No. 6. P. 1–7.

21. *Peng Z.K., Peter W.T., Chu F.L.* A comparison study of improved Hilbert–Huang transform and wavelet transform: Application to fault diagnosis for rolling bearing // Mech. Syst. Signal Process. 2005. Vol. 19. P. 974–988.

22. *Zayed A.I.* Hilbert transform associated with the fractional Fourier transform // IEEE Signal Process. Lett. 1998. Vol. 5. P. 206–208.

23. *De Souza U.B., Escola J.P.L., da Cunha Brito L.* A survey on Hilbert–Huang transform: Evolution, challenges and solutions // Digit. Signal Process. 2021. Vol. 120. No. 4. Article No. 103292. https://doi.org/10.1016/j.dsp.2021.103292

24. *Bhattacharyya A., Singh L., Pachori R.B.* Fourier–Bessel series expansion based empirical wavelet transform for analysis of non-stationary signals // Digit. Signal Process. 2018. Vol. 78. P. 185–196.

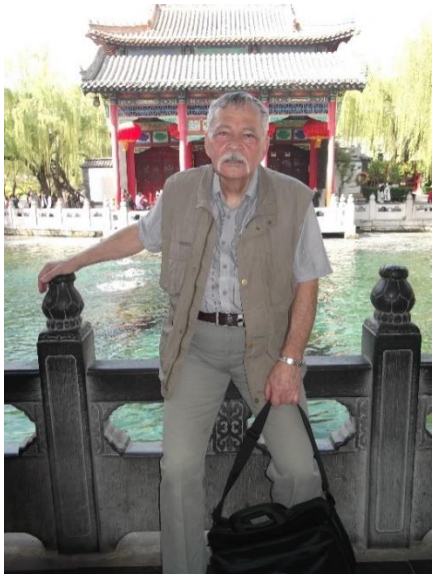
25. *Chaudhary P.K., Gupta V., Pachori R.B.* Fourier–Bessel representation for signal processing: A review // Digit. Signal Process. 2023. Vol. 135. Article No. 103938. https://doi.org/10.1016/j.dsp.2023.103938Get rights and content

26. *Huang N.E., Shen Z., Long S.R., Wu M.C., Shih H.H., Zheng Q., Yen N.C., Tung C.C., Liu H.H.* The empirical mode decomposition and the Hilbert spectrum for nonlinear and non-stationary time series analysis // Proc. R. Soc. Lond. A Math. Phys. Eng. Sci. 1998. Vol. 454. P. 903–995.

27. *Wu Z., Huang N.E.* Ensemble empirical mode decomposition: A noise-assisted data analysis method // Adv. Adapt. Data Anal. 2009. Vol. 1. P. 1–41.

28. *Nigmatullin R.R., Alexandrov V., Agarwal P., Jain S., Ozdemir N.* Description of Multi-Periodic Signals Generated By Complex Systems: NOCFASS – New Possibilities of the Fourier Analysis // Numerical Algebra, Control and Optimization. March 2024. Vol. 14. No. 1. P. 1–19. https://doi.org/10.3934/naco.2022008.

## СВЕДЕНИЯ ОБ АВТОРАХ



***НИГМАТУЛЛИН Равиль Рашидович*** – доктор физико-математических наук, профессор, преподаватель кафедры Радиоэлектроники и информационно-измерительной техники Казанского национального исследовательского университета им. А.Н. Туполева.

Подробнее: http://tre.kai.ru/staff/Nigmatullin/

***Raoul RashidovichNIGMATULLIN*** – Doctor of Phys.-Math. Science, Full Professor, Teacher of the Radioelectronics and Informative-Measurements Technics Department, Kazan National Research Technical University (KNRTU-KAI) named after A.N. Tupolev.

More details: http://tre.kai.ru/staff/Nigmatullin/

email: renigmat@gmail.com

ORCID 0000-0003-2931-4428

***ЛИТВИНОВ Александр Алексеевич*** – научный сотрудник, Институт информационных технологий и интеллектуальных систем, Казанский (Приволжский) федеральный университет, Казань, Россия.

***Alexander Alekseevich LITVINOV*** – Researcher, Institute of Information Technology and Intelligent Systems, Kazan Federal University, Kazan, Russia.

email: sharebox@bk.ru

ORCID 0009-0000-3901-3704


***ОСОКИН Сергей Игоревич*** – доцент, кандидат физико-математических наук, заместитель директора по научной деятельности Института информационных технологий и интеллектуальных систем Казанского федерального университета. Подробнее: https://kpfu.ru/Sergey.Osokin

***Sergey Igorevich OSOKIN*** – Associate Professor, Candidate of Physical and Mathematical Sciences, Deputy Director for Research at the Institute of Information Technologies and Intelligent Systems of Kazan Federal University. More details: https://eng.kpfu.ru/Sergey.Osokin

email: s.osokin@it.kfu.ru

ORCID 0000-0002-0699-5390

*Материал поступил в редакцию 14 февраля 2025 года*

УДК 004.43+004.42

# ГИБРИДНАЯ СИСТЕМА ПРОГРАММИРОВАНИЯ ДЛЯ УЧЕБНЫХ ИСПОЛНИТЕЛЕЙ НА PYTHON

**М. В. Райко**[0000-0002-6448-6471]

*НИЦ «Курчатовский институт» – НИИСИ, г. Москва, Россия*

*Московский педагогический государственный университет, г. Москва*

rayko@niisi.ru

***Аннотация***

Рассмотрена методика разработки учебных формальных исполнителей с использованием комбинированного пиктограммно-текстового интерфейса на языке программирования Python. Актуальность исследования обусловлена необходимостью совершенствования подходов к обучению алгоритмизации и программированию в школьном курсе информатики. Представлен разработанный инструментарий для создания формальных исполнителей, сочетающий наглядность пиктограмм с возможностями текстового программирования. Особое внимание уделено практическим аспектам реализации, включая использование встроенных методов Python для обработки графических и текстовых данных.

***Ключевые слова****: формальный исполнитель, визуализация, программирование, пиктограммный интерфейс, Python.*

## ВВЕДЕНИЕ

В современных условиях развития цифровой экономики и перехода к технологическому суверенитету особую актуальность приобретает проблема обучения алгоритмизации и программированию в школьном курсе информатики. Относящиеся к предмету информатика понятия и практики, которыми в обязательном порядке должны овладеть выпускники 9-х и 11-х классов, определяются рядом федеральных документов. С 1 сентября 2025 года вступает в действие приказ Министерства просвещения Российской Федерации от 09.10.2024 № 704, который определяет «проверяемые требования к результатам освоения основной образовательной программы основного общего образования» и «проверяемые элементы содержания» [1]. В этом федеральном документе в раздел «проверяемые

элементы содержания» по предмету информатика (8-й класс) включены элементы двух типов:

с одной стороны, обучаемый должен быть ознакомлен с понятием формального исполнителя и приобрести опыт составления и выполнения алгоритмов управления формальными исполнителями на компьютере с использованием базовых алгоритмических конструкций;

с другой стороны, обучаемый должен ознакомиться с одним из текстовых языков программирования (Python, C++, Java, C#, Школьный Алгоритмический Язык) и поддерживающей этот язык системой программирования (редактор текста программ, транслятор, отладчик).

Для ознакомления учащихся с понятием формального исполнителя и практикой составления программ управления такими исполнителями педагог может использовать одну из свободно распространяемых сред пиктограммного программирования: среду Scratch Junior или отечественную цифровую образовательную среду (ЦОС) ПиктоМир [2].

Однако опыт работы со школьниками, а также студентами педагогических факультетов университетов показывает, что освоенные в таких средах навыки обучаемых по составлению программ с *«использованием циклов и ветвлений»* переносятся на текстовые языки программирования с большими затруднениями. Поэтому возникает идея предоставить школьникам возможность работы с формальными исполнителями на одном из текстовых языков программирования. В качестве такого языка в настоящей статье предложен Python. В качестве минимального набора формальных исполнителей для среды Python выбраны шесть формальных исполнителей: Робот, Вертун, Двигун, Тягун, Паровозик и Ползун которые реализованы нами в среде Python в форме библиотеки, которая названа Moscow Robots (Московские роботы).

Первый из этих формальных исполнителей, Робот, хорошо известен в педагогическом сообществе, так как он впервые появился в российской системе образования в 1990 году в учебнике информатики [3]. Исполнитель Робот приведён как пример формального исполнителя в действующей сегодня Федеральной рабочей программе предмета информатика (базовый уровень), поэтому формальный исполнитель Робот регулярно используется в заданиях ОГЭ по информатике.

## МЕТОДЫ И МАТЕРИАЛЫ

### Пиктограммное программирование

Простейшие задания по освоению базовых управляющих конструкций современных языков программирования эффективнее всего осваиваются новичками в графических средах программирования. В работе [4] был предложен язык Пикто, позволяющий составлять программы управления формальными исполнителями в пиктограммной среде.

Пиктограммы стали важной визуальной особенностью современных графических компьютерных интерфейсов [5]. Эстетическая привлекательность и возможность быстрого распознавания пиктограмм делают их незаменимым элементом цифровых образовательных сред для детей, что позволяет включать образовательный контент на самых ранних этапах развития ребенка, ещё до освоения последним чтения и письма. Универсальность пиктограмм ЦОС ПиктоМир делает его доступным детям вне зависимости от национальной принадлежности [6]. Знакомая визуальная форма, обозначающая конкретный референт (объект или понятие), позволяет ребёнку-интерпретатору легко и быстро понять предъявляемое значение, вне зависимости от языка, и сосредоточиться на изучении и изобретении алгоритмов в цифровой образовательной среде, а не на долгом изучении языка программирования.

Пиктографический язык Пикто – это настоящий язык программирования, используемый в системе ПиктоМир. Редактор-компилятор языка Пикто использует представление LL(1) грамматикой (англ. Left-to-right Leftmost derivation with 1 symbol look-ahead). При этом множество терминалов может быть представлено как в пиктографическом, так и в текстовом виде. Последнее требует разработки системы обозначений для пиктограмм языка.

Однако использование вне среды этого языка затруднялось тем, что ученики, используя вложение управляющих конструкций, обучаемые регулярно, совершали синтаксические ошибки, а диагностика и исправление таких ошибок требовали освоения новичками большого объёма дополнительного материала, не связанного с содержательными алгоритмическими вопросами. Теоретически в среде программирования можно добиться, чтобы любая составленная в нём программа стала автоматически правильной, синтаксически верной. Можно даже

расширить этот язык, позволяющий составлять «безошибочные» программы при использовании арифметических и логических выражений. Например, можно разработать редактор-компилятор, не позволяющий обучаемому совершать синтаксические ошибки. На практике удалось найти более простой подход, который исключает возможность совершения синтаксических ошибок обучаемым. Этот подход, использующий новую версию языка Пикто, представлен в статье [7].

Новый подход состоит в том, чтобы наделить программу управления формальными исполнителями, состоящую из пиктограмм, априорной двумерной структурой и восстанавливать блочную структуру программы по горизонтальным и вертикальным отступам (аналог приема, используемого в языке Python). Оказалось, что такой подход позволяет рассматривать любое расположение пиктограмм в двумерной таблице как синтаксически правильную, исполнимую программу. Если планировать использовать этот подход на начальных этапах составления алгоритмов управления формальными исполнителями, то каждая процедура библиотеки Moscow Robots снабжена не только уникальным текстовым именем, но и уникальной пиктограммой, и еще две дополнительные программы включены в разрабатываемую библиотеку:

а) простейший редактор, который размещает пиктограммы в прямоугольной таблице;

б) компилятор, который превращает любую такую таблицу в корректную программу на Python, которая может быть выполнена в стандартной среде Python.

Наличие а) и б) позволяет обучаемому решать задачи управления формальными исполнителями в два этапа: сначала пиктограммными средствами создается и отлаживается алгоритм движения робота в игровой обстановке, а затем происходит переход в «настоящую» текстовую среду программирования на языке Python, в которой обучаемый может добавить работу с числовыми или символьными данными для завершения решения задачи.

**Библиотека Moscow Robots**

Набор абстрактных исполнителей[1] Moscow Robots включает 6 движущихся роботов и 4 неподвижных устройства, моделирующих одиночные числовые и логические переменные. В набор роботов входят Вертун, Двигун, Тягун, Ползун, Робот и Паровозик, которые действуют на клетчатом игровом поле (карте) и выполняют задания по программе, созданной учащимся.

На карте в зависимости от типа размещенного на ней исполнителя (робота) могут присутствовать разделяющие клетки стенки, а также различные грузы и пометки клеток. Неподвижные устройства располагаются вне карты и могут быть добавлены составителем задания к роботу любого типа.

В рамках настоящего исследования предложено текстовое представление пиктограммных команд роботов.

Далее рассмотрена практика работы в библиотеке Moscow Robots на примере робота Вертун.

**Робот Вертун**

По легенде робот Вертун отвечает за ремонт полей-клеток космодрома, представленного в виде матричной клеточной модели. Клетки бывают трёх видов и цветов, каждый цвет несет свой функционал: голубая – обычная клетка, серая – поломанная, требующая починку командой закрасить, синяя – клетка закрашенная (после ремонта).

Вертун умеет выполнять команды-приказы:

*вперёд* – сделать, если возможно, шаг вперёд;

*налево* – повернуться на 90° налево;

*направо* – повернуться на 90° направо.

При невозможности выполнить команду *вперёд* Вертун сигнализирует об этом и перестаёт выполнять дальнейшие команды (отказ).

Робот Вертун умеет выполнять команды-вопросы (с обратной связью), которые возвращают логический ответ «да» или «нет» в зависимости от клетки, в

[1] Под абстрактным исполнителем здесь понимается исполнитель с фиксированной функциональностью, с возможностью внешнего управления работой в некоторой обстановке.

которой в данный момент находится робот, и обстановки рядом с ней: впереди стена?, впереди свободно?, клетка голубая?, клетка серая?, клетка синяя?

В ЦОС ПиктоМир к каждой из таких логических команд-вопросов прилагалась команда-отрицание: клетка голубая?, клетка не голубая?, дающая противоположный результат прямому вопросу и наоборот. В производственных языках программирования необходимость такой двойственности отсутствует, так как логическое отрицание легко осуществляется средствами самого языка («за скобками»): **не** в КуМире или **not** в Python.

Для языка Python эти команды заменяются на их латинские варианты:

step_forward, turn_left, turn_right, fix_cell для команд-приказов;

way_clean, way_blocked, cell_normal, cell_broken, cell_fixed для команд-вопросов.

В текстовой кодировке, разработанной для библиотеки Moscow Robots, команды Вертуна преобразовались в следующие:

rfwd – *вперёд* (по этой команде робот делает шаг вперёд из середины одной клетки в середину соседней по направлению движения), rright – *направо* (по этой команде робот поворачивается направо на угол 90°, не выходя за пределы клетки, в которой находится), rleft – налево (по этой команде робот поворачивается налево на угол 90°, не выходя за пределы клетки, в которой находится), <rclear> – вперёди свободно?, <!rclr> – вперёди не свободно?, а также некоторое количество команд, специфичных для отдельных типов роботов, как то rfix – закрасить текущую клетку, <rcnor> – клетка нормальная, <rcbro> – клетка поломана, <rcfix> – клетка починена, вместе с их отрицаниями и аналогами для использования в циклах.

Обстановки для роботов представлены текстовыми файлами в формате JSON.

Описание обстановки на примере обстановки для Вертуна (рис. 1):

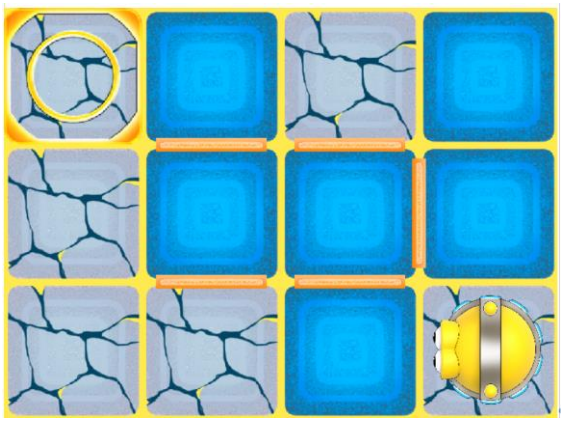


Рис. 1. Обстановка к заданию с Вертуном

```
{
    "robot": {
        "kind": 0,
        "x": 3,
        "y": 2,
        "dir": 3,
        "fpos": [0, 0]
    },
    "field": {
        "sx": 4,
        "sy": 3,
        "cells": [
            ["b", " ", "b", " "],
            ["b", " ", " ", " "],
            ["b", "b", " ", "b"]
        ],
        "vfences": [
            "   ",
            "  |",
            "   "
        ],
        "hfences": [
            " -- ",
            " -- "
        ]
    }
}
```

- Раздел robot содержит
  - целочисленное поле kind, кодирующее тип робота; 0 соответствует Вертуну.
  - целочисленные поля x и y, задающие начальное положение робота; верхней левой клетке соответствует координата x = 0, y = 0.
  - целочисленное поле dir, задающее начальную ориентацию робота; 0 – вверх, 1 – вправо, 2 – вниз, 3 – влево;
  - целочисленную пару координат fpos, задающую требуемое финальное положение робота.
- Раздел field, задающий состояние клеток и других элементов поля
  - целочисленные поля sx, sy, задающие размер поля в клетках по горизонтали и вертикали соответственно;
  - двумерный строковый массив cells, задающий закодированные состояния каждой клетки. Для вертуна элементы кодировки такие: пробел “ ” – нормальная клетка, “b” – разбитая клетка, “p” – закрашенная (отремонтированная) клетка;
  - строковый массив vfences размером sy, задающий в строках длины sx – 1 расположение вертикальных стенок внутри поля, при наличии таковых;
  - строковый массив hfences размером sy – 1, задающий в строках длины sx расположение горизонтальных стенок внутри поля, при наличии таковых.
- В выходных обстановках могут быть дополнительные поля верхнего уровня:
  - целочисленное поле alive, отвечающее за успешное функционирование робота;
  - целочисленное поле ok, отвечающее за успешное выполнение задания.

Обстановка, представленная на рис. 1, формируется библиотекой Moscow Robots с использованием графических файлов формата PNG, хранящихся в составе самой библиотеки.

**Редактор-компилятор программ, использующих библиотеку Moscow Robots**

Редактор-компилятор принято отождествлять с интегрированной средой разработки (англ. – Integrated Development Environment, IDE), которая объединяет в себе как минимум функции редактора программ и компилятора. Удобство такой среды состоит в запуске и(или) компиляции программы (программного проекта) одной кнопкой, когда не требуется покидать редактор и вручную запускать, выходя из среды разработки. В настоящее время такой инструмент допускает фоновую компиляцию программы, например, когда разработчик закончил исправлять один файл проекта и перешел к редактированию другого. Такой подход позволяет сократить время запуска проекта, поскольку большинство файлов с программным кодом уже скомпилировано автоматически. Однако, если вернуться к появлению IDE как производственного средства программиста [8], важнейшей целью являлось то, что отдельные функции среды разработки объединены общей основой: грамматикой языка программирования. То есть программист при написании программы имел поддержку от IDE в виде интерактивной диагностики, поддерживающей программный код в синтаксически-корректном виде. Компиляция программы следовала принципу, известному из управления производством и логистикой, «точно вовремя» (англ. Just-In-Time, JIT).

В 1990-е годы создание учебных интерактивных сред программирования, компьютерных практикумов требовало решения проблемы сократить время на компиляцию практически до нуля, чтобы повысить эффективность обучения. Даже в школьном алгоритмическом языке программирования, часто называемым сейчас языком КуМир [9], изменение всего одной строки в глобальных объявлениях потенциально ведёт к необходимости перекомпиляции сотен строк программы. При разработке этой среды программирования были предприняты специальные методы, такие, например, как обратные ссылки на использованные объекты, которые позволили построить реальный редактор-компилятор, когда процесс компиляции состоял в непрерывной поддержке полного соответствия текста учебной программы скомпилированному коду [10].

Несмотря на взрывной рост производительности вычислительной техники, в системах КуМир, ПиктоМир и ПиктоМир К используется тот же подход и сегодня.

Получив задание на составление программы, использующей библиотеку Moscow Robots, обучаемый может выбрать одну из двух возможностей: составить программу на языке Python с помощью текстового редактора или создать эту программу в пиктограммной форме с помощью прилагаемого редактора программ, использующего безошибочный двумерный синтаксис языка Пикто, описанный в работе [7]. Редактор-компилятор позволяет составить программу для любого Робота, располагая пиктограммы команд этого робота и ряд служебных пиктограмм в клетках таблицы ширины 6 и (предполагаемой) высоты 12. После составления программы ученик может в той же среде преобразовать программу в представление для языка Python, использующее библиотеку Moscow Robots.

Далее рассмотрено решение учебной задачи с использованием языка Пикто на примере задания для Вертуна.

Задание. Робот находится в обстановке, приведенной выше (рис. 1).

Требуется: составить программу управления Вертуном, закрашивающую все сломанные клетки (и только их) и приводящую робота в требуемое финальное положение (0, 0).

В традиционном пиктограммном синтаксисе программа может быть представлена следующим образом:

Эта же программа в безошибочном двумерном синтаксисе представляется в более компактной форме:

Редактор-компилятор переводит её в текстовое представление:

```
(6) [rclear] rfwd <rcbro> rfix
_ rright
```

а также в Python-представление:

```
import moscow_robots as mr


pitcher = 0
memory = 0
fl0 = False
fl1 = False

def proc_main():
    for _ in range(6):
        while mr.path_clear():
            mr.step_forward()
            if mr.cell_broken():
                mr.fix_cell()
        mr.turn_right()
    pass

def proc_A():
    pass

def proc_B():
    pass

def proc_C():
    pass

def proc_D():
    pass

with mr.GameVertun("a.json") as mr:
         proc_main()
```

В составе Moscow Robots есть три неподвижных «робота», имитирующих две целочисленные (pitcher и memory) и две логические (fl0, fl1) переменные. Заготовки для использования этих роботов включаются в любую программу, создаваемую с помощью указанной библиотеки. Аналогично включаются заготовки-заголовки четырёх подпрограмм без параметров proc_A, proc_B,

proc_C и proc_D. Эта программа должна быть выполнена в среде Python, что и будет решением задачи, рис. 2.

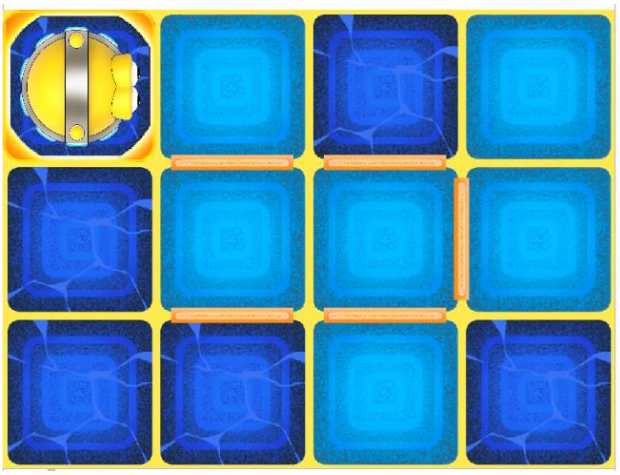


Рис. 2. Результат выполнения программы

Выполнение задания на базе библиотеки Moscow Robots может потребовать реализации некоторого алгоритма движения робота и сбора информации по маршруту с использованием переменных. В этом случае обучаемый с помощью редактора-компилятора сможет сначала составить программу движения по требуемому маршруту, а затем в обычном текстовом редакторе добавить работу с переменными языка Python.

## РЕЗУЛЬТАТЫ ЭКСПЕРИМЕНТАЛЬНЫХ ИССЛЕДОВАНИЙ

Практика работы в начальной школе и на педагогических факультетах университетов показала, что при обучении азам программирования новичков многоязыковый подход более эффективен, чем моноязыковый.

Разработанная модель обучения с использованием библиотеки Moscow Robots апробирована в многоязыковом курсе «Азы программирования» для студентов по сдвоенным специальностям – учитель информатики и учитель начальной школы. Для этого курса был разработан многоязыковый практикум, включающий более 1000 заданий разной степени сложности со встроенными механизмами автоматической проверки правильности решений. Все эти задания касались трех образовательных сред программирования: ПиктоМир -> ПиктоМир-К -> КуМир [11].

Использование дополнительно библиотеки Moscow Robots позволило не только повысить эффективность освоения предмета, но и добавить привлекательность, вариативность содержания и средств обучения этого многоязыкового курса программирования.

В рамках апробации было создано более 30 задач для управления виртуальными роботами ПиктоМир, реализованных в указанной (библиотеке Python).

Включение заданий и библиотеки Moscow Robots в практические занятия было положительно оценено студентами, изучающими курс, позволяя им обрести уверенность в своих способностях при изучении не только образовательных, но и производственных языков программирования.

Алгоритмическая схожесть большинства задач для разных ЦОС позволяет разделить алгоритмические, синтаксические и интерфейсные трудности и сосредоточиться на них в различное время при изучении материала. Это привело к росту эффективности работы студентов при самостоятельном составлении программ в рамках данного курса.

В общей сложности в рамках этого курса, состоящего из 72 полуторачасовых занятий, в 2022–2023 учебном году было предложено около 550 заданий, а в 2023–2024 учебном году – 600 заданий.

**ЗАКЛЮЧЕНИЕ**

Основываясь на результатах проведённого исследования, полученных в 2023–2024 учебном году, можно сделать вывод о том что комбинация пиктограммного и текстового программирования учебных исполнителей на языке Python в многоязыковых системах программирования дает ряд преимуществ обучаемым:

- повышает эффективность работы студентов на вводных курсах программирования, что объясняется тем, что учащиеся знакомятся как с образовательными, так и с профессиональными языками и средами программирования;

- помогает им лучше понимать материал и применять его на практике;

- включение производственной среды программирования, дополненной библиотекой Moscow Robots, повышает мотивацию студентов и побуждает их продолжать изучать программирование.

**Благодарности**

Работа выполнена в рамках темы государственного задания НИЦ «Курчатовский институт» – НИИСИ по теме № FNEF-2024-0001 (1023032100070-3-1.2.1).

**СПИСОК ЛИТЕРАТУРЫ**

1. О внесении изменений в некоторые приказы Министерства просвещения Российской Федерации, касающиеся федеральных образовательных программ начального общего образования, основного общего образования и среднего общего образования // Приказ Министерства просвещения Российской Федерации от 09.10.2024 № 704. Зарегистрирован 11.02.2025. № 81220. http://publication.pravo.gov.ru/document/0001202502120007

2. *Бесшапошников Н.О., Кушниренко А.Г., Леонов А.Г., Райко М.В., Собакинских О.В.* Цифровая образовательная среда «ПиктоМир»: опыт разработки и массового внедрения годового курса программирования для дошкольников // Информатика и образование. 2020. № 10. С. 28–40. https://doi.org/10.32517/0234-0453-2020-35-10-28-40

3. *Кушниренко А. Г., Лебедев Г. В., Сворень Р. А.* Основы информатики и вычислительной техники: учеб. пособие для 10–11-х классов общеобразовательных учреждений. М.: Просвещение; 1990. 224 с. Режим доступа: https:// www.niisi.ru/kumir/books/1.pdf

4. *Бесшапошников Н.О., Леонов А.Г.* Пиктограммный язык программирования «Пикто» // Вестник кибернетики. 2017. № 4 (28). С. 173–180.

5. *Пирс Ч. С.* Что такое знак? // Вестник Томского государственного университета. Философия. Социология. Политология. 2009. № 3. С. 88–95.

6. Раскин Д. Интерфейс: новые направления в проектировании компьютерных систем. М.: Символ-Плюс. 2005.

7. *Кушниренко А.Г., Леонов А.Г., Поликарпов С.А.* Безошибочный двумерный пиктограммный синтаксис в учебной среде программирования для дошкольников // Доклады Российской академии наук. Математика, информатика, процессы управления. 2023. Том 511. № 1. С. 13–19. https://doi.org/10.31857/S2686954323700169

8. *Teitelbaum T., Reps, T.* The Cornell program synthesizer: a syntax-directed programming environment // Communications of the ACM. 1981. Vol. 24. № 9. P. 563–573. https://doi.org/10.1145/358746.358755

9. *Леонов А.Г.* Тенденции объектно-ориентированного программирования в разработке системы КуМир // Программные продукты и системы. 2012. № 4. С. 53.

10. *Леонов А.Г., Эпиктетов М.Г.* Компоненты операционной системы для конструирования педагогических программных продуктов // Аннотированный библиографический указатель «Депонированные научные работы» ВИНИТИ РАН. 1988. № 4278-В88.

11. *Леонов А.Г., Райко М.В.* Знакомство с Цифровой Образовательной Средой «ПиктоМир-К». От знаков к тексту. (Учимся и играем) // Материалы всероссийской научно-практической конференции «STEAM образование: от дошкольника до выпускника ВУЗа». 2022.

## A COMBINATION OF PICTOGRAPHIC AND TEXT PROGRAMMING WHEN CREATING LEARNING EXECUTORS IN PYTHON

**M. V. Rayko**[0000-0002-6448-6471]

*NRC «Kurchatov Institute» – SRISA*, *Moscow*, *Russia*

*Moscow Pedagogical State University, Moscow, Russia*

rayko@niisi.ru

***Abstract***

The article discusses the methodology of developing educational algorithm executors using a combined pictographic-text interface in the Python programming language. The relevance of the research is due to the need to improve approaches to teaching algorithms and programming in the school course of computer science. The author presents the toolkit for creating algorithm executors, combining visibility of icons with text programming capabilities. Particular attention is paid to the practical

aspects of implementation, including the use of builtin Python routines for processing graphics and text.

***Keywords****: algorithm executor, visualization, programming, pictographic interface, Python.*

**REFERENCES**

1. On Amendments to Certain Orders of the Ministry of Education of the Russian Federation Concerning Federal Educational Programs for Primary General Education, Basic General Education, and Secondary General Education // Order of the Ministry of Education of the Russian Federation № 704 dated 09.10.2024. Registered 11.02.2025. № 81220. http://publication.pravo.gov.ru/document/0001202502120007

2. *Besshaposhnikov N.O., Kushnirenko A.G., Leonov A.G., Raiko M.V., Sobakinskikh O.V.* Digital educational environment PictoMir: Experience of development and mass implementation of an annual programming course for preschoolers // Informatics and Education. 2020. № 10. P. 28–40.
https://doi.org/10.32517/0234-0453-2020-35-10-28-40

3. *Kushnirenko A. G., Lebedev G. V., Svoren R. A*. Fundamen tals of informatics and computer engineering: Textbook for 10–11th grade educational institutions. Moscow, Prosveshhenie; 1990. 224 p. (In Russian.) Available at:
https://www. niisi.ru/kumir/books/1.pdf]

4. *Besshaposhnikov N.O., Leonov A.G.* Pictographic programming language Pikto // Proceedings in Cybernetics. 2017. № 4 (28). S. 173–180.

5. *Peirce C.S.* What Is a Sign? // The Essential Peirce. Vol. 2. P. 4–10. Peirce Edition Project. https://peirce.indianapolis.iu.edu/

6. *Raskin J.* The Humane Interface: New Directions for Designing Interactive Systems. 2000. Addison-Wesley Professional.

7. *Kushnirenko A.G., Leonov A.G., Polikarpov S. A.* Error-free 2D pictogrammic syntax in a programming learning environment for preschool children // Proceedings of the Russian Academy of Science. Mathematics, informatics, control processes. 2023. Vol. 511. № 1. P. 13–19. https://doi.org/10.31857/S2686954323700169

8. *Teitelbaum T.,Reps, T.* The Cornell program synthesizer: a syntax-directed programming environment // Communications of the ACM. 1981. Vol. 24. № 9. P. 563–573. https://doi.org/10.1145/358746.358755

9. *Leonov A.G.* Object-oriented programming trends in the development of the KuMir programming environment // Software products and systems. 2012. № 4. P. 53.

10. *Leonov A.G., Epiktetov M.G.* Operating system components for designing educational software products // Annotated bibliographic index "Deposited scientific papers" of VINITI RAS. 1988. № 4278-B88.

11. *Leonov A.G., Rayko M.V.* Introduction to the PictoMir-K digital educational environment. From icons to text. (Learn and play) // Proceedings of the All-Russian scientific and practical conference "STEAM education: from preschooler to university graduate". 2022.

**СВЕДЕНИЯ ОБ АВТОРЕ**

***РАЙКО Миля Вячеславовна***. Научный сотрудник, старший преподаватель кафедры ДПО НИЦ «Курчатовский институт» – НИИСИ, преподаватель Института детства МПГУ.

Области исследований: информатика, информационные системы, робототехнические системы, технологии в образовании.

***Milya V. RAYKO.*** Researcher, senior lecturer of the Department of Additional Professional Education of the NRC «Kurchatov Institute» – SRISA, teacher of the Institute of Childhood of Moscow State University.

Research areas: computer science, information systems, robotic systems, technologies in education.

email: rayko@niisi.ru

ORCID: 0000-0002-6448-6471

*Материал поступил в редакцию 5 апреля 2025 года*

УДК 004.932.2

# ОЦЕНКА ФЛУКТУАЦИОННЫХ ХАРАКТЕРИСТИК РАСПРЕДЕЛЁННЫХ ОБЪЕКТОВ НА ОСНОВЕ ОПТИЧЕСКОГО ПОТОКА

**А. М. Синица**[0000-0001-9869-4909]

*Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В. И. Ульянова (Ленина)*

amsinitca@etu.ru

***Аннотация***

Предложен метод оценки флуктуационных характеристик распределённых объектов на основе флуктуационного анализа в предположении, что оценка оптического потока является аналогом приращений случайного блуждания. Достоверность и применимость метода проверены в двух вычислительных экспериментах. Первый эксперимент посвящён анализу броуновского движения компактного объекта. Во втором подтверждена адекватность метода для оценки динамических характеристик пространственно-распределённого флуктуирующего объекта. В рамках обоих экспериментов контроль показателя Херста осуществлен методом флуктуационного анализа с исключением тренда. Полученные результаты указывают на потенциальную применимость метода и необходимость его дальнейшей оптимизации с целью повышения робастности.

***Ключевые слова**: флуктуационный анализ, компьютерное зрение, распределенные объекты.*

## ВВЕДЕНИЕ

Классический флуктуационный анализ является результатом объединения наработок в области аномальной диффузии [1–3] и R/S анализа [4]. Будучи изначально широко известным в статистической физике, этот подход к анализу временных рядов начиная с конца XX века нашёл свое применение в анализе последовательностей ДНК нуклеотидов [5], подвижности клеток [6, 7], климатических систем [8], движений модельных организмов [9–12].

Так, например, недавно была предложена автоматизированная методология идентификации и параметризации модели и разработан алгоритм, основанный на анализе флуктуаций с исключённым трендом (DFA) [5] и частичных кросс-корреляций с исключённым трендом (DPCCA) [13] для оценки свойств подвижности рыб *D. rerio* в тесте в аквариуме [14], а также крыс *R. Norvegicus* [15] и мышей *M. musculus* [16] в экспериментах в открытом поле соответственно. Результаты этих работ показали, что полный набор обычно оцениваемых параметров движения может быть успешно воспроизведён с помощью упрощённой модели случайного блуждания с единственным переменным параметром, при этом их оценка на основе модельно-параметрического подхода уменьшает внутригрупповые дисперсии по сравнению с прямыми измерениями с использованием зашумлённых видеозаписей в более чем половине рассмотренных случаев, что в целом приводит к повышению точности диагностики [14].

Кроме того, существенным ограничением традиционных методов флуктуационного анализа является требование наличия траекторной информации, что в случае безмаркерного дистанционного наблюдения за объектами, например, с использованием видеозаписи является вычислительно сложной задачей, к тому же не всегда реализуемой в силу различных причин (например, перекрытия объектов на изображениях). Кроме того, такие временные ряды не всегда представляется возможным восстановить, например, в задачах дистанционного зондирования водоемов, лесных массивов и других флуктуирующих объектов наблюдения, где интерес представляют не столько траектория движения, сколько характер относительных флуктуаций. Это также не всегда возможно в задачах, где отслеживание траекторий отдельных объектов затруднено, например, в плотных клеточных популяциях или при анализе крупных людских скоплений.

В задачах анализа последовательностей изображений используется ряд специализированных инструментов, таких как варианты алгоритма обнаружения и отслеживания краев Shi-Tomasi-Kanade (STK), хорошо известного из контекста видеоаналитики [17]. Представляя собой комбинацию двух классических методов компьютерного зрения, извлечения признаков Shi-Tomasi [18] и оптического потока Lucas-Kanade [19], алгоритм STK обеспечивает в значительной степени универсальный и мощный инструмент отслеживания различных объектов, применимый даже к необработанным сериям изображений, хотя это обычно приводит

к появлению ложных треков. Однако плотный оптический поток в каждой точке сам по себе аналогичен случайному блужданию, что потенциально позволяет избегать промежуточного шага со сложными и не всегда робастными методами отслеживания траекторий и напрямую применять методы флуктуационного анализа, что позволит анализировать не только компактные, но и пространственно-распределённые объекты.

Исходя из вышесказанного, в работе предложен оригинальный подход к анализу динамических характеристик распределённых объектов на основе флуктуационного анализа оптического потока.

Статья организована следующим образом: во введении изложены предпосылки к работе, в разделе «Методы» приведены исходные методы, а также предложенный подход к анализу оптического потока; в следующем разделе «Результаты» представлены результаты применения метода в контролируемых условиях имитационного моделирования, наконец, в разделе «Заключение» подведены итоги и обобщены результаты проведенного исследования.

## МЕТОДЫ

### Флуктуационный анализ временных последовательностей

Как уже было сказано, классический подход к флуктуационному анализу является результатом объединения наработок в области аномальной диффузии [1–3] и R/S анализа [4] и представляет собой подход, когда наблюдаемую траекторию $X$ разбивают на фрагменты, каждый из которых рассматривают как самостоятельную реализацию задачи случайного блуждания, а именно используют следующую последовательность шагов.

1. Последовательность $X$ разделяют на $K_S$ окон $X_{vS}$ с заданными шагом и размером $S$, где $S$ — масштаб флуктуационной функции.

2. Для каждого окна с номером $v$ и размером $S$ вычисляют квадраты отклонений между значениями первого и последнего отсчётов окна, оценивая таким образом квадратичное смещение (SD) за время $S$:

$$F_v^2(S) = (X_{vS}[0] - X_{vS}[-1])^2,$$

где $X_{vS}[0]$ и $X_{vS}[-1]$ — первый и последний отсчёты в окне $X_{vS}$ соответственно.

3. Оценивают флуктуационную функцию как квадратный корень из среднего смещения на заданном масштабе времени:

$$F(S) = \sqrt{1/K_S \sum_{v=1}^{K_S} F_v^2(S)}.$$

Стоит отметить, что, будучи аналогичным R/S-анализу, этот подход является базовым и, как следствие, обладает большим количеством недостатков. Так, метод чувствителен к длине выборки, обладает невысокой точностью и не работает на нестационарных сигналах (с показателем Херста больше 1).

Но, несмотря на то что предложенный далее метод анализа оптического потока является расширением метода, описанного выше, в качестве референса при оценке показателя Хёрста для второго вычислительного эксперимента используем анализ флуктуаций с исключённым трендом (DFA) [5] на базе следующей процедуры.

1. На первом шаге последовательность $X$ разделим на $K_S$ окон $X_{vS}$ с заданными шагом и размером $S$, где $S$ — масштаб флуктуационной функции.

2. В каждом окне с номером $v$ и размером $S$ проведём аппроксимацию полиномиальной функцией $P_{vS}$ (мы используем второй порядок, как и рекомендовано в оригинальной работе) и оценим остатки последовательности относительно аппроксимации: $Y_{vS} = X_{vS} - P_{vS}$.

3. Для каждого окна оценим дисперсию флуктуаций:

$$F_v^2(S) = \frac{1}{S}\sum_{i=0}^{S}|Y_{vS}|^2.$$

4. Оценим флуктуационную функцию как квадратный корень из средней дисперсии флуктуаций на заданном масштабе времени:

$$F_D(S) = \sqrt{1/K \sum_{v=1}^{K_S} F_v^2(S)}.$$

Нетрудно заметить, что DFA со степенью полиномиальной функции 0 сводится к предыдущему алгоритму. Кроме того, отметим, что классический подход включает в себя кумулятивную сумму на первом шаге (интегрирование), который опущен в силу того, что в работе анализируются траектории, а не приращения.

**Флуктуационный анализ оптического потока**

Анализ оптического потока является мощным набором методов, способных извлекать и количественно оценивать динамические характеристики для заданных областей изображения, что позволяет использовать более продвинутые приложения, включая отслеживание не только компактных, но и распределённых объектов. В последние годы появились сложные алгоритмы анализа оптического потока, такие как Farnebäck [20], TVL-1 [21] и некоторые другие, отличающиеся достаточными показателями качества.

На базе классического подхода к флуктуационному анализу мы предлагаем методику, основанную на оценке маскированного оптического потока. В пределах на маскированной области производим вычисление плотного оптического потока, например, с использованием метода, предложенного G. Farnebäck [20], при этом для упрощения вычислений горизонтальную и вертикальную составляющие оптического потока комбинируем в комплексное число, формируя таким образом трёхмерный массив комплексных чисел $X(x, y, t)$, где две оси — пространственные и одна – временная. Таким образом, имеем $w \times h$ временных рядов $\{x_i^{1,1}\}, \{x_i^{1,2}\}, \dots, \{x_i^{1,h}\}, \dots, \{x_i^{w,h}\}$, где $i = 1, 2, 3, \dots, N$. Каждый временной ряд принимаем за приращения случайного блуждания. На этом основании путём интегрирования (суммирования) оцениваем блуждание на масштабе времени $S$ в точке $w, h$ в момент времени $k$:

$$P_k^{w,h}(S) = \sum_{i=k}^{k+S} x_i^{w,h}.$$

Далее, для каждого рассматриваемого $S$ оцениваем флуктуационную характеристику $F(s)$ как среднее квадратичное отклонение (root-mean-square displacement, RMSD) блуждания во временном окне:

$$F(S) = \sqrt{\frac{\sum_{i,j}\left|P_k^{i,j}(S) - \overline{P_k^{i,j}(S)}\right|^2}{N}}.$$

Стоит отметить, что в некоторых контекстах упомянутая ранее функция используется без извлечения корня, то есть вместо среднего квадратичного отклонения используется момент второго порядка (mean-square displacement $MSD$).

## РЕЗУЛЬТАТЫ

Поскольку предложенный метод основан на классическом подходе к флуктуационному анализу, но применяется к анализу распределённых объектов, ожидается, что его недостатки будут сопоставимы с недостатками базового метода, а также могут проявляться недостатки, связанные с дополнительной процедурой взятия оптического потока. Поэтому мы провели два эксперимента для проверки полезности предложенного метода.

### Эксперимент 1. Броуновское движение

Для подтверждения эквивалентности оценки флуктуационных характеристик на основе оптического потока и траекторий был поставлен следующий эксперимент: реализована агентная модель броуновского движения, на основе которой получены видеозаписи движения частиц и соответствующие траектории (рис. 1).

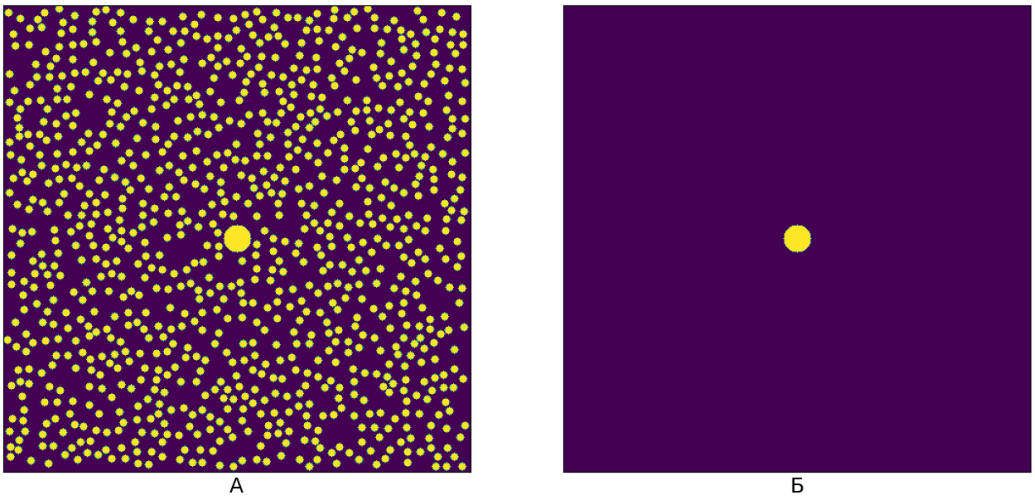
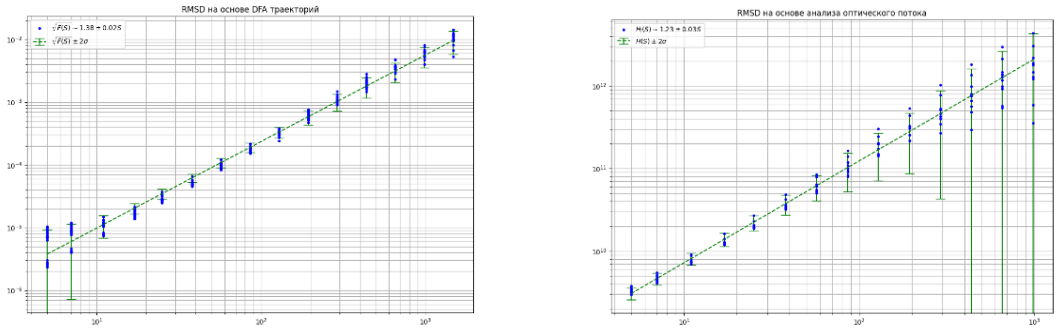



Рис. 1. Визуализация кадров в вычислительном эксперименте 1 «Броуновское движение». (А) Исходный кадр из модели, (Б) маскированный кадр для анализа поведения целевого объекта

Размеры поля, больших ($r_l$), малых ($r_s$) частиц и их плотность ($\rho$) в рамках эксперимента приняты безразмерными и равными $1 \times$ 1, 0.1, 0.01 и 1 соответственно. Масса частиц определялась, исходя из сферической формы, по формуле $4/3\,\pi\, r^3 \times \rho$. Столкновения частиц полагались абсолютно упругими. Начальные условия эксперимента: большая частица расположена в центре поля (координаты $[0.5, 0.5]$) и имеет нулевую скорость, малые частицы располагаются случайно и

имеют случайные нормальное распределение амплитуды скорости и равномерное распределение начального угла. Результатом моделирования являются последовательность кадров, соответствующих временным отсчётам, и истинные траектории объектов.




А Б

Рис. 2. Результаты оценки флуктуационной функции в эксперименте 1 «Броуновское движение». (А) методом DFA, (Б) по оптическому потоку

Так как имитационная модель описывает броуновское движение, типичным процессом диффузии, описанным Эйнштейном и Смолуховским [2, 3], где MSD является линейным во времени, а именно $MSD(S) = 2dDS$, где $d$ — число измерений, а $D$ — коэффициент диффузии.

Для более стабильной оценки показателей флуктуационных характеристик эксперимент по моделированию броуновского движения был повторен 10 раз. Полученные траектории были проанализированы методом флуктуационного анализа с исключением тренда (DFA), как описано ранее, и предложенным методом на основе оптического потока.

Так как в данном эксперименте объектом анализа является подвижный объект, а не поле, для селекции оптического потока только интересующей частицы исходные кадры видео были маскированы, как показано на рис. 1Б, обеспечивая таким образом нулевой оптический поток в пикселях, не принадлежащих объекту.

Поскольку движения по осям рассматривались как независимые, для траекторного анализа получено 20 реализаций броуновского движения. Результаты вычислительного анализа траекторий приведены на рис. 2А, показатель Хёрста,

оценённый DFA, равен $1.24$. В силу того, что оптический поток рассматривался совместно с направлением, для анализа было доступно 10 последовательностей оптического потока. Результаты анализа с использованием предложенного алгоритма представлены на рис. 2Б. Видно, что показатель Хёрста примерно равен $1.23$, что сопоставимо со значениями, полученными с помощью DFA.

**Эксперимент 2. Флуктуации распределённых объектов**

В рамках второго эксперимента была оценена применимость предложенного метода для анализа флуктуаций распределённых объектов. Примеры таких объектов встречаются повсеместно в биомедицинских изображениях, например клеточные колонии в микромире и различные водные объекты или лесные массивы в ДЗЗ. Стоит отметить, что прямая оценка флуктуационных характеристик таких объектов нередко затруднена, так как для этого при классическом подходе их флуктуации необходимо параметризовать, что не всегда реализуемо.

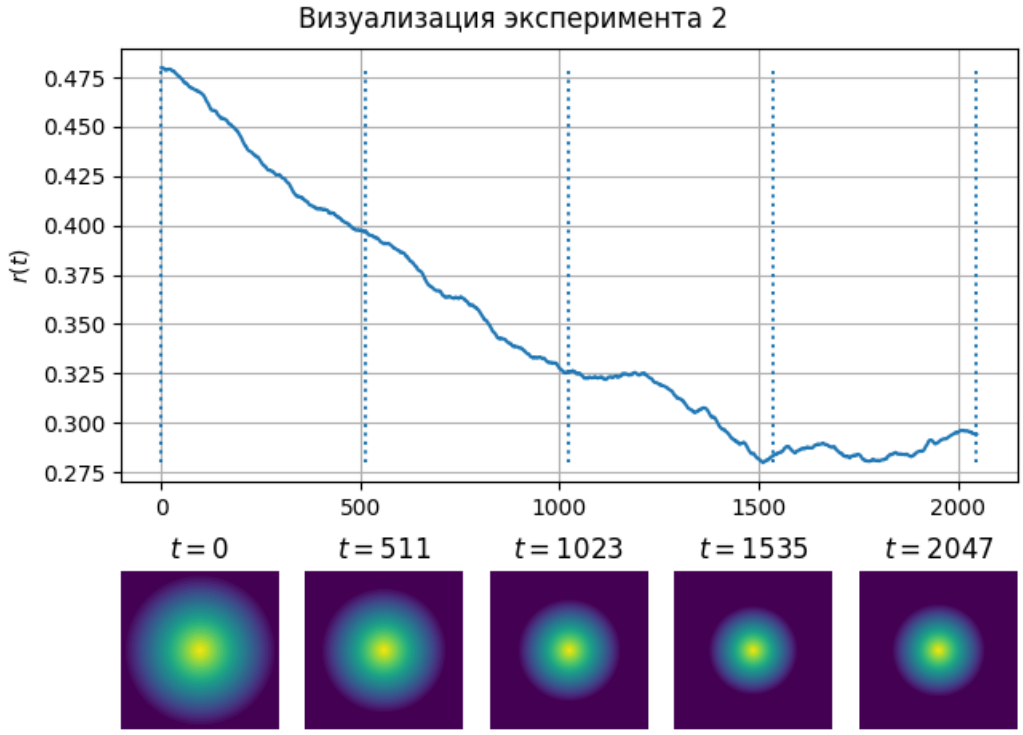



Рис. 3. Визуализация эксперимента 2. Вверху — реализация случайной зависимости радиуса от времени при $h = 0.9$, внизу — визуализация кадра в моменты времени $t_i$

В качестве объекта флуктуаций для эксперимента был выбран круг с градиентной окраской, имитирующий «озеро», а флуктуирующей переменной является радиус ($r$), имеющий функциональную зависимость от уровня «воды» (глубины) и являющийся реализацией случайной величины с заданным показателем Хёрста ($h$). На рис. 3 показаны пример такой реализации случайной величины с $h = 0.9$ и кадры из соответствующего ей ряда изображений.

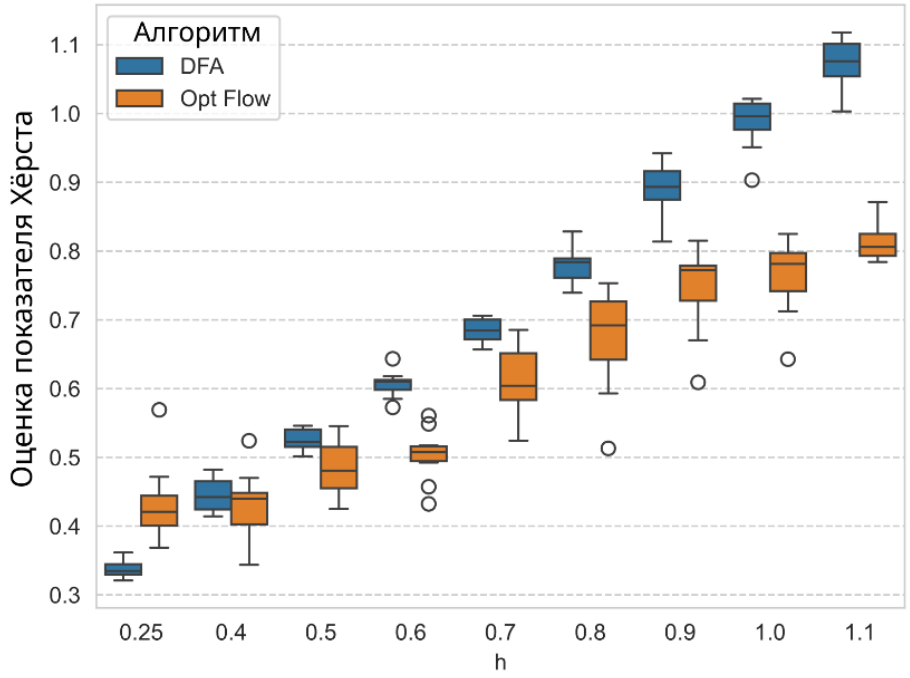



Рис. 3. Боксовые диаграммы, полученные при статистическом анализе результатов оценки показателей Хёрста с помощью алгоритма DFA по данным истинной траектории флуктуации и предложенным методом на основе оптического потока (Opt Flow)

Эксперимент был организован следующим образом. Для каждого из рассматриваемых показателей Хёрста $h = [0.25, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 1.1]$ 10 раз выполнялась следующая последовательность шагов.

**Шаг 1.** Синтезируется реализация $R_{h,i}$ случайной величины длины $T = 2048$ и показателем Хёрста $h$ с помощью метода, предложенного N. Kasdin [23], а именно с использованием фильтрации сигнала с нормальным распределением

с помощью авторегрессионного фильтра с коэффициентами, определёнными следующим образом:

$$a_1 = 1$$
$$a_k = (k - 1 - \frac{2h-1}{2})\frac{a_{k-1}}{k}$$

где $a_k, k = [1, 2, \ldots, T]$ — коэффициенты авторегрессионного фильтра, $T$ — максимальная длина фильтра.

**Шаг 2.** По заданной реализации $R_{h,i}$ синтезируется видеоряд с пульсирующим кругом с градиентной заливкой, где радиус круга на шаге $t$ определяется как $r_{h,i}(t) = R_{h,i}(t)$.

**Шаг 3.** Для контроля показателя Хёрста производится контроль путем оценки ФФ $F_{h,i}^{DFA}(S)$ методом DFA.

**Шаг 4.** Для синтезированного видеоряда оценивается ФФ ($F_{h,i}^{OF}(S)$), как описано ранее, на масштабах $S = [5, 7, 11, 17, 25, 38, 57, 86, 129, 194, 291]$ и оценкой оптического потома с использованием метода, предложенного G. Farnebäck [20].

На рис. 4 показаны боксовые диаграммы, полученные при статистическом анализе результатов оценки показателей Хёрста с помощью алгоритма DFA по данным истинной траектории флуктуации и предложенным методом на основе оптического потока (Opt Flow). Видно, что оценка показателя Хёрста имеет несколько большую дисперсию, чем более устойчивый метод DFA, а также имеет тенденцию к занижению показателя. Однако на всём диапазоне оценённых показателей Хёрста метод на основе оптического потока монотонно возрастает, сохраняя пропорциональность по отношению к результатам, полученным с помощью DFA, что говорит о том, что оценка показателя Хёрста на основе оптического потока возможна.

**ЗАКЛЮЧЕНИЕ**

Предложен метод анализа динамики распределённых объектов на основе флуктуационного анализа оптического потока. Экспериментально показана зависимость оцененного показателя Хёрста от заданного при синтезе и определяемого альтернативными методами, что демонстрирует возможность использования такого подхода для оценки динамических характеристик и идентификации

параметров модели движения объектов. Однако выявлены недостатки, в частности высокая чувствительность к длине видеоряда, значительная дисперсия оценки по сравнению с более продвинутым базовым методом. Результаты, полученные для распределенного объекта, согласуются по крайней мере для случая стационарного ряда с положительно коррелированными отсчётами, что соответствует ограничениям на оцениваемый показатель Херста $0.5 < H < 1.0$. Также к недостаткам наивной реализации алгоритма можно отнести высокое потребление памяти.

Выявленные недостатки метода, по всей видимости, объясняются его чувствительностью к качеству оптического потока, что потенциально потребует больших как пространственного, так и временного разрешений изображений исследуемых объектов, а также недостатками классического подхода к флуктуационному анализу, взятого за прототип, который имеет ограничения на оценку показателей Хёрста >1.0.

Несмотря на существенные недостатки предложенного метода, это направление работы является перспективным, так как потенциально открывает возможность анализа динамики распределённых объектов методами, устойчивыми к нестационарности. В частности, на следующих этапах работы целесообразно адаптировать для анализа оптического потока упомянутый в работе метод DFA или другие продвинутые версии алгоритмов флуктуационного анализа, что потенциально способно устранить большинство недостатков базового метода.

**БЛАГОДАРНОСТИ**

Выражаю благодарность Богачеву Михаилу Игоревичу за проявленный интерес к исследованию, значимые замечания и советы при оформлении статьи.

**СПИСОК ЛИТЕРАТУРЫ**

1. *Fick A.* Ueber Diffusion // Ann Phys. John Wiley & Sons, Ltd, 1855. Vol. 170, No. 1. P. 59–86.

2. *Einstein A.* Über die von der molekularkinetischen Theorie der Wärme geforderte Bewegung von in ruhenden Flüssigkeiten suspendierten Teilchen // Ann Phys. 1905. Vol. 4, t. 17.

3. *von Smoluchowski M.* Zur kinetischen Theorie der Brownschen Molekularbewegung und der Suspensionen // Ann Phys. John Wiley & Sons, Ltd, 1906. Vol. 326, No. 14. P. 756–780.

4. *Hurst H.E.* Long-Term Storage Capacity of Reservoirs // Transactions of the American Society of Civil Engineers. American Society of Civil Engineers, 1951. Vol. 116, No. 1. P. 770–799.

5. *Peng C.K. et al.* Mosaic organization of DNA nucleotides // Phys. Rev. E. American Physical Society, 1994. Vol. 49, No. 2. P. 1685.

6. *Selmeczi D. et al.* Cell motility as persistent random motion: Theories from experiments // Biophys J. Biophysical Society, 2005. Vol. 89, No. 2. P. 912–931.

7. *Huda S. et al.* Lévy-like movement patterns of metastatic cancer cells revealed in microfabricated systems and implicated in vivo // Nature Communications 2018 9:1. Nature Publishing Group, 2018. Vol. 9, No. 1. P. 1–11.

8. *Bogachev M.I., Eichner J.F., Bunde A.* Effect of nonlinear correlations on the statistics of return intervals in multifractal data sets // Phys. Rev. Lett. American Physical Society, 2007. Vol. 99, No. 24. P. 240601.

9. *Reynolds A.M.* Scale-free animal movement patterns: Lévy walks outperform fractional Brownian motions and fractional Lévy motions in random search scenarios // J. Phys. A Math. Theor. IOP Publishing, 2009. Vol. 42, No. 43. P. 434006.

10. *Bearup D. et al.* Revisiting Brownian motion as a description of animal movement: a comparison to experimental movement data // Methods Ecol. Evol. John Wiley & Sons, Ltd, 2016. Vol. 7, No. 12. P. 1525–1537.

11. *Torney C.J., Morales J.M., Husmeier D.* A hierarchical machine learning framework for the analysis of large scale animal movement data // Mov. Ecol. BioMed Central Ltd, 2021. Vol. 9, No. 1. P. 1–11.

12. *Hooten M.B., Johnson D.S.* Basis Function Models for Animal Movement // J Am Stat Assoc. Taylor & Francis, 2017. Vol. 112, No. 518. P. 578–589.

13. *Yuan N. et al.* Detrended Partial-Cross-Correlation Analysis: A New Method for Analyzing Correlations in Complex System // Scientific Reports. Nature Publishing Group, 2015. Vol. 5, No. 1. P. 1–7.

14. *Bogachev M.I. et al.* Understanding the complex interplay of persistent and antipersistent regimes in animal movement trajectories as a prominent characteristic of their behavioral pattern profiles: Towards an automated and robust model based quantification of anxiety test dat // Biomed Signal Process Control. 2023. Vol. 81.

15. *Lyanova A.I. et al.* Animal Movement Pattern Model Identification based on Detrended Fluctuation Analysis // 2022 IEEE International Multi-Conference on Engineering, Computer and Information Sciences, SIBIRCON 2022.

16. *Bogachev M. et al.* Video-based marker-free tracking and multi-scale analysis of mouse locomotor activity and behavioral aspects in an open field arena: A perspective approach to the quantification of complex gait disturbances associated with Alzheimer's disease // Front Neuroinform. 2023. Vol. 17.

17. *Gonzalez J.J. et al.* Robust tracking and segmentation of human motion in an image sequence // ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing – Proceedings. Institute of Electrical and Electronics Engineers Inc., 2003. Vol. 3. P. 29–32.

18. *Shi J., Tomasi C.* Good features to track // Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. Publ. by IEEE, 1994. P. 593–600.

19. *Lucas B.D., Kanade T.* An Iterative Image Registration Technique with an Application to Stereo Vision. 1981. Vol. 2. P. 674–679.

20. *Farnebäck G.* Two-Frame Motion Estimation Based on Polynomial Expansion // Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). Springer, Berlin, Heidelberg, 2003. Vol. 2749. P. 363–370.

21. *Guan S., Li H., Zheng W.S.* Unsupervised learning for optical flow estimation using pyramid convolution LSTM // Proc (IEEE Int Conf Multimed Expo). IEEE Computer Society, 2019. Vol. 2019-July. P. 181–186.

22. *Hurst H.E.* A Suggested Statistical Model of some Time Series which occur in Nature // Nature 1957 180:4584. Nature Publishing Group, 1957. Vol. 180, No. 4584. P. 494–494.

23. *Kasdin N.J.* Discrete Simulation of Colored Noise and Stochastic Processes and 1/fα Power Law Noise Generation // Proceedings of the IEEE. 1995. Vol. 83, No. 5. P. 802–827.

## FLUCTUATIONAL ANALYSIS OF DISTRIBUTED OBJECTS BASED ON OPTICAL FLOW

**A. M. Sinitca**[0000-0001-9869-4909]

*Saint Petersburg Electrotechnical University "LETI", Department of Radio Engineering Systems.*

amsinitca@etu.ru

***Abstract***

The paper proposes a method for estimating the fluctuation characteristics of distributed objects based on the fluctuation analysis and the assumption that the optical flow estimate is equivalent to random-walk increments. The reliability and applicability of the proposed method are evaluated in two computational experiments. The first experiment analyses the Brownian motion of a compact object. The second evaluates the adequacy of the method for estimating the dynamic characteristics of a spatially distributed fluctuating object. In both experiments, the Hurst exponent has been validated using detrended fluctuation analysis (DFA). The results obtained indicate the applicability of the method and the need to improve its robustness.

***Keywords****: fluctuation analysis, computer vision, distributed objects.*

**REFERENCES**

1. *Fick A.* Ueber Diffusion // Ann Phys. John Wiley & Sons, Ltd, 1855. Vol. 170, No. 1. P. 59–86.

2. *Einstein A.* Über die von der molekularkinetischen Theorie der Wärme geforderte Bewegung von in ruhenden Flüssigkeiten suspendierten Teilchen // Ann Phys. 1905. Vol. 4, t. 17.

3. *von Smoluchowski M.* Zur kinetischen Theorie der Brownschen Molekularbewegung und der Suspensionen // Ann Phys. John Wiley & Sons, Ltd, 1906. Vol. 326, No. 14. P. 756–780.

4. *Hurst H.E.* Long-Term Storage Capacity of Reservoirs // Transactions of the American Society of Civil Engineers. American Society of Civil Engineers, 1951. Vol. 116, No. 1. P. 770–799.

5. *Peng C.K. et al.* Mosaic organization of DNA nucleotides // Phys. Rev. E. American Physical Society, 1994. Vol. 49, No. 2. P. 1685.

6. *Selmeczi D. et al.* Cell motility as persistent random motion: Theories from experiments // Biophys J. Biophysical Society, 2005. Vol. 89, No. 2. P. 912–931.

7. *Huda S. et al.* Lévy-like movement patterns of metastatic cancer cells revealed in microfabricated systems and implicated in vivo // Nature Communications 2018 9:1. Nature Publishing Group, 2018. Vol. 9, No. 1. P. 1–11.

8. *Bogachev M.I., Eichner J.F., Bunde A.* Effect of nonlinear correlations on the statistics of return intervals in multifractal data sets // Phys. Rev. Lett. American Physical Society, 2007. Vol. 99, No. 24. P. 240601.

9. *Reynolds A.M.* Scale-free animal movement patterns: Lévy walks outperform fractional Brownian motions and fractional Lévy motions in random search scenarios // J. Phys. A Math. Theor. IOP Publishing, 2009. Vol. 42, No. 43. P. 434006.

10. *Bearup D. et al.* Revisiting Brownian motion as a description of animal movement: a comparison to experimental movement data // Methods Ecol. Evol. John Wiley & Sons, Ltd, 2016. Vol. 7, No. 12. P. 1525–1537.

11. *Torney C.J., Morales J.M., Husmeier D.* A hierarchical machine learning framework for the analysis of large scale animal movement data // Mov. Ecol. BioMed Central Ltd, 2021. Vol. 9, No. 1. P. 1–11.

12. *Hooten M.B., Johnson D.S.* Basis Function Models for Animal Movement // J Am Stat Assoc. Taylor & Francis, 2017. Vol. 112, No. 518. P. 578–589.

13. *Yuan N. et al.* Detrended Partial-Cross-Correlation Analysis: A New Method for Analyzing Correlations in Complex System // Scientific Reports. Nature Publishing Group, 2015. Vol. 5, No. 1. P. 1–7.

14. *Bogachev M.I. et al.* Understanding the complex interplay of persistent and antipersistent regimes in animal movement trajectories as a prominent characteristic of their behavioral pattern profiles: Towards an automated and robust model based quantification of anxiety test dat // Biomed Signal Process Control. 2023. Vol. 81.

15. *Lyanova A.I. et al.* Animal Movement Pattern Model Identification based on Detrended Fluctuation Analysis // 2022 IEEE International Multi-Conference on Engineering, Computer and Information Sciences, SIBIRCON 2022.

16. *Bogachev M. et al.* Video-based marker-free tracking and multi-scale analysis of mouse locomotor activity and behavioral aspects in an open field arena: A perspective approach to the quantification of complex gait disturbances associated with Alzheimer's disease // Front Neuroinform. 2023. Vol. 17.

17. *Gonzalez J.J. et al.* Robust tracking and segmentation of human motion in an image sequence // ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing – Proceedings. Institute of Electrical and Electronics Engineers Inc., 2003. Vol. 3. P. 29–32.

18. *Shi J., Tomasi C.* Good features to track // Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. Publ. by IEEE, 1994. P. 593–600.

19. *Lucas B.D., Kanade T.* An Iterative Image Registration Technique with an Application to Stereo Vision. 1981. Vol. 2. P. 674–679.

20. *Farnebäck G.* Two-Frame Motion Estimation Based on Polynomial Expansion // Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). Springer, Berlin, Heidelberg, 2003. Vol. 2749. P. 363–370.

21. *Guan S., Li H., Zheng W.S.* Unsupervised learning for optical flow estimation using pyramid convolution LSTM // Proc (IEEE Int Conf Multimed Expo). IEEE Computer Society, 2019. Vol. 2019-July. P. 181–186.

22. *Hurst H.E.* A Suggested Statistical Model of some Time Series which occur in Nature // Nature 1957 180:4584. Nature Publishing Group, 1957. Vol. 180, No. 4584. P. 494–494.

23. *Kasdin N.J.* Discrete Simulation of Colored Noise and Stochastic Processes and 1/fα Power Law Noise Generation // Proceedings of the IEEE. 1995. Vol. 83, No. 5. P. 802–827.

## СВЕДЕНИЯ ОБ АВТОРЕ

***СИНИЦА Александр Михайлович*** – старший научный сотрудник СПбГЭТУ «ЛЭТИ».

***Aleksandr Michailovich SINITCA*** – senior researcher of Saint Petersburg Electrotechnical University "LETI".

email: amsinitca@etu.ru

ORCID: 0000-0001-9869-4909

*Материал поступил в редакцию 5 февраля 2025 года*

УДК 004.432

# ПЕРСПЕКТИВЫ РОСТА ПРОИЗВОДИТЕЛЬНОСТИ ПАРАЛЛЕЛЬНЫХ ВЫЧИСЛЕНИЙ С ПОМОЩЬЮ ТЕХНОЛОГИИ СУБИНТЕРПРЕТАТОРОВ В PYTHON

**Р. Д. Синицын**[0009-0003-6750-5222]

*МИРЭА – Российский технологический университет*

sinicinr2003@mail.ru

***Аннотация***

Рассмотрена проблема влияния глобальной блокировки интерпретатора на производительность многопоточных приложений в Python. Описана концепция субинтерпретаторов как одного из решений, позволяющих обходить ограничения GIL и обеспечивать эффективное параллельное выполнение кода. Проведен сравнительный анализ субинтерпретаторов с традиционными методами параллельных вычислений, такими как использование процессов и потоков. Результаты экспериментов показали, что субинтерпретаторы значительно повышают производительность в условиях высоких вычислительных нагрузок. Кроме того, исследованы возможности применения субинтерпретаторов в веб-разработках. Отмечены преимущества использования названного подхода для обработки запросов и управления ресурсами в современных веб-приложениях, что может значительно улучшить их масштабируемость и отклик. Новизна проведенного исследования заключается в глубоком анализе субинтерпретаторов в контексте конкретных сценариев использования, что ранее не получило достаточного освещения в научной литературе. Результаты работы подчеркивают необходимость дальнейшего изучения субинтерпретаторов как альтернативного подхода в Python и интерес к этому разработчиков и исследователей в области высокопроизводительных вычислений.

***Ключевые слова****: Python, CPython, PEP, GIL, субинтерпретатор, многопоточность, многопроцессорность, асинхронность, интерпретатор, параллельные вычисления.*

## ВВЕДЕНИЕ

В эпоху стремительного цифрового прогресса и роста вычислительных мощностей параллельные вычисления становятся неотъемлемой частью современных разработок. Они позволяют эффективно использовать доступные ресурсы, значительно повышая производительность и масштабируемость приложений. С выходом Python 3.12 [1] был представлен экспериментальный API для субинтерпретаторов, который предлагает инновационный подход к управлению параллелизмом, особенно в задачах, интенсивно использующих процессорные ресурсы. Этот шаг является значительным прорывом в контексте ограничений, связанных с глобальной блокировкой интерпретатора (GIL), которая долгое время сдерживала потенциал Python для истинного параллелизма.

Субинтерпретаторы позволяют реализовывать многопоточность без влияния GIL, обеспечивая возможность параллельного выполнения кода в рамках одного процесса. Эта модель открывает новые горизонты для программистов, стремящихся к более эффективному использованию многоядерных систем. В настоящей статье рассмотрены основные принципы работы субинтерпретаторов, их роль в параллельных вычислениях, а также принципиальные отличия от существующих подходов.

Разработка и интеграция эффективных методов обработки параллельных вычислений принципиально необходимы в свете растущих потребностей в высокопроизводительных вычислениях в таких областях, как наука о данных, искусственный интеллект и интернет вещей. Введение нового API для субинтерпретаторов не только открывает новые пути для повышения производительности, но и ставит перед разработчиками и исследователями важные вопросы о будущем параллельных вычислений на Python.

Цель настоящей статьи — дать всестороннее представление о новых возможностях, которые открывает использование субинтерпретаторов, а также оценить их потенциальное влияние на производительность параллельных вычислений и эффективность разработки приложений на Python. Читатели смогут глубже понять перспективы и вызовы, которые стоят перед сообществом Python в эру растущих требований к параллелизму и многопоточности.

## ПРОБЛЕМА ИСПОЛЬЗОВАНИЯ GIL

Глобальная блокировка интерпретатора (GIL) — это механизм, используемый в CPython для синхронизации потоков и позволяющий выполнять только один поток в любой момент времени [2]. Эта блокировка связана с тем, что управление памятью в Python не является потокобезопасным. Однако GIL создает проблемы, особенно в многопоточных программах, ориентированных на вычисления. Например, при запуске четырех потоков Python на четырехъядерном процессоре одновременно будет работать только один поток. Это приводит к тому, что многопоточные программы, требующие высокой процессорной загрузки, будут работать медленнее, чем однопоточные. Хотя процессор имеет несколько ядер, Python может использовать только одно из них, что делает выполнение вычислений менее эффективным [3].

Дополнительная нагрузка на систему от переключений контекста также негативно сказывается на производительности многопоточных программ. В последних версиях Python стандартный интервал для переключения потоков составляет 5 миллисекунд. Уменьшив этот интервал, мы можем наблюдать заметное замедление работы многопоточных приложений. В итоге использование потоков в Python будет менее эффективным для задач с высокой вычислительной нагрузкой по сравнению с задачами, связанными с вводом-выводом. Убедиться в этом можно, например, создав список и рассчитав сумму всех чисел в этом списке (см. рис. 1):

```
import numpy


2 usages
def sum_range(range):
    res = 0
    for el in range:
        res += el
    return res


range = numpy.random.randint(0, 100, 100_000)
sum_range(range)
```

Рис. 1. Код расчёта суммы чисел последовательности, выполняемого в однопоточном режиме

Предполагается, что разделение вычислительной нагрузки на фрагменты и распределения ее по потокам приведёт к уменьшению времени исполнения программы.

```
from threading import Thread

import numpy

ranges = numpy.random.randint(0, 100, 100_000)
threads = []

for range in numpy.split(ranges, 100):
    thread = Thread(target=sum_range, args=(range,))
    thread.start()
    threads.append(thread)

for thread in threads:
    thread.join()
```

Рис. 2. Код расчёта суммы чисел последовательности, выполняемого в многопоточном режиме

Однако во втором примере расчёт выполняется в два раза медленнее, чем в первом. Это связано с тем, что все потоки привязаны к одному и тому же GIL, и в любой момент времени будет выполняться только один поток. Дополнительное время возникает из-за создания потоков и их переключений между собой, исходя из чего такой подход даёт малый прирост в производительности вычислений.

Запуск нескольких интерпретаторов в рамках одного процесса существовал с версии Python 1.5, но в таком подходе возникало достаточное большое количество проблем, связанных с разделением значительной части глобального состояния между интерпретаторами. PEP684 [4] окончательно изменил это, прекратив совместное использование GIL между интерпретаторами – теперь каждый интерпретатор в процессе Python имеет свой собственный GIL и, следовательно, может работать параллельно. Основной причиной того, что работа над GIL для каждого интерпретатора занимала много времени, было то, что CPython внутренне был основан на GIL как на источнике потокобезопасности. Из-за этого возникло много расширений, написанных на C и имеющих глобальное общее состояние.

В Python 3.12 и 3.13 расширения стандартной библиотеки Python, написанные на C, тестируются, и любое глобальное общее состояние перемещается в новый API, который помещает это состояние либо в модуль, либо в состояние интерпретатора. Даже когда эта работа будет окончательно завершена, сторонние расширения, написанные на C, придётся тестировать в парадигме субинтерпретаторов.

**ЧТО ТАКОЕ СУБИНТЕРПРЕТАТОР**

Как известно, системная архитектура языка программирования Python состоит из трёх основных частей [5]:

- Процесс Python, содержащий один или несколько интерпретаторов;
- Интерпретатор, содержащий GIL и один или несколько потоков Python;
- Поток, содержащий информацию о текущем выполняемом коде.

Субинтерпретатор — это интерпретатор, который запускает код Python параллельно с основным интерпретатором, в том же процессе и в том же адресном пространстве, но с полностью изолированными ресурсами выполнения. Это способ добиться изоляции памяти и разделить выполнение кода в зависимости от функциональности или предметной области.

Субинтерпретаторы в Python могут быть полезны в нескольких случаях использования. Их можно использовать для параллельного выполнения кода на отдельных ядрах процессора, что может позволить повысить производительность программ, связанных с нагрузкой на процессор. Субинтерпретаторы также можно использовать для изоляции различных частей программы для повышения безопасности и снижения риска конфликтов, поскольку каждый субинтерпретатор работает в своей отдельной изолированной среде со своими собственной памятью и состоянием. Это может быть полезно в ситуациях, когда необходимо запустить ненадежный код или есть потребность в изоляции определенных частей программы. Кроме того, субинтерпретаторы можно использовать для лучшего параллелизма в программах, связанных с вводом-выводом.

С момента появления Python 1.5 разработчики внедрили C-API, позволяющее использовать несколько интерпретаторов. Однако это решение сталкивалось с серьёзными ограничениями из-за наличия GIL, что препятствовало настоящему

параллелизму и вызывало сложности. Поэтому наиболее распространённым методом для параллельного выполнения кода был встроенный модуль multiprocessing [6].

Разработчики проводили исследование различных подходов для эффективной работы с многоядерным Python, но у каждого из предложенных решений были свои недостатки, и ни одно из них не оказалось простым:

- Существующая практика реализации GIL в модулях расширения никак не помогала с кодом, написанным на Python;
- Другие реализации языка Python, такие как Jython и IronPython, являлись не столь популярными решениями, так как CPython доминирует в сообществе;
- Удаление GIL являлось слишком большим техническим риском на тот момент;
- Проект Трента Нельсона PyParallel являлся неполным и только поддерживающим платформу Windows [7];
- Изменение модуля multiprocessing представляло собой обширный пласт работы, чтобы сделать его достаточно эффективным;
- Другие инструменты параллелизма, такие как dask, ray, MPI, не подходят для среды выполнения.

В ходе проведенных исследований в 2014 году стало достаточно ясно, что решение с использованием изолированных интерпретаторов не несёт в себе высокого уровня технического риска и что большую часть работы в любом случае стоит проделать [8].

В 2017 году разработчики ядра CPython представили инициативу по изменению структуры интерпретаторов с целью улучшения их изоляции от процесса владельца Python и возможности параллельной работы. Эта масштабная задача оказалась сложной и до сих пор остаётся до конца незавершённой. Она была разделена на два PEP: PEP684 [4], который предлагает замену глобальной блокировки интерпретатора (GIL) для каждого из интерпретаторов, и PEP554 [9], который вводит API для создания субинтерпретаторов и обмена данными между ними.

В 2023 году было одобрено предложение PEP703 [10] сделать GIL необязательным в CPython, которое работает совместно с GIL для каждого интерпретатора. Для этого имелись такие предпосылки, как «бессмертные» объекты и обновление расширений, написанных на C, чтобы не использовать общее глобальное состояние [11].

Ещё одним важным моментом является то, что, хотя PEP703 [10] был принят, это было сделано с оговоркой, что это необязательный флаг, и если он окажется слишком проблематичным или сложным, то изменения будут отменены в будущем. С другой стороны, GIL для каждого интерпретатора в значительной степени уже завершён и не требует альтернативного флага во время компиляции в CPython.

## СРАВНЕНИЕ СУБИНТЕРПРЕТАТОРОВ С ДРУГИМИ МЕТОДАМИ ПАРАЛЛЕЛЬНЫХ ВЫЧИСЛЕНИЙ

Python имеет несколько вариантов параллельного выполнения, которые могут быть выбраны в зависимости от задачи:

- Потоки – создаются быстро и позволяют совместно использовать объекты с небольшими накладными расходами. Однако их основной недостаток заключается в том, что они подвержены глобальной блокировке интерпретатора (GIL). Это означает, что при высокой нагрузке на процессор прирост производительности будет минимальным. Тем не менее потоки отлично подходят для фоновых задач, таких как прослушивание сообщений в очереди;
- Сопрограммы – создаются очень быстро и позволяют совместно использовать любые объекты с минимальными накладными расходами. Сопрограммы отлично подходят для операций ввода-вывода [12];
- Многопроцессорность — этот подход создаёт отдельные процессы, которые работают параллельно и не зависят друг от друга. Создание процессов происходит медленно, поэтому для достижения заметного прироста производительности нагрузка должна быть достаточно высокой. В отличие от потоков, каждый процесс имеет свой собственный GIL, что позволяет реализовать полноценное параллельное выполнение;
- Субинтерпретаторы – новая концепция, обладающая параллелизмом многопроцессорной обработки, но с гораздо более быстрым запуском.

Для лучшего понимания итоговые результаты сравнительного анализа моделей параллельного выполнения обработки сведены в таблицу 1.

Таблица 1. Анализ моделей параллельности

| Модель | Время запуска | Исполнение | Обмен данными |
|---|---|---|---|
| Сопрограммы | Самое малое | Условно параллельное | Любой |
| Потоки | Малое | Условно параллельное | Любой |
| Процессы | Долгое | Параллельное | Сериализация |
| Субинтерпретаторы | Среднее | Параллельное | Сериализация или общая память |

Для сравнения производительности субинтерпретаторов в реальном выражении проведен тест скорости создания потоков, процессов и субинтерпретаторов. Результаты скорости создания 100 сущностей представлены на рис. 4.

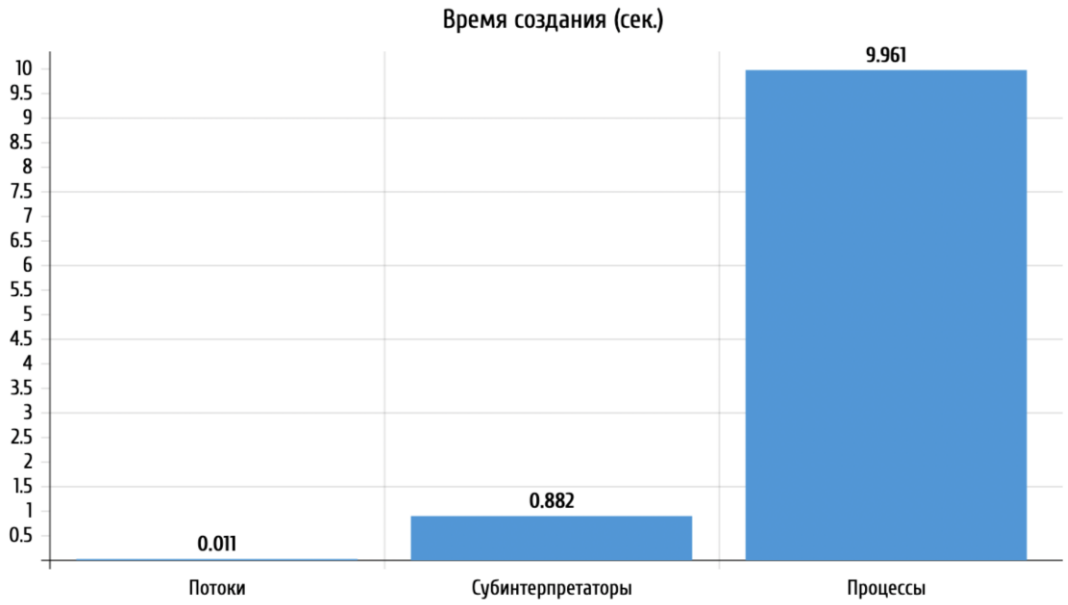



Рис. 4. График скорости запуска 100 сущностей в различных моделях параллельных вычислений

Данный тест показал, что многопоточность запускается примерно в 100 раз быстрее, чем субинтерпретаторы, которые примерно в 10 раз быстрее, чем многопроцессорная обработка. На приведенном рисунке явно видно, что различие в скорости создания интерпретаторов и потоков намного меньше, чем для процессов.

Далее проведено тестирование скорости выполнения параллельных вычислений с интенсивным использованием процессора для расчёта числа $\pi$ с точностью до 2000 десятичных знаков. Результаты эффективности расчётов показаны на рис. 5.

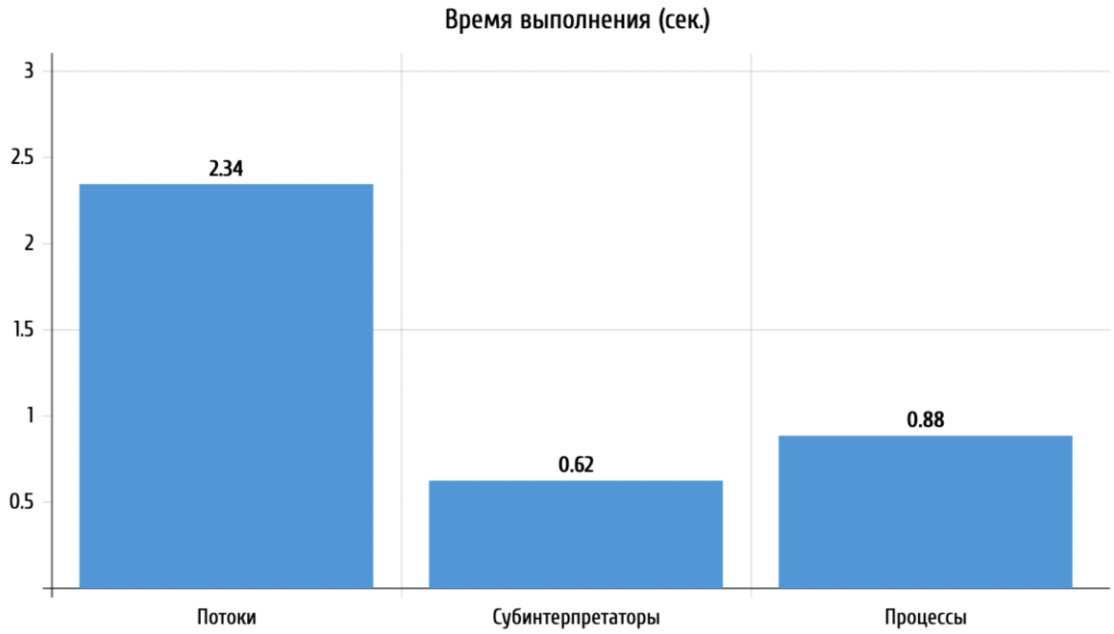



Рис. 5. Скорости вычисления числа $\pi$ с точностью до 2000 десятичных знаков в разных моделях параллельных вычислений

Из рис. 5 видно, что в данном случае наиболее быстрым способом вычисления оказались субинтерпретаторы. Однако результаты первого теста показали, субинтерпретаторы также запускаются в 100 раз медленнее, чем потоки. Таким образом, если задача небольшая, например, вычисление числа $\pi$ до 200 цифр, то преимущества накладных расходов при запуске перевешивают параллелизм, и обработка потоков выполняется быстрее, что видно на рис. 6 (скорости вычислении числа $\pi$ до 200 знаков).

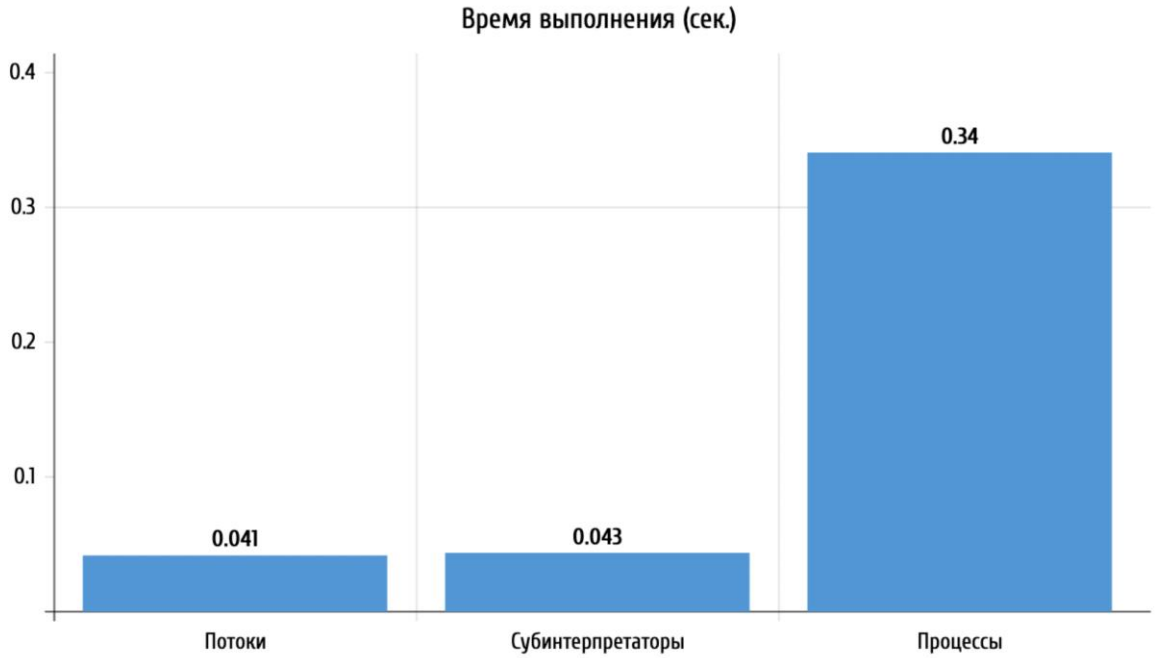



Рис. 6. График скорости вычисления числа $\pi$ с точностью до 200 десятичных знаков в разных моделях параллельных вычислений

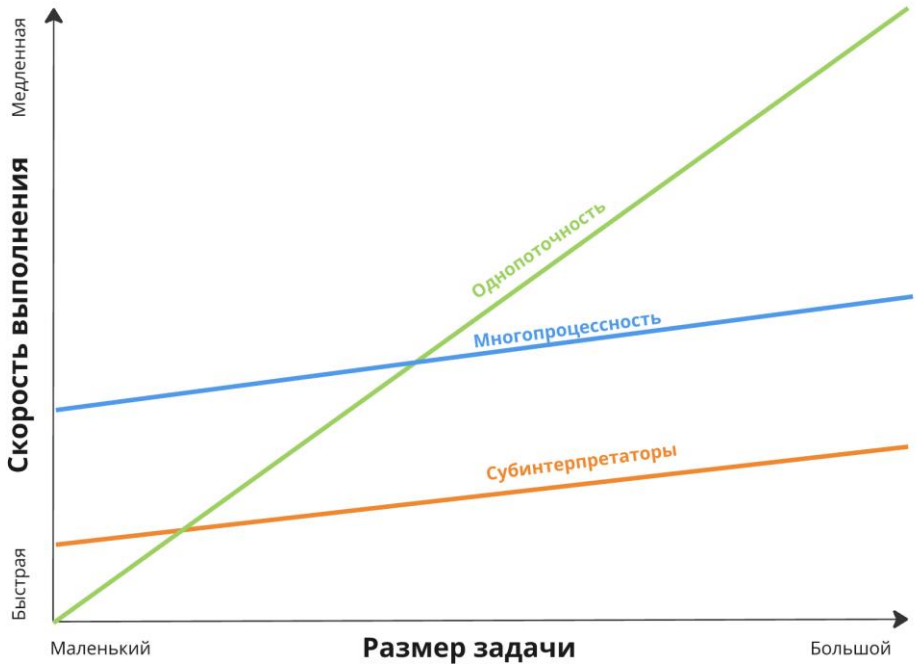



Рис. 7. График скорости выполнения от размера вычисляемых задач

Чтобы наглядно продемонстрировать концепцию границы, при которой параллелизм начинает демонстрировать более высокую эффективность, на рис. 7 представлены размер рабочей нагрузки и скорость увеличения времени выполнения. Эта граница является подвижным значением, зависящим от множества различных факторов.

Важным аспектом многопроцессорности является то, что она чаще всего применяется в моделях, где процессы выполняются длительное время и обрабатывают множество задач, а не создаются и уничтожаются для выполнения одной конкретной задачи. Прекрасным примером этого является популярный веб-сервер Gunicorn [13], который создает «работников», использующих многопроцессность, и эти рабочие процессы остаются активными на протяжении всего времени существования основного процесса. В таких условиях время запуска нового процесса или субинтерпретатора теряет свою значимость, особенно когда веб-воркер может функционировать продолжительное время. Оптимальным способом использования этих параллельных исполнителей для небольших задач является оставление их активными и использование основного процесса для координации и распределения нагрузки.

Многопроцессорность и субинтерпретаторы имеют собственное состояние импорта, что кардинально отличается от потоков и сопрограмм. При использовании асинхронного или многопоточного подходов не нужно беспокоиться об импорте необходимых модулей. Например, можно импортировать что-то в свой модуль и ссылаться на него изнутри функции потока.

Запуск интерпретатора занимает значительное время на выполнение модуля импорта site, который представлен файлом в установке Python. У каждого интерпретатора есть свои кэши и встроенные функции, которые делают его мини-процессом Python. Запуск потока или сопрограммы происходит очень быстро, поскольку они разделяют состояние с основным интерпретатором, что позволяет им избежать лишних вычислений. Однако при этом они подвержены блокировке и не могут работать параллельно.

Кроме того, важным моментом при использовании модели параллельного выполнения, такой как многопроцессорная обработка или субинтерпретаторы, является то, как происходит обмен данными. Независимо от того, используются

ли субинтерпретаторы или многопроцессорная обработка, нельзя просто отправлять существующие объекты Python в рабочие процессы.

Многопроцессорность использует по умолчанию pickle, когда происходят запуск процесса или использование пула процессов, и можно использовать каналы, очереди и общую память в качестве механизмов для отправки данных между рабочими процессами и основным процессом. Эти механизмы основаны на сериализации.

Pickle [14] – это встроенная библиотека сериализации, позволяющая преобразовать большинство объектов Python в байтовые строки и обратно. Она очень гибкая и поддерживает сериализацию различных типов объектов и даже позволяет определять методы сериализации для конкретных объектов. Кроме того, pickle справляется с вложенными объектами и их свойствами. Однако эта гибкость приводит к снижению производительности и может стать узким местом, особенно в моделях, основанных на частом обмене сложными обработанными данными между работниками, так как процесс сериализации – это достаточно медленная операция.

Субинтерпретаторы тоже могут принимать сериализованные данные, но у них также есть второй механизм, называемый общими данными. Общие данные – это высокоскоростное общее пространство памяти, в которое интерпретаторы могут записывать данные и обмениваться ими с другими интерпретаторами. Оно поддерживает только неизменяемые типы:

- Строки;
- Байтовые строки;
- Целые числа и числа с плавающей запятой;
- Булево значение и None;
- Кортежи.

Чтобы поделиться данными с интерпретатором, необходимо установить их как данные инициализации, использовав shared-аргумент, и предоставить словарь с общими значениями (int, float, bool, bytes, str, None, tuple). Если интерпретатор запущен, то можно обмениваться данными, используя канал. Модуль каналов также является частью PEP554 [9] и доступен с помощью секретного импорта.

## ПРИМЕНЕНИЕ СУБИНТЕРПРЕТАТОРОВ В ВЕБ-РАЗРАБОТКЕ КАК ЗАМЕНА КАНОНИЧЕСКИМ МЕТОДАМ

Веб-серверы Python, расположенные перед фреймворками, такими как Django, Flask [15] или FastAPI, используют интерфейс WSGI или ASGI для синхронной и асинхронной обработки запросов соответственно [16]. Веб-серверы прослушивают HTTP-порт, на который приходят запросы, а затем распределяют запросы между набором предварительно проинициализированных процессов. В случае инициализации одного рабочего процесса пользователь, сделав HTTP-запрос, блокирует выполнение других запросов, пока не закончит обработку первого запроса.

Это означает, что при использовании многопроцессорной обработки существуют один GIL для каждого ядра процессора и пул потоков для конкурентной обработки входящих запросов. У такого подхода есть некоторые недостатки. Потоки не параллельны, поэтому, если существует два потока внутри одного рабочего процесса, которые очень заняты, Python не сможет «переместить» или запланировать решение этой задачи на другом ядре процессора.

В ходе настоящего исследования стояла цель разработать подход с субинтерпретаторами в качестве механизма работы воркеров, чтобы заменить подход с многопроцессорностью. Преимуществом нового подхода должно стать использование высокопроизводительного API каналов общей памяти для взаимодействия между работниками, вследствие чего воркеры будут меньше весить, занимая меньше памяти на хосте и оставляя больше памяти и ресурсов для обработки запросов.

Второй целью было скомпилировать основную ветку CPython 3.13 с реализацией потоков без GIL PEP703 [10], чтобы изучить возможность запуска потоков без GIL внутри этой модели.

Для достижения поставленных целей вручную был скомпилирован CPython 3.13a04 из исходного кода, а также разработан веб-воркер Hypercorn, работающий с помощью концепции субинтерпретаторов. В ходе работы также были решены следующие проблемы:

- Создание субинтерпретатора;

- Создание сигнального канала для передачи команд о запросах на выключение;
- Реализация подкласса threading.Thread и собственного метода .stop(), который отправляет сигнал субинтерпретатору;
- Запуск каждого дополнительного интерпретатора в потоке;
- Преобразование списка сокетов в кортеж кортежей.

Для сравнения производительности потоков, процессов и субинтерпретаторов было реализовано веб-приложение, написанное на Flask и вычисляющее факториал от 30. В качестве фреймворка для реализации веб-приложения был выбран Flask, так как, например, запустить приложение, написанное на Django, не получилось из-за тесной интеграции этого фреймворка с библиотекой для работы со временем datetime, которое использует расширение, написанное на C и использующее общее глобальное состояние.

Нагрузочное тестирование производилось с помощью инструмента Locust [17] с настройкой конкурентности, равной 100 пользователям, и скоростью появления 5 пользователей в секунду.

Результаты тестирования количества обрабатываемых запросов в секунду, потребление оперативной памяти и нагрузки процессора при использовании потоков, процессов и субинтерпретаторов представлены на рис. 8–13, для сравнения производительности были использованы следующие параметры запуска:

- Потоки – были запущены в 4 потока и 1 процесс;
- Процессы – был произведен запуск 4 рабочих процесса в однопоточном режиме;
- Субинтерпретаторы – запуск производился при использовании 4 субинтерпретаторов.

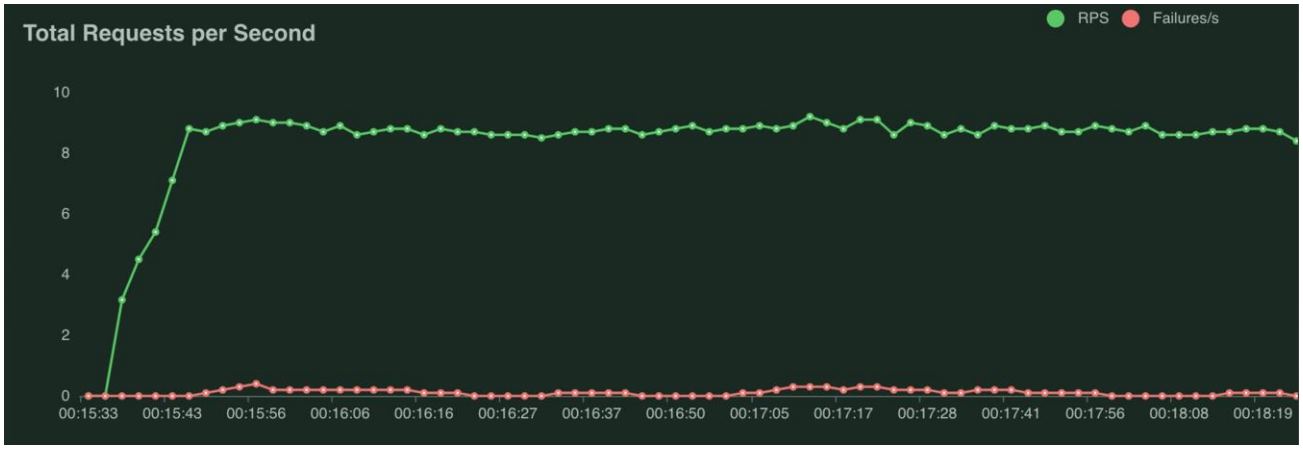



Рис. 8. График обрабатываемых запросов в секунду при многопоточной модели

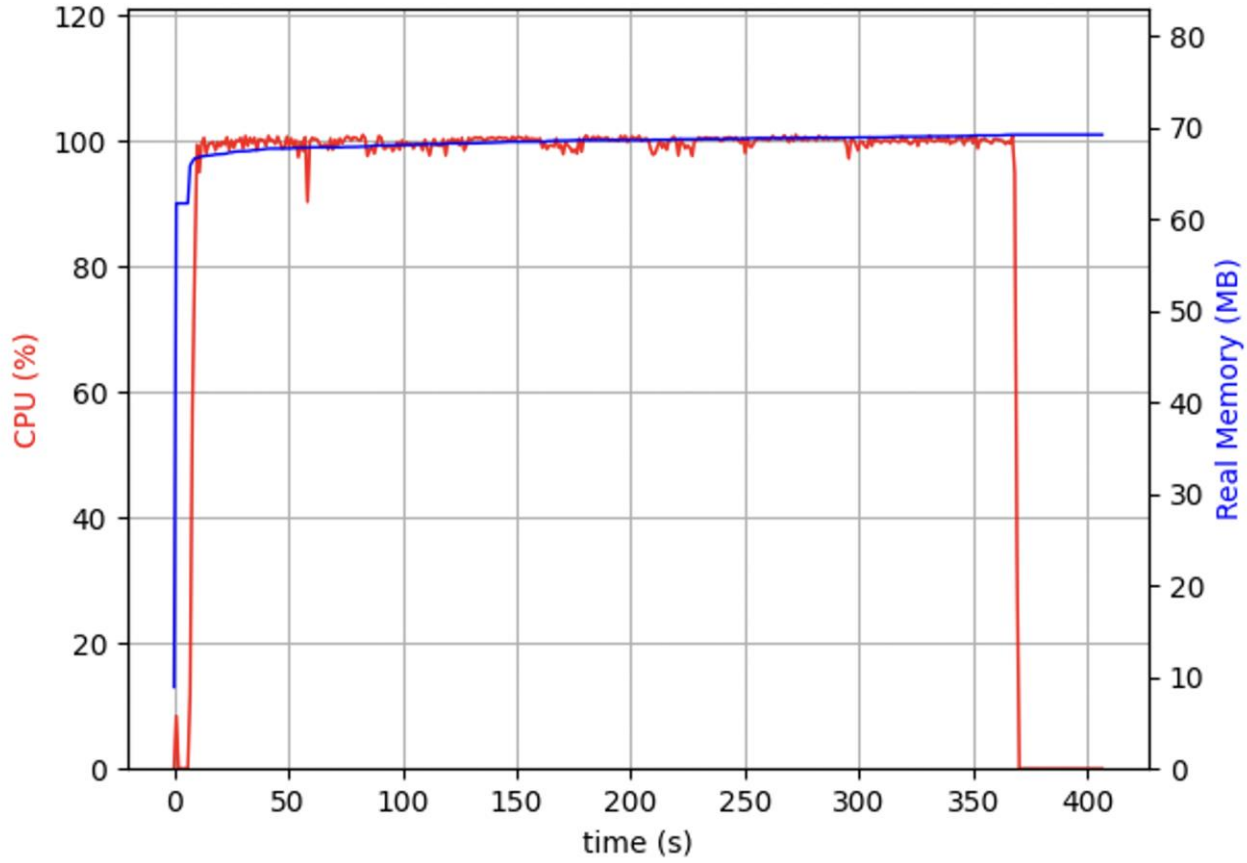



Рис. 9. График потребления памяти и ресурсов процессора при многопоточной модели

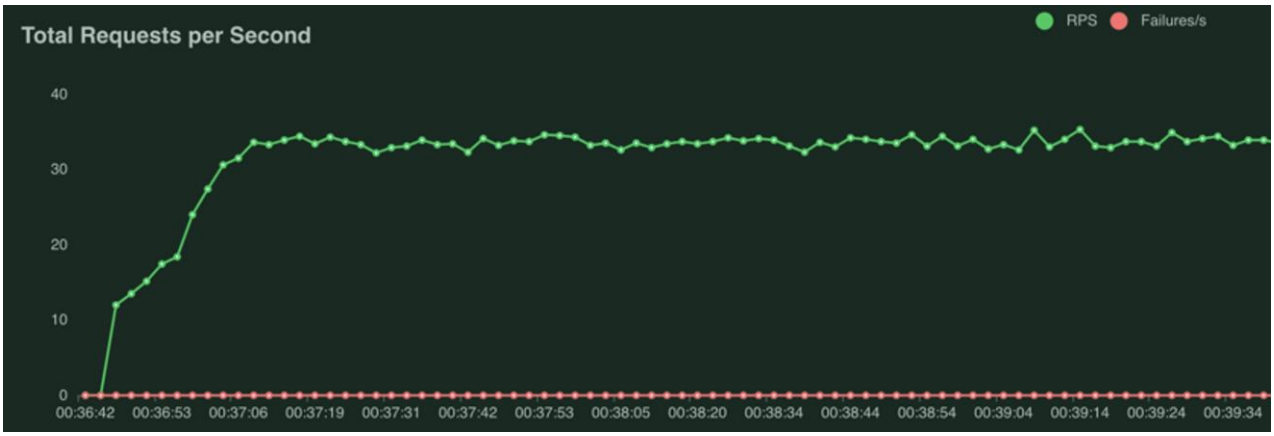



Рис. 10. График обрабатываемых запросов в секунду при многопроцессорной модели

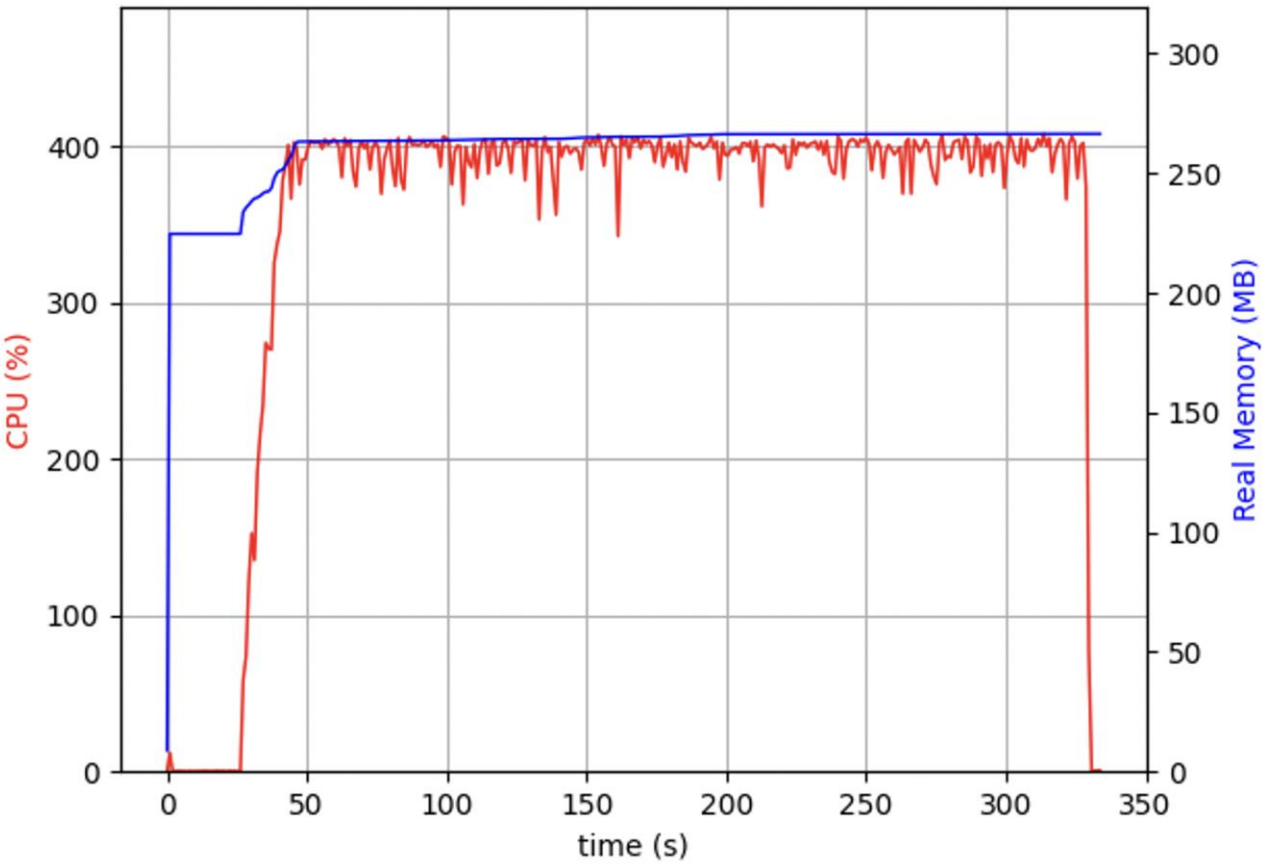



Рис. 11. График потребления памяти и ресурсов процессора при многопроцессорной модели

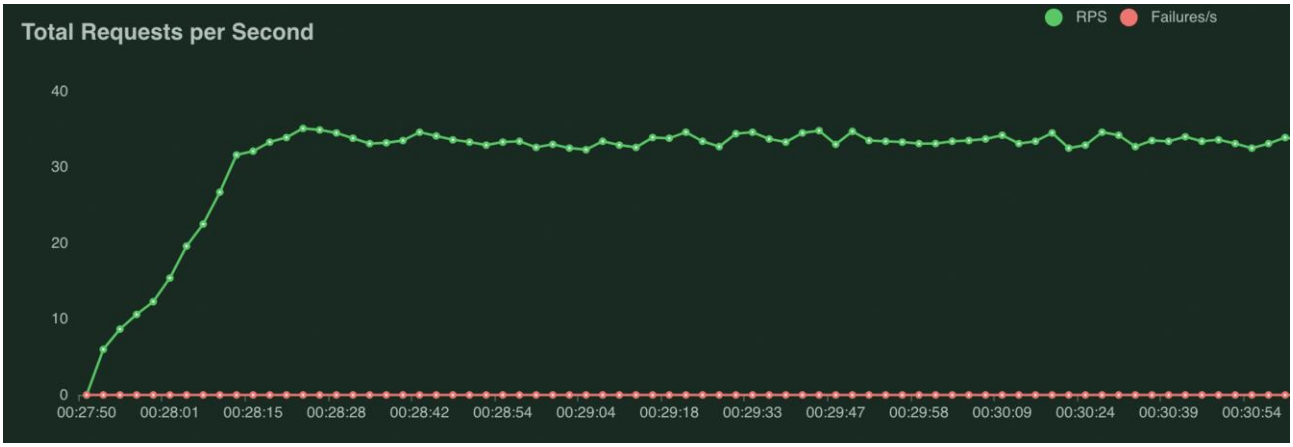



Рис. 12. График обрабатываемых запросов в секунду при использовании субинтерпретаторов

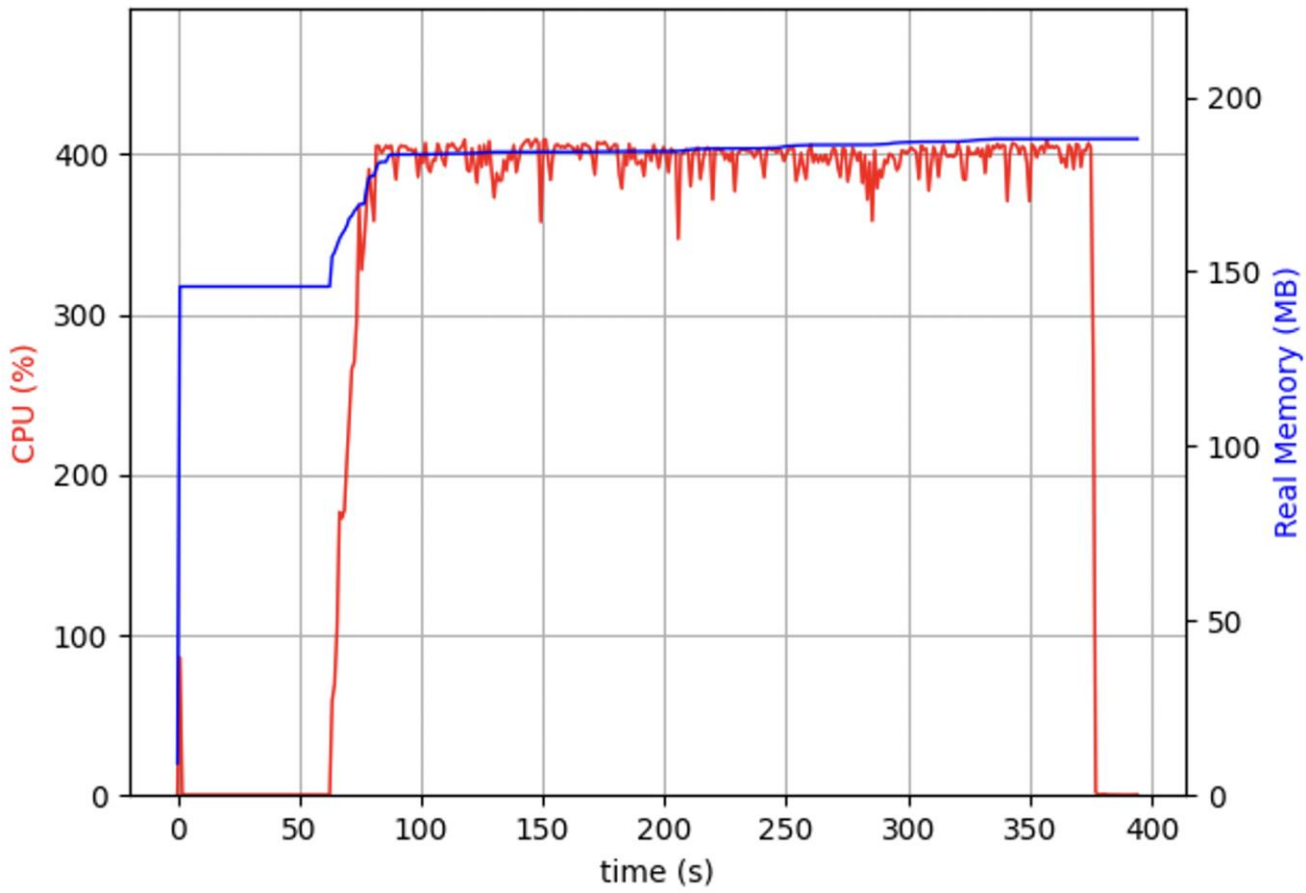



Рис. 13. График потребления памяти и ресурсов процессора при использовании субинтерпретаторов

Из полученных результатов видно, что при использовании многопоточности производительность приложения достаточно низкая и не превышает 10 запросов в секунду. При использовании субинтерпретаторов достигается производительность, не превышающая 40 запросов в секунду. Однако такой подход является самым экономным по памяти и занимает приблизительно 70 МБ оперативной памяти. Субинтерпретаторы, в свою очередь, являются средним вариантом между многопоточностью и процессами, сохраняя неменьшую производительность и потребляя 180 МБ оперативной памяти, в то время как многопроцессорность занимает 270 МБ оперативной памяти. Это доказывает преимущества и дальнейший потенциал использования предложенного подхода для веб-приложений.

**ЗАКЛЮЧЕНИЕ**

Рассмотрена проблема ограничений, накладываемых GIL на производительность многопоточных приложений в Python. Проанализирована концепция субинтерпретаторов как эффективного решения, способствующего повышению параллельной обработки задач. Результаты проведённых экспериментов показали, что субинтерпретаторы могут существенно улучшить производительность вычислений, особенно в сценариях с высокими нагрузками.

Кроме того, было проведено сравнение субинтерпретаторов с подходами многопроцессности и многопоточности. Результаты показали, что, в отличие от многопоточности, субинтерпретаторы позволяют избежать проблем, вызванных GIL, обеспечивая лучшую масштабируемость при выполнении CPU-bound задач. В то же время многопроцессный подход предоставляет преимущества в использовании многоядерных систем, но требует большего объёма ресурсов из-за необходимости создавать отдельные процессы и обмениваться данными между ними.

Сравнительный анализ показал, что субинтерпретаторы, будучи более лёгковесными, обеспечивают баланс между производительностью и эффективностью использования ресурсов. Особенно это заметно в ситуациях, где число задач превышает количество доступных ядер процессора. Таким образом, использование субинтерпретаторов обеспечивает синергетическую комбинацию лучших характеристик многопоточности и многопроцессности.

Выявлен значительный потенциал субинтерпретаторов для применения в различных областях, включая веб-разработку и обработку больших объёмов данных, что имеет важное значение в контексте современных вычислительных задач. Важно отметить, что, несмотря на частичную реализацию данной концепции, субинтерпретаторы представляют собой инновационный шаг, и существует необходимость в дальнейших исследованиях и разработках, направленных на полноценную интеграцию субинтерпретаторов в экосистему Python.

**Благодарности**

Автор выражает признательность преподавателям кафедры инструментального и прикладного программного обеспечения РТУ МИРЭА за ценные консультации по аспектам рассматриваемой предметной области.

**СПИСОК ЛИТЕРАТУРЫ**

1. Официальная документация Python. URL: https://www.python.org/
2. *Фаулер М.* Asyncio и конкурентное программирование на Python. М.: ДМК Пресс, 2023. 398 с.
3. *Beazley D.* Understanding the python GIL // PyCON Python Conference. Atlanta, Georgia, 2010.
4. PEP 684 – A Per-Interpreter GIL. URL: https://peps.python.org/pep-0684/
5. *Shaw A.* CPython Internals: Your Guide to the Python 3 Interpreter. 2021. 394 p.
6. *Brownlee J.* Python Multiprocessing Jump-Start: Develop Parallel Programs, Side-Step the GIL, and Use All CPU Cores (Python Concurrency Jump-Start Series). 2022. 144 p.
7. Официальный сайт PyParallel. URL: https://pyparallel.org/
8. *Roghult A.* Benchmarking Python Interpreters: Measuring Performance of CPython, Cpython, Jython and PyPy: Degree project in computer science and engineering. Sweden, 2016.
9. PEP 554 – Multiple Interpreters in the Stdlib.
URL: https://peps.python.org/pep-0554/
10. PEP 703 – Making the Global Interpreter Lock Optional in CPython.
URL: https://peps.python.org/pep-0703/

11. PEP 683 – Immortal Objects, Using a Fixed Refcount.
URL: https://peps.python.org/pep-0683/

12. *Савостин П.А., Ефремова Н.Э.* Практическое применение асинхронного программирования на языке Python при помощи пакета asyncio // Программные системы и вычислительные методы. 2018. №2. С. 11–16.

13. Gunicorn - Python WSGI HTTP Server for UNIX. URL: https://gunicorn.org/

14. Pickle — Python object serialization.
URL: https://docs.python.org/3/library/pickle.html

15. *Grinberg M.* Flask Web Development: Developing Web Applications with Python. O'Reilly Media, 2014. 258 p.

16. *Gardner J.* The Web Server Gateway Interface (WSGI) / In: The Definitive Guide to Pylons. Apress, 2009. 513 p.

17. Locust – A modern load testing framework. URL: https://locust.io/

## PROSPECTS FOR IMPROVING THE PERFORMANCE OF PARALLEL COMPUTING USING PYTHON SUBINTERPRETER TECHNOLOGY

**R. D. Sinitsyn**[0009-0003-6750-5222]

*MIREA – Russian Technological University*

sinicinr2003@mail.ru

***Abstract***

The problem of the impact of global interpreter locking on the performance of multithreaded applications in Python is considered. The concept of subinterpreters is described as one of the solutions that circumvent the limitations of the GIL and ensure efficient parallel code execution. A comparative analysis of subinterpreters with traditional methods of parallel computing, such as the use of processes and threads, is carried out. The experimental results have shown that subinterpreters significantly increase performance in conditions of high computing loads. In addition, the possibilities of using subinterpreters in web development are explored. The advantages of using

this approach for query processing and resource management in modern web applications are indicated, which can significantly improve their scalability and responsiveness. The novelty of the research lies in the in-depth analysis of subinterpreters in the context of specific use cases, which had not previously received sufficient coverage in the scientific literature. The results of the work emphasize the need for further study of subinterpreters as an alternative approach in Python and the interest of developers and researchers in the field of high-performance computing in this.

***Keywords****: Python, CPython, PEP, GIL, subinterpreter, multithreading, multiprocessing, asynchrony, interpreter, parallel computing.*

**REFERENCES**

1. Official Python Documentation. URL: https://www.python.org/
2. *Fowler M.* Asyncio i konkurentnoe programmirovanie na Python. M.: DMK Press, 2023. 398 p.
3. *Beazley D.* Understanding the python GIL // PyCON Python Conference. Atlanta, Georgia, 2010.
4. PEP 684 – A Per-Interpreter GIL. URL: https://peps.python.org/pep-0684/
5. *Shaw A.* CPython Internals: Your Guide to the Python 3 Interpreter. 2021. 394 p.
6. *Brownlee J.* Python Multiprocessing Jump-Start: Develop Parallel Programs, Side-Step the GIL, and Use All CPU Cores (Python Concurrency Jump-Start Series). 2022. 144 p.
7. PyParallel's official website. URL: https://pyparallel.org/
8. *Roghult A.* Benchmarking Python Interpreters: Measuring Performance of CPython, Cpython, Jython and PyPy: Degree project in computer science and engineering. Sweden, 2016.
9. PEP 554 – Multiple Interpreters in the Stdlib.
URL: https://peps.python.org/pep-0554/
10. PEP 703 – Making the Global Interpreter Lock Optional in CPython.
URL: https://peps.python.org/pep-0703/
11. PEP 683 – Immortal Objects, Using a Fixed Refcount.
URL: https://peps.python.org/pep-0683/

12. *Savostin P.A., Efremova N.E.* Prakticheskoe primenenie asinkhronnogo programmirovaniya na yazyke Python pri pomoshchi paketa asyncio // Programmnye sistemy i vychislitel'nye metody. 2018. №2. S. 11–16.

13. Gunicorn - Python WSGI HTTP Server for UNIX. URL: https://gunicorn.org/

14. Pickle — Python object serialization.
URL: https://docs.python.org/3/library/pickle.html

15. *Grinberg M.* Flask Web Development: Developing Web Applications with Python. O'Reilly Media, 2014. 258 p.

16. Gardner J. The Web Server Gateway Interface (WSGI) / In: The Definitive Guide to Pylons. Apress, 2009. 513 p.

17. Locust – A modern load testing framework. URL: https://locust.io/

## СВЕДЕНИЯ ОБ АВТОРЕ

***СИНИЦЫН Роман Дмитриевич*** – Студент Российского технологического университета МИРЭА по направлению «Программная инженерия».

***Roman Dmitrievich SINITSYN*** – Student of the Russian Technological University MIREA with a degree in Software Engineering.

email: sinicinr2003@mail.ru

ORCID: 0009-0003-6750-5222

Материал поступил в редакцию 16 марта 2025 года

УДК 004.451.53

# ОНТОЛОГИЧЕСКОЕ ПРЕДСТАВЛЕНИЕ ВНУТРЕННЕГО ДОКУМЕНТА ПО ОСНОВНОЙ ДЕЯТЕЛЬНОСТИ ОБРАЗОВАТЕЛЬНОГО УЧРЕЖДЕНИЯ

**В. В. Чуйкова[1] [0009-0000-6784-8135], М. О. Таныгин[2] [0000-0002-4099-1414]**

[1, 2]Юго-западный государственный университет, Курская обл., Курск, 305040, Россия

[1]chuikova_v_v@bk.ru, [2]tanygin@yandex.ru

***Аннотация***

Проанализирована структура служебных записок кафедры высшего учебного заведения с целью упрощения ретроспективного поиска среди массива документации и выделения групп однотипных документов. Установленные правила классификации служебных записок на основе формальных признаков позволят значительно упростить процесс поиска и работы с ними, а также создания новых типовых документов.

***Ключевые слова:*** *онтология документа, структура документа, онтологическая модель, анализ структуры документов.*

## ВВЕДЕНИЕ

Нормативные документы имеют определённую логическую структуру. Например, как правило, приказы делятся на логические разделы, научные статьи состоят из аннотации, введения, обзора существующих работ и других секций, а служебные записки обычно содержат краткую информацию о каком-либо процессе или событии. Информация о логической структуре полезна для автоматического анализа документа.

В условиях увеличения объёма внутренней служебной документации необходимо быстро находить подобные документы для создания на их основе новых, а также легко ориентироваться в организационно-распорядительной документации (ОРД).

## ПРОБЛЕМАТИКА

В [1] авторы предложили извлекать логическую структуру из сканированных документов. Этот процесс имеет ряд трудностей: сканированный документ может не содержать текстового слоя, часть документа может быть частично отформатирована и, самое главное, документы имеют разную структуру. В [2] представлено решение задачи оперативного формирования документов, основанное на формальном моделировании информационной структуры и процессов генерации документов. В [3] авторы попытались выделить физическую и логическую структуры PDF-документа для его дальнейшего распознавания и обработки. В [4] рассмотрены структура данных и модель документа в Microsoft Word и обоснована концептуальную модель программного обеспечения для нормоконтроля печатных работ. В [5] предложено выделять из сценариев видеоигр необходимые сущности и передавать их дальнейшим шагам алгоритма, который по текстовым описаниям будет генерировать игровые ресурсы. Такая декомпозиция позволяет оптимизировать процесс создания сценария игры.

Актуальность настоящего исследования заключается в необходимости проведения ретроспективного поиска среди организационно-распорядительной документации. Поисковой запрос обычно формируется простыми разговорными фразами и на основе субъективных понятий, тогда как в документе текст носит официально-деловой характер, с соответствующей терминологией. Соответственно, необходимы правила классификации ОРД и описания их смысла на основе формальных признаков.

## СТРУКТУРА СЛУЖЕБНЫХ ЗАПИСОК В ОРГАНИЗАЦИИ: ЧЕМ ОНИ ОТЛИЧАЮТСЯ

Традиционно служебная записка представляет собой документ, созданный в каком-либо текстовом редакторе. Отличает его от обычных документов то, что в его состав входят только структурированный текст и таблицы, а графика, математические выражения, диаграммы в документах такого рода не допускаются [6, ГОСТ Р. 7.0.97-2016]. В настоящей статье мы рассматриваем сферу образования, а конкретно, документальное обеспечение подготовки специалистов в высших учебных заведениях.

Служебные записки (СЗ) имеют следующую общую структуру:

1. Автор (от кого пишется СЗ).
2. Адресат (кому направляется документ).
3. Заголовок «Служебная записка».
4. Основной текст.
5. Подпись автора и/или визы задействованных сотрудников.
6. Резолюция адресата (опционально), накладываемая после анализа содержимого СЗ.

На рис. 1 представлена типовая СЗ кафедры в рамках учебного процесса.

| | |
|---|---|
| Кафедра | Ректору<br>ФИО |

СЛУЖЕБНАЯ ЗАПИСКА
07.02.2025 г. № 07.07.23 / 53

Об отчислении аспиранта

Прошу отчислить ФИО из состава аспирантов университета в связи с невыполнением обучающимся по образовательной программе высшего образования – программе подготовки научно-педагогических кадров в аспирантуре (ОП ВО) обязанностей по добросовестному освоению ОП ВО и выполнению учебного плана с 14.02.2025 года.

| | |
|---|---|
| Зав. кафедрой | ФИО |
| | |

Рис. 1. Типовая СЗ кафедры в рамках учебного процесса

Пункты 1–6 являются, по сути, атрибутами. Если проводить систематизацию документов только на их основании, то с практической точки зрения в этом не так много смысла. Действительно, так ли важно, что в конкретную дату на имя определённого сотрудника из разных структурных подразделений организации поступило определённое количество СЗ. Обычно взаимосвязи «Автор – Адресат» определяются организационно-штатной структурой конкретного учебного заведения,

а не характером сведений, содержащихся в служебной записке, тогда как основной текст содержит информацию, которая будет содержаться в поисковых запросах, о затрагиваемых событиях, процессах, действиях, которые и определяют назначение каждого конкретного документа. В то же время даже такой достаточно сложно формализуемый элемент документа может содержать типовые лингвистические конструкции, которые как определяют отношение «Автор – Адресат», так и позволяют провести первичную классификацию документа.

Мы предположили, что для формализации типа документа надо рассмотреть виды связок «глагол (действие) – дополнение (тип действия)».

Было проанализировано более 700 служебных записок организации за несколько лет. Выявлены следующие связки: «прошу – зачислить», «прошу – отчислить», «прошу – рассмотреть», «прошу – заверить», «прошу – назначить», «прошу – утвердить». Их можно обобщить в связку вида «Прошу – дополнение».

Кроме того, установлены более редкие связки: «предлагаю – следующее», «предлагаю – состав», «прошу – разрешения», «предлагается», «довожу – до вашего сведения», «прошу – установить (надбавку)», «прошу – возместить».

По результатам проведённого анализа выделены обобщённые связки трёх типов, которые наиболее характерны для служебных записок:

1. «Прошу – дополнение»;
2. «Предлагаю – дополнение»;
3. «Довожу до сведения (информирование)».

## ОНТОЛОГИЧЕСКАЯ МОДЕЛЬ

Для построения онтологической модели [7] проанализируем эти сочетания и выделим основные сущности служебной записки.

Если рассматривать более глубоко содержимое основной части СЗ, то для него, с учётом описанных выше атрибутов (таких как автор, адресат, заголовок «служебная записка», основной текст, подпись автора и/или визы задействованных сотрудников, резолюция адресата) можно выделить следующие сущности.

1. Объект воздействия (на что хотим повлиять) представлен на рис. 2.


Рис. 2. Объект воздействия

2. Действия, которые инициирует служебная записка, показаны на рис. 3.


Рис. 3. Действия

3. «Третьи лица» (кому переадресовано)

- ✓ структурные подразделения;
- ✓ работники.

4. Результат рассмотрения адресатом рис. 4

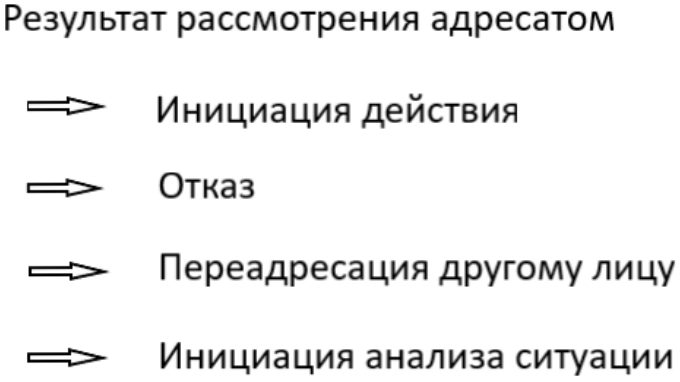



Рис. 4. Результат рассмотрения адресатом

Определим объекты, классы и отношения между ними.

1. Объекты представлены на рис. 5.

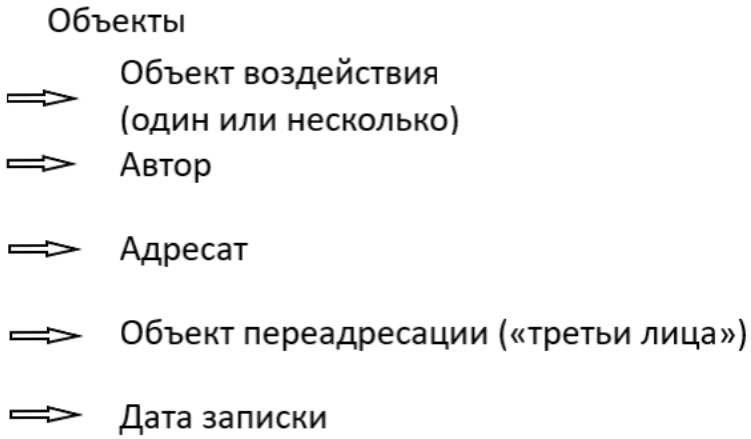



Рис. 5. Объекты

2. Классы показаны на рис. 6.

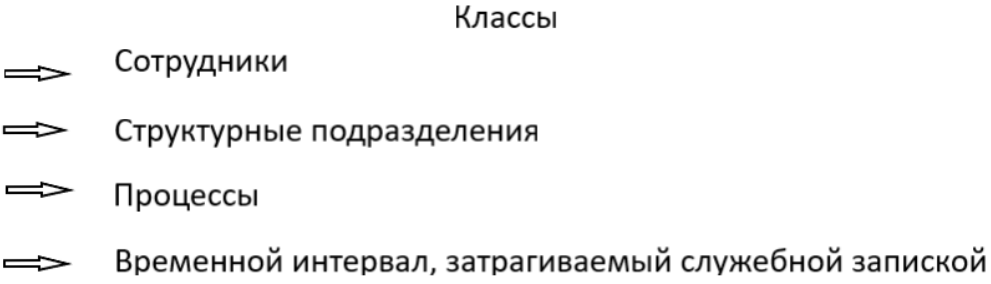



Рис. 6. Классы

Все объекты входят в те или иные классы. Так, «Объект воздействия» (один или несколько) может входить во все три класса: он может быть и сотрудником, и структурным подразделением, и процессом. Объект «Автор» может быть только сотрудником. Объекты «Автор» и «Адресат» также могут быть только сотрудниками. «Объект переадресации» («третьи лица») может быть сотрудником или структурным подразделением. Объект «Дата записки», относящийся к классу «Временной интервал, затрагиваемый служебной запиской», косвенно относится к процессам, характеризуя период времени влияния на «Объект воздействия» и в целом актуальность служебной записки.

3. Отношения между объектами:

- ✓ «Автор – Адресат» (рис. 7).


Рис. 7. Отношение «Автор – Адресат»

- ✓ «Адресат – Объект переадресации»
- ✓ «Адресат – Время»
- ✓ «Автор – Объект переадресации» представлено на рис. 8.


Рис. 8. Отношение «Автор – Объект переадресации»

Для объекта «Дата записки» можно выделить три свойства, определяющие её, этот объект характеризуется следующим.

1. Время «актуальности» записки
   - ✓ Текущее;
   - ✓ Интервал в прошлом;
   - ✓ Интервал в будущем.
2. Для отношения «Дата записки – Объект воздействия» можно выделить 10 характеристик, зависящих от даты инициализации служебной записки и её расположения на оси времени относительно времени, в течение которого документ актуален (рис. 9).

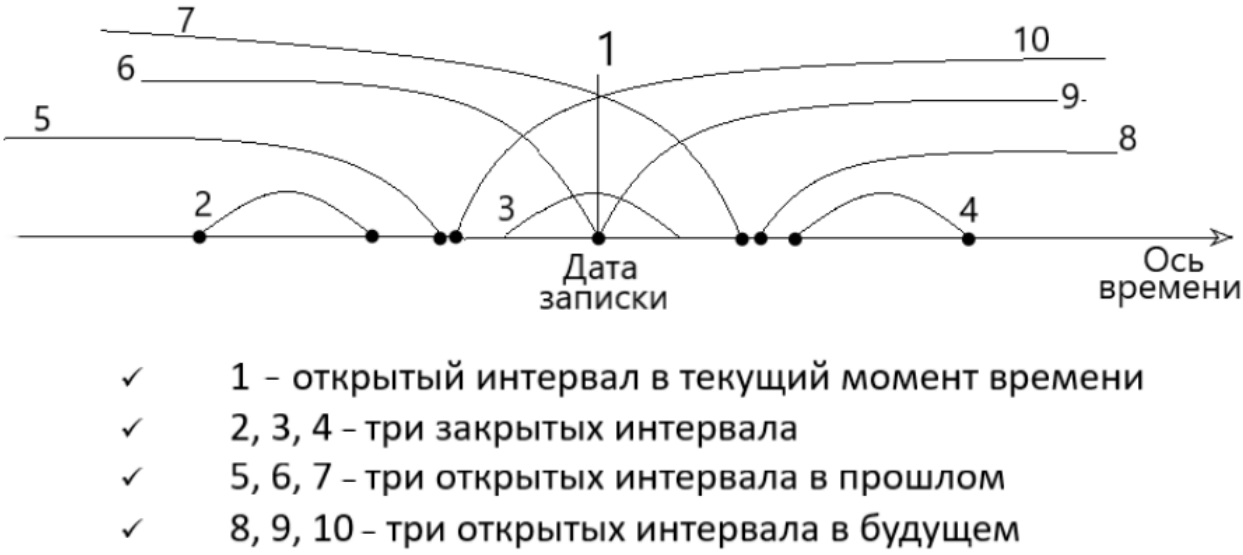



Рис. 9. Характеристики для отношения «Дата записки – Объект воздействия»

Такая подробная декомпозиция документа, выделение объектов и определение отношений между объектами нужны, чтобы получить возможность легко идентифицировать служебные записки и строить логические выводы о том, какая именно служебная записка перед нами, а также быстро провести поиск и найти нужную. Кроме того, был проведён поиск среди имеющегося массива служебных записок и найдены значения количественных показателей СЗ обозначенных видов (табл. 1).

Таблица 1. Количество СЗ разных видов

| Смысловые связки в СЗ | Количество |
|---|---|
| 1. «Прошу – зачислить» | 4% |
| 2. «Прошу – отчислить» | 4% |
| 3. «Прошу – рассмотреть» | 5% |
| 4. «Прошу – заверить» | 6% |
| 5. «Прошу – назначить» | 10% |
| 6. «Прошу – утвердить» | 7% |
| 7. «Прошу – назначить | 4% |
| 8. «Предлагаю – следующее» | 7% |

| | |
|---|---|
| 9. «Предлагаю – состав» | 6% |
| 10. «Прошу – разрешения» | 5% |
| 11. «Предлагается» | 7% |
| 12. «Довожу – до вашего сведения» | 21% |
| 13. «Прошу – установить (надбавку)» | 9% |
| 14. «Прошу – возместить» | 5% |

Можно сделать вывод, о том что наибольшее количество документов типа «служебная записка» встречается в исследуемой сфере деятельности (образование) со связками «Прошу – дополнение» (табл. 2).

Таблица 2 – Количество СЗ обобщённых типов

| Обобщённые связки в СЗ | Количество |
|---|---|
| 1. «Прошу – дополнение» | 59% |
| 2. «Довожу – до сведения» (информирование) | 21% |
| 3. «Предлагаю – дополнение» | 20% |

**ЗАКЛЮЧЕНИЕ**

Использовав представленную онтологическую модель документа, основанную на формальной структуре служебных записок, можно легко организовывать хранение, автоматическую обработку, ретроспективный поиск в архивах и найти документы, имеющие одинаковые связки типа «Прошу – дополнение», «Предлагаю – дополнение», «Довожу до сведения (информирование)» в зависимости от того, что именно требуется найти. Очевидно, что СЗ с какими-либо предложениями мы не будем искать среди СЗ со связками «Прошу – дополнение» и «Довожу до сведения (информирование)». Кроме того, традиционная структура привязки служебной записки к дате её создания не всегда позволяет корректно установить тот временной интервал, который в ней рассмотрен или затрагивается. Предложенное нами описание отношения «Объект воздействия – Дата записки» позволяет добавлять объектам, выделенным в составе служебной записки, свойство «темпоральности», то есть изменчивости свойств объекта во времени. С учётом

того, что под объектами мы понимаем не только работников, но и целые структурные подразделения и процессы, это даст возможность использовать предложенную модель поиска не только в электронных архивах, но и современных CRM-системах.

**СПИСОК ЛИТЕРАТУРЫ**

1. *Богатенкова А.О., Козлов И.С., Беляева О.В., Перминов А.И.* Извлечение логической структуры из сканированных документов // Труды ИСП РАН. 2020. №4. URL: https://cyberleninka.ru/article/n/izvlechenie-logicheskoy-struktury-iz-skanirovannyh-dokumentov (дата обращения: 24.02.2025).

2. *Пенькова Т.Г.* Модели и методы оперативного формирования документов // Вычислительные технологии. 2009. Т. 14. № 2. С. 98–109.

3. *Беляева О.В., Перминов А.И., Козлов И.С.* Использование синтетических данных для тонкой настройки моделей сегментации документов // Труды ИСП РАН. 2020. Т. 32, вып. 4. С. 189–202.
https://doi.org/10.15514/ISPRAS–2020–32(4)–14 (дата обращения: 24.02.2025).

4. *Виноградская М.Ю., Крючок А.Ю.* Описание структуры документов Microsoft Word для разработки программ нормоконтроля // Вестник Калужского университета. 2019. №3 (44). С. 113–116.

5. *Нурлыгаянов Н.Р., Кугуракова В.В.* Подход к созданию корпуса текстов видеоигр на основе универсальной структуры // Электронные библиотеки. 2024. Т. 27. №4. С. 578–597.

6. ГОСТ Р. 7.0.97-2016. Национальный стандарт Российской Федерации. Система стандартов по информации, библиотечному и издательскому делу. Организационно-распорядительная документация. Требования к оформлению документов. URL: https://docs.cntd.ru/document/1200142871.
URL: http://publication.pravo.gov.ru/Document/View/0001202002070036.

7. *Горшков С.* Введение в онтологическое моделирование. Ревизия 2.4. ООО «ТриниДата», 2014–2018, 150 с.

# INTERNAL DOCUMENT ONTOLOGICAL REPRESENTATION IN THE MAIN ACTIVITY OF THE EDUCATIONAL ORGANIZATION

**V. V. Chuikova[1] [0009-0000-6784-8135], M. O. Tanygin[2] [0000-0002-4099-1414]**

[1, 2]*South-West State University*

[1]chuikova_v_v@bk.ru, [2]tanygin@yandex.ru

***Abstract***

The paper analyzes the structure of the department's memos to simplify a retrospective search among an array of documentation and identify groups of similar documents, which will greatly simplify the process of searching and working with them, as well as the creation of new standard documents.

***Keywords****: document ontology, document structure, ontological models, document structure analysis.*

**REFERENCES**

1. *Bogatenkova A.O., Kozlov I.S., Belyaeva O.V., Perminov A.I.* Izvlechenie logicheskoj struktury iz skanirovannyh dokumentov // Trudy ISP RAN. 2020. №4. URL: https://cyberleninka.ru/article/n/izvlechenie-logicheskoy-struktury-iz-skanirovannyh-dokumentov (data obrashcheniya: 24.02.2025).

2. *Pen'kova T.G.* Modeli i metody operativnogo formirovaniya dokumentov // Vychislitel'nye tekhnologii. 2009. T. 14. № 2. S. 98–109.

3. *Belyaeva O.V., Perminov A.I., Kozlov I.S.* Ispol'zovanie sinteticheskih dannyh dlya tonkoj nastrojki modelej segmentacii dokumentov // Trudy ISP RAN. 2020. T. 32, vyp. 4. S. 189–202.
https://doi.org/10.15514/ISPRAS–2020–32(4)–14 (data obrashcheniya: 26.02.2025).

4. *Vinogradskaya M.Yu., Kryuchok A.Yu.* Opisanie struktury dokumentov Microsoft Word dlya razrabotki programm normokontrolya // Vestnik Kaluzhskogo universiteta. 2019. №3 (44). S. 113–116.

5. *Nurlygayanov N.R., Kugurakova V.V.* Podhod k sozdaniyu korpusa tekstov videoigr na osnove universal'noj struktury // Elektronnye biblioteki. 2024. T. 27. №4. S. 578–597.

6. GOST R. 7.0.97-2016. Nacional'nyj standart Rossijskoj Federacii. Sistema standartov po informacii, bibliotechnomu i izdatel'skomu delu. Organizacionno-raspor-yaditel'naya dokumentaciya. Trebovaniya k oformleniyu dokumentov.
URL: https://docs.cntd.ru/document/1200142871.
URL: http://publication.pravo.gov.ru/Document/View/0001202002070036.

7. *Gorshkov S.* Vvedenie v ontologicheskoe modelirovanie. Reviziya 2.4. OOO «TriniData», 2014-2018, 150 s.

## СВЕДЕНИЯ ОБ АВТОРАХ

**ЧУЙКОВА Виктория Владимировна** – старший преподаватель и аспирант кафедры «Информационная безопасность», ЮЗГУ. Область научных интересов: онтологические модели, семантический поиск, автоматизированные системы документооборота. Число научных публикаций – 15.

**Viktoriia Vladimirovna CHUIKOVA** – Senior Lecturer and aspirant of the Information security department, South-West State University. Research interests: ontological models, semantic search, automated document management systems. The number of publications – 15.

email: chuikova_v_v@bk.ru

ORCID: 0009-0000-6784-8135

***ТАНЫГИН Максим Олегович*** – д. т. н., профессор кафедры «Информационная безопасность», ЮЗГУ. Область научных интересов: анонимные одноранговые сети, аутентификация в многоагентных системах, автоматизированные системы документооборота. Число научных публикаций – 119.

***Maxim Olegovich TANYGIN*** – Doctor of technical sciences, professor. Professor of the Information security department, South-West State University. Research interests: anonymous peer-to-peer networks, authentication in multi-agent systems, automated document management systems. The number of publications – 119.

email: tanygin@yandex.ru

ORCID: 0000-0002-4099-1414

*Материал поступил в редакцию 25 марта 2025 года*