

НАУЧНЫЙ ЖУРНАЛ

# РАЗРАБОТКА МОБИЛЬНЫХ ПРИЛОЖЕНИЙ И КОМПЬЮТЕРНЫХ ИГР



Тут будет предисловие

А тут должно быть оглавление

# **Разработка системы мониторинга и контроля состояния растений.**

## **Компоненты и область применения**

*А.О. Нуреев*

*ФГИИТ, 1 курс магистратуры, 1 группа*

**Аннотация:** Производство сельскохозяйственной продукции в настоящее время тесно связано с применением цифровых технологий, элементов точного земледелия, автоматизацией и роботизацией сельского хозяйства. Особое внимание уделяется необходимости комплексного использования цифровых технологий и искусственного интеллекта для систем, способных осуществлять постоянный мониторинг и повышать и использования ресурсов путем автоматизации процессов. В данной статье приводится разбор применения технологий Интернета Вещей и цифровых протоколов, использующихся на сегодняшний день алгоритмов машинного обучения в контексте системы мониторинга и контроля состояния растений. Также, в данной работе показаны область применения системы такого рода: анализ готовности сельскохозяйственных культур к сбору урожая; отслеживание хода вегетации растений; прогнозирование показателей эффективности производства; своевременное выявление производственных рисков.

**Ключевые слова:** Интернет Вещей, машинное обучение, мониторинг, автоматизация сельскохозяйственного производства, распределенная система, анализ данных, Интернет-протоколы.

## **Введение**

Современные цифровые технологии, активно интегрирующиеся в повседневную жизнь, утверждают концепцию Интернета Вещей в центре своего внимания. Интернет Вещей представляет собой концепцию, в которой различные устройства способны взаимодействовать как с окружающей средой, так и между собой, освобождаясь от необходимости активного вмешательства человека в процессы функционирования системы.

На сегодняшний день практическое применение Интернета Вещей выходит за пределы ограниченного круга областей и охватывает различные секторы. Одним из значимых направлений внедрения является сельское хозяйство, в котором данная технология используется упрощения мониторинга производственных процессов и контроля состояния растений.

Актуальность темы данной работы заключается в том, что на сегодняшний день ощущается острая необходимость в автоматизации трудозатратных процессов сельского хозяйства. Автоматизация процессов ухода за растениями и их мониторинга, начиная от небольших участков и

теплиц, может сильно облегчить труд фермеров и повысить производительность их труда.

**Цель исследования** – разработка архитектуры и анализ системы мониторинга и контроля состояния растений с применением электронных датчиков в виде веб-сервиса.

**Объект исследования** – растениеводческие и сельскохозяйственные хозяйства, применяющие в своей деятельности новые цифровые технологии.

**Предмет исследования** – изучение влияния новых цифровых технологий на эффективность ведения сельского хозяйства.

### Компоненты Интернета Вещей в растениеводстве

Технологии Интернета Вещей существенно изменили традиционные сельскохозяйственные практики, облегчая сбор, анализ и автоматизацию данных. В результате, управление сельским хозяйством стало более точным и эффективным [1].

Традиционные методы полива и орошения сельскохозяйственных культур полагались на заранее установленные графики полива, что приводило к неэффективному использованию водных ресурсов и неравномерному распределению влаги. Интернет Вещей, в свою очередь, использует умные датчики, встроенные в почву, для контроля уровня влажности и передачи информации на центральные серверы. На основе собранных данных рассчитывается точное количество воды, необходимое для орошения различных участков. Автоматизированные системы полива могут адаптировать свою деятельность в зависимости от текущих данных, что позволяет значительно снизить потребление воды и избежать чрезмерного полива, который может вызвать эрозию почвы. Такой подход способствует оптимальному росту растений, что положительно сказывается на урожайности [2].

Помимо этого, современное сельское хозяйство сталкивается с постоянной угрозой заболеваний растений или появлении вредителей на

сельскохозяйственных участках [3]. Технологии Интернета Вещей помогают вовремя получить информацию об этом. Датчики, расположенные на месте, могут отслеживать такие параметры, как температура и влажность, которые влияют на поведение насекомых. Интернет Вещей также позволяет фермерам применять точечное распыление пестицидов, что сокращает потребность в массовом применении химикатов, снижает расходы и минимизирует экологический ущерб. Это также помогает точно определять, когда и где нужно вмешательство, что делает методы защиты растений более эффективными и экологически безопасными.

Эталонная организационная модель системы мониторинга Интернета Вещей (IoT-системы) представлена на (рис. 1.). Схема IoT-системы мониторинга включает в себя три уровня [4]: сенсорный уровень (sensor layer), сетевой уровень (network layer) и прикладной уровень (application layer).

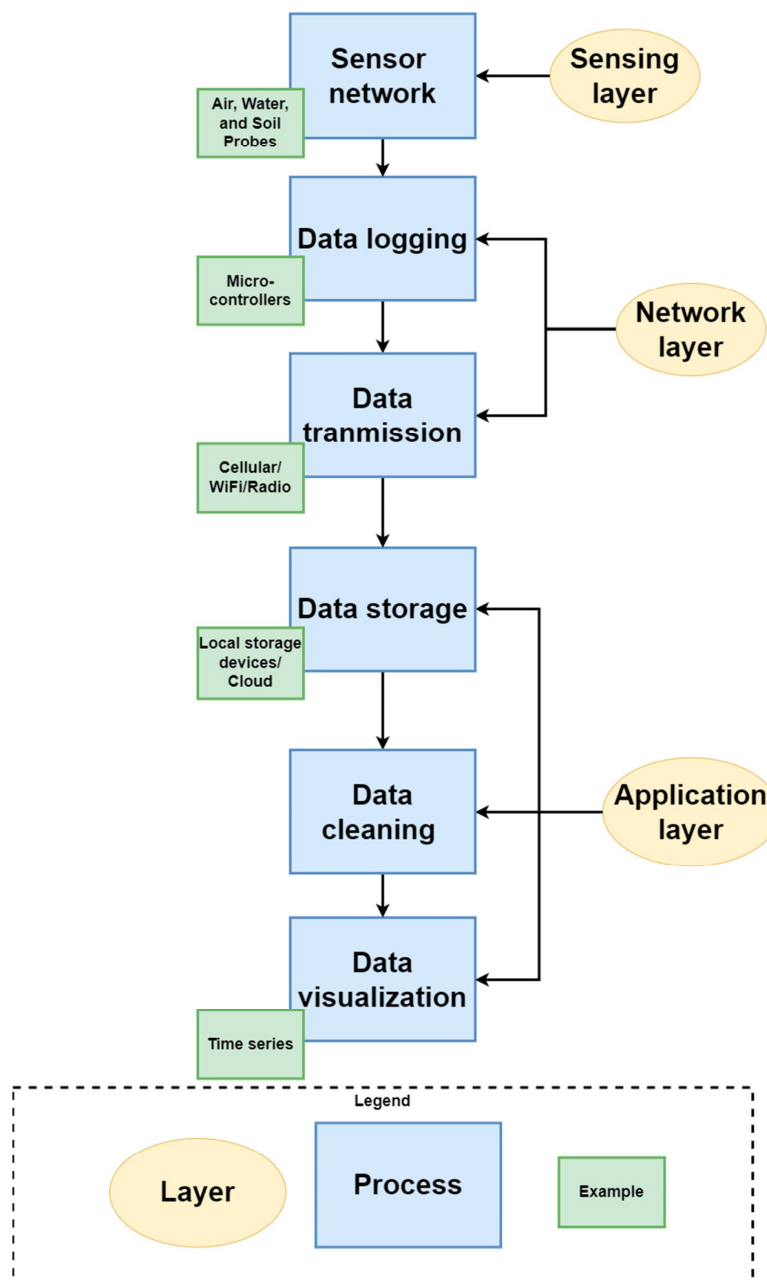


Рис. 1. Организационная модель IoT-системы мониторинга

На сенсорном уровне происходит сбор необходимой информации на местности. Датчики, сенсоры и прочие технические средства, объединённые в единую сеть, собирают необходимые данные. Для дальнейшей передачи полученных данных используются средства сетевого уровня. Задача сетевого уровня состоит своевременной, правильной и безопасной передаче показаний с датчиков на специально выделенный облачный сервер. Прикладной уровень IoT-системы мониторинга отвечает за хранение полученной информации, ее необходимую обработку и визуализацию данных в удобном для конечного пользователя виде.

## Протоколы передачи данных

Система передачи данных является одним из ключевых компонентов технологий Интернета вещей (IoT), используемых в сельском хозяйстве. Эффективное управление процессами в этой области требует получения информации в режиме реального времени, поскольку задержки в передаче данных могут привести к нарушению функционирования всей системы. На сегодняшний день беспроводные сенсорные сети (WSN) стали значительно популярнее проводных сетей, благодаря их эффективности, простоте обслуживания и относительной экономичности [5,6].

К современным технологиям передачи данных, которые находят применение в сельском хозяйстве, относят (рис. 2.):

**IEEE 802.15.4.** Стандартный протокол WSN IEEE 802.15.4 является одним из наиболее распространенных протоколов связи. Он использует уровни управления доступом к среде (MAC) и физические уровни (PHY) беспроводных персональных сетей с низкой скоростью передачи данных (LR-WPAN). Эти сети характеризуются низким энергопотреблением, простотой установки и низкими требованиями к обработке. Протокол обеспечивает эффективную передачу данных в пределах ограниченного диапазона, а также отличается длительным сроком службы батареи и низкой стоимостью.

**ZigBee.** Этот протокол, основанный на стандарте LR-WPAN, обеспечивает безопасную и надежную передачу данных между устройствами. ZigBee широко используется как глобальный стандарт связи в WSN, предоставляя гибкие топологии сетей, включая звездообразные, ячеистые и древовидные структуры. Основными преимуществами являются энергоэффективность и надежность, что делает его идеальным для применения в аграрной отрасли.

**BLE (Bluetooth Low Energy).** BLE используется в устройствах IoT для связи на коротких дистанциях (до 10 м) и при низкой пропускной способности. Его преимущества включают минимальное энергопотребление и быстрое

время инициализации. BLE применяется для сбора данных датчиков, таких как температура и влажность почвы, с помощью мобильных устройств. Однако данный протокол ограничен низким уровнем безопасности и возможностью взаимодействия только с двумя устройствами одновременно.

**6LoWPAN.** Протокол IPv6, поддерживающий топологии «звезда» и «ячеистая сеть», основан на стандарте IEEE 802.15.4. Его основными характеристиками являются низкая пропускная способность и энергоэффективность. Протокол используется для мониторинга климатических и почвенных условий, что важно для оптимизации сельскохозяйственных процессов.

**Протоколы сотовой связи.** Технологии 3G и 4G обеспечивают высокую скорость передачи данных и применяются для создания подземных беспроводных сетей. Основные недостатки включают высокое энергопотребление и более высокую стоимость эксплуатации, что ограничивает их применение в сельском хозяйстве.

**LoRa WAN.** Эта технология, основанная на стандарте IEEE 802.11ah, поддерживает связь на больших расстояниях при минимальном энергопотреблении. LoRa WAN широко используется в сельском хозяйстве благодаря своей способности охватывать обширные территории и обеспечивать надежное соединение.

**Wi-Fi.** Наиболее популярный протокол WSN, основанный на стандартах IEEE 802.11, используется для мониторинга климатических условий, управления орошением и других процессов. Основными недостатками являются уязвимость к неблагоприятным погодным условиям и проблемы с интерференциями, что может снижать скорость передачи данных.



TABLE V  
TYPICAL REPORTED WSN COMMUNICATION PROTOCOLS THAT WERE USED IN SMART GREENHOUSES

| WSN protocols | Topologies                  | Data transmission rate (bps) | Operating range      | Power Consumption |
|---------------|-----------------------------|------------------------------|----------------------|-------------------|
| IEEE 802.15.4 | Star and Peer-to-Peer [123] | 20-250 kbps [92]             | 10 m [124]           | Low [123]         |
| Zigbee        | Star, Tree and Mesh [125]   | 250 kbps [126]               | 10-100 m [69]        | Low [125]         |
| BLE           | Star and Tree [127]         | 1-2 Mbps [69]                | 30 m [69]            | Low [127]         |
| 6LowPAN       | Star and Mesh               | 0.3-50 kbps [69]             | 10-100 m [128]       | Low [128]         |
| Cellular      | 2G/3G frequency bands [129] | 10 Mbps [129]                | Several miles [129]  | High [129]        |
| LoRa WAN      | Star and Peer-to-peer [69]  | Less than 50 kbps [128]      | 5-50 km [128]        | Low [128]         |
| RFID          | Peer to peer [69]           | 50 tags/s [69]               | 10-20 cm [69]        | Ultra low [69]    |
| Wi-Fi         | Star [69]                   | 0.1-54 Mbps [128]            | Less than 92 m [128] | Low [128]         |

Рис. 2. Протоколы беспроводной связи, используемых в Интернете Вещей

### Применение алгоритмов машинного обучения

Машинное обучение является важным инструментом поддержки принятия решений для прогнозирования урожайности, включая информационную поддержку решений о наиболее подходящих культурах для выращивания, а также о том, что делать в период вегетации культур. С помощью алгоритмов машинного обучения можно учитывать широкий спектр факторов, таких как климатические условия, тип почвы, методы обработки земли, а также данные о состоянии посевов. Применение алгоритмов машинного обучения в прогнозировании урожайности предоставляет возможности для повышения эффективности сельскохозяйственного производства и снижения рисков, связанных с природными и климатическими изменениями [7].

Методы глубокого обучения нейронных сетей продемонстрировали выдающиеся результаты в задачах сельского хозяйства [8]. Алгоритмы глубоких нейронных сетей основаны на схожих концепциях с традиционными искусственными нейронными сетями, но включают в себя значительно больше скрытых слоев.

Основными алгоритмами глубокого обучения, применяемыми для предсказания урожайности, являются (рис. 3.) [9]:

**Глубокие нейронные сети (DNN).** Эти алгоритмы схожи с традиционными искусственными нейронными сетями, однако имеют больше

скрытых слоев, что позволяет им эффективно моделировать сложные зависимости в данных.

**Сверточные нейронные сети (CNN).** В отличие от полностью связанных сетей, CNN имеет меньше параметров для обучения благодаря использованию сверточных слоев, которые включают фильтры для обработки входных данных. Общая схема для моделей CNN заключается в том, что за одним или несколькими сверточными слоями следует слой объединения, и эта структура повторяется несколько раз, и, наконец, применяются полностью связанные слои.

**Сети с долгосрочной краткосрочной памятью (LSTM).** Данные сети являются разновидностью рекуррентных нейронных сетей (RNN), специально разработанных для работы с последовательными данными, что делает их особенно подходящими для задач прогнозирования. В отличие от традиционных RNN, LSTM имеют механизмы, которые позволяют моделировать долговременные зависимости, что крайне важно при работе с временными рядами данных, где прошлые события могут существенно влиять на будущее.

**Гибридные сети.** Данные сети объединяют различные алгоритмы глубокого обучения. Для прогнозирования урожайности тех или иных культур возможно объединение CNN, RNN, DNN, LSTM и других сетей между собой.

**Table 9**  
Distribution of deep learning algorithms.

| Algorithms used                | # of usages | Percentage (%) |
|--------------------------------|-------------|----------------|
| CNN                            | 10          | 30,30          |
| LSTM                           | 7           | 21,21          |
| DNN                            | 7           | 21,21          |
| Hybrid                         | 4           | 12,12          |
| Autoencoder                    | 1           | 3,03           |
| Multi-Task Learning (MTL)      | 1           | 3,03           |
| Deep Recurrent Q-Network (DQN) | 1           | 3,03           |
| 3D CNN                         | 1           | 3,03           |
| Faster R-CNN                   | 1           | 3,03           |
| Total                          | 33          | 100            |

Рис. 3. Используемые в сельском хозяйстве алгоритмы глубокого обучения

В итоге, использование алгоритмов машинного обучения для анализа и прогнозирования урожая может помочь фермерам оптимизировать сократить трудозатраты и повысить рентабельность своей деятельности. Машинное обучение позволяет оценить большой объем данных за короткий промежуток времени и выдать рекомендации относительно лучшего времени для полива, удобрения и посева сельскохозяйственных культур.

### **Выводы**

Система мониторинга и контроля состояния растений, использующая для сбора и агрегации данных технологии Интернета Вещей, а для анализа и прогнозирования – методы машинного обучения, сможет кардинально повысить уровень автоматизации основных производственных процессов сельских хозяйств, включая малые и средние производства [10].

Данная система способна решать следующие задачи:

1. Анализ готовности сельскохозяйственных культур к сбору урожая;
2. Отслеживание ход вегетации растений;
3. Прогнозирование показателей эффективности производства;
4. Своевременное выявление производственных рисков.

### **Литература**

1. Sreya John, P. J. Arul Leena Rose. Smart Farming and Precision Agriculture and Its Need in Today's World. 2024. [Электронный ресурс] [https://www.researchgate.net/publication/379129930\\_Smart\\_Farming\\_and\\_Precision\\_Agriculture\\_and\\_Its\\_Need\\_in\\_Today%27s\\_World](https://www.researchgate.net/publication/379129930_Smart_Farming_and_Precision_Agriculture_and_Its_Need_in_Today%27s_World)
2. З.В. Нагоев, В.М. Шуганов, А.У. Заммиев, К.Ч. Бжихатлов, З.З. Иванов. Разработка интеллектуальной интегрированной системы «Умное Поле». 2022. [Электронный ресурс] <https://cyberleninka.ru/article/n/razrabotka-intellektualnoy-integrirovannoy-sistemy-umnoe-pole>

3. Анищенко А.Н., Шутьков А.А. Agriculture 4. 0 как перспективная модель научно-технологического развития аграрного сектора современной России. 2019. [Электронный ресурс] <https://cyberleninka.ru/article/n/agriculture-4-0-kak-perspektivnaya-model-nauchno-tehnologicheskogo-razvitiya-agrarnogo-sektora-sovremennoy-rossii>

4. Rakiba Rayhana, Gaozhi Xiao, Zheng Liu. Internet of Things Empowered Smart Greenhouse. 2020. [Электронный ресурс] [https://www.researchgate.net/publication/340326857\\_Internet\\_of\\_Things\\_Empowered\\_Smart\\_Greenhouse](https://www.researchgate.net/publication/340326857_Internet_of_Things_Empowered_Smart_Greenhouse)

5. Носов А.В. анализ прикладных протоколов передачи данных для систем Интернета Вещей. 2021. [Электронный ресурс] <https://cyberleninka.ru/article/n/analiz-prikladnyh-protokolov-peredachi-dannyh-dlya-sistem-interneta-veschey>

6. Sarika Choudhary, Gaurav Meena. Internet of Things: Protocols, Applications and Security Issues. 2022. [Электронный ресурс] <https://www.sciencedirect.com/science/article/pii/S1877050922021019>

7. Vishal Meshram, Kailas Patil, Vidula Meshram, Dinesh Hanchate, S.D. Ram kteke. Machine learning in agriculture domain: A state-of-art survey. 2021. [Электронный ресурс] <https://www.sciencedirect.com/science/article/pii/S2667318521000106>

8. Zhicheng Ye, Xu Zhai, Tianlong She, Xiaoyan Liu, Yuanyuan Hong, Lihui Wang, Lili Zhang, Qiang Wang. Winter Wheat Yield Prediction Based on the ASTGNN Model Coupled with Multi-Source Data. 2024. [Электронный ресурс] <https://www.mdpi.com/2073-4395/14/10/2262>

9. Thomas van Klompenburg, Ayalew Kassahun, Cagatay Catal. Crop yield prediction using machine learning: A systematic literature review. 2020. [Электронный ресурс] <https://www.sciencedirect.com/science/article/pii/S0168169920302301>

10. С.М. Москалев, Н.В. Клименок-Кудинова. Искусственный интеллект и интернет вещей как инновационные методы совершенствования

агропромышленного сектора. 2018. [Электронный ресурс]

<https://cyberleninka.ru/article/n/iskusstvennyy-intellekt-i-internet-veschey-kak-innovatsionnye-metody-sovershenstvovaniya-agropromyshlennogo-sektora>

## **Рецензия на статью «Разработка системы мониторинга и контроля состояния растений. Компоненты и область применения»**

Статья посвящена применению технологий Интернета Вещей (IoT) и машинного обучения для мониторинга состояния растений в сельском хозяйстве. Автор рассматривает архитектуру IoT-системы, протоколы передачи данных и алгоритмы машинного обучения, подчеркивая их потенциал для автоматизации процессов. Работа актуальна и хорошо структурирована, но требует углубленного анализа технологий, практических примеров и критического обсуждения ограничений. Рекомендуется доработка для усиления научной и практической ценности.

Основные замечания касаются теоретической глубины исследования. В разделе о протоколах передачи данных перечислены технологии (ZigBee, LoRa WAN и др.), но отсутствует их критический анализ. Например, не указано, что BLE (Bluetooth Low Energy) неприменим для крупных хозяйств из-за малого радиуса действия, а LoRa WAN требует значительных инфраструктурных затрат. В разделе о машинном обучении алгоритмы (CNN, LSTM) описаны схематично без примеров их практического применения, таких как использование CNN для анализа мультиспектральных снимков посевов или LSTM для прогнозирования урожайности.

Практическая значимость работы снижается отсутствием конкретных кейсов внедрения описанных решений. Автор не приводит примеров реализации систем мониторинга, хотя такие проекты существуют (например, "Умное поле" в России или системы точного земледелия в ЕС). Не хватает количественных данных об эффективности технологий: насколько снижается водопотребление при использовании IoT-орошения или как алгоритмы машинного обучения повышают точность прогнозов урожайности.

Научная строгость работы страдает из-за неподкреплённых утверждений. Заявление о "кардинальном повышении уровня автоматизации" не подтверждено ссылками на исследования. Не обсуждаются важные ограничения технологий: вопросы кибербезопасности IoT-устройств, зависимость от интернет-покрытия в сельской местности, высокая стоимость внедрения для малых хозяйств.

Терминология и структура статьи требуют уточнений. В разделе о машинном обучении термин "гибридные сети" следует заменить на "гибридные архитектуры нейронных сетей". Рисунки 1-3 нуждаются в более подробных пояснениях. Например, на Рис. 1 не указано, какие конкретно датчики (влажности, температуры, освещённости) относятся к сенсорному уровню.

Для улучшения статьи рекомендуется: 1) дополнить анализ технологий сравнительной таблицей протоколов передачи данных по ключевым параметрам (дальность связи, энергопотребление, стоимость); 2) включить конкретные примеры применения алгоритмов машинного обучения; 3) добавить кейсы внедрения IoT в сельском хозяйстве с количественными показателями эффективности; 4) обсудить ограничения и риски технологий; 5) улучшить оформление рисунков и устранить терминологические неточности.

Статья представляет собой добротный обзор современных технологий для цифровизации сельского хозяйства, но требует доработки для повышения научной строгости и практической ценности. Рекомендуется принять статью к публикации после соответствующей доработки с учетом высказанных замечаний. Рецензент предлагает рассматривать данный отзыв как основу для дальнейшего совершенствования работы.

# Реализация ядра Jupyter для PascalABC.NET для работы в многопользовательской среде JupyterHub

*С.Н. Валяев*

*Южный федеральный университет, Ростов-на-Дону*

**Аннотация:** В данной статье рассматривается реализация ядра Jupyter для языка программирования PascalABC.NET, адаптированного для функционирования в малопроизводительной многопользовательской среде JupyterHub. Описаны архитектурные особенности разработанного ядра и его интеграция с Jupyter.

**Ключевые слова:** проект jupyter, jupyter notebook, pascalabc.net, jupyter kernel, jupyterhub, удалённое исполнение кода.

## Введение

Jupyter — это мощная интерактивная среда для вычислений, анализа данных и визуализации, получившая широкое распространение в академической и исследовательской среде. Основным преимуществом Jupyter является его архитектура, основанная на ядрах, которые позволяют исполнять код на различных языках программирования в рамках единого интерфейса Jupyter Notebook. Это делает платформу гибким инструментом как для преподавания, так и для разработки научных прототипов и проведения экспериментов.

PascalABC.NET — современный компилятор языка Pascal, ориентированный на образовательные цели. Он сочетает классический Pascal с возможностями платформы .NET, включая поддержку ООП, обобщений, лямбда-выражений и других современных средств. Благодаря интуитивно понятному синтаксису и широким возможностям, язык активно используется в школьном и вузовском обучении программированию.

В данной статье описана реализация ядра Jupyter для языка PascalABC.NET, ориентированная на работу в малопроизводительной среде JupyterHub.



## Реализация и принцип работы ядра

Для обеспечения исполнения кода на языке PascalABC.NET в интерактивной среде Jupyter было необходимо реализовать поддержку протокола взаимодействия между Jupyter Notebook (далее фронтенд) и ядром, как это описано в спецификации Jupyter Messaging Protocol [1]. Данный протокол определяет формат обмена сообщениями, необходимый для корректной обработки команд, вывода результатов, ошибок и интерактивного взаимодействия. Для решения этой задачи был использован промежуточный слой — ядро IPython, у которого была переопределена логика исполнения кода. Это позволило сосредоточиться исключительно на обработке кода PascalABC.NET, не вдаваясь в низкоуровневую реализацию транспортного протокола и структуры сообщений.

При поступлении кода ядро инициирует процесс его компиляции и исполнения. На первом этапе ядро информирует фронтенд о начале выполнения и очищает предыдущее содержимое вывода. Далее исходный код передаётся компилятору, который либо возвращает сообщение об ошибке, которое передаётся на фронтенд, либо скомпилированный исполняемый файл.

В случае успешной компиляции файл сохраняется во временное хранилище и запускается. Ядро создаёт два потока — первый отвечает за перенаправление стандартного вывода программы на фронтенд, а второй за перенаправление сообщений об ошибках и отслеживание специальных сигналов [2], используемых в PascalABC.NET для инициализации пользовательского ввода. В случае получения такого сигнала Jupyter запрашивает у пользователя данные через стандартный ввод и передаёт их обратно в программу. После исполнения кода или его принудительной

остановки ядро возвращает фронтенду информацию о завершении выполнения.

### Работа ядра в Jupyterub

Функционирование ядра PascalABC.NET в среде JupyterHub [3] предъявляет особые требования к управлению ресурсами и обеспечению устойчивости при одновременной работе большого числа пользователей. В условиях ограниченной вычислительной мощности необходимо предотвращать ситуации, при которых чрезмерное количество одновременно исполняемых задач может привести к деградации производительности или полной недоступности сервиса.

С этой целью в реализованном ядре используется механизм межпроцессного взаимодействия — POSIX-семафор, с помощью которого контролируется количество одновременно выполняемых процессов. При поступлении нового запроса на выполнение кода ядро пытается захватить семафор; если максимальное допустимое число параллельных запусков уже достигнуто, выполнение блокируется до освобождения одного из ресурсов. Это позволяет гарантировать, что система не будет перегружена.

Дополнительно, для снижения нагрузки на сервер и ускорения компиляции кода был разработан отдельный сервис, принимающий исходный код по протоколу HTTP и производящий его трансляцию с использованием одного из нескольких параллельно работающих экземпляров компилятора PascalABC.NET.

Разделение функций компиляции и исполнения между двумя изолированными компонентами обеспечивает гибкость и более эффективное использование ограниченных ресурсов JupyterHub, что критично для образовательных платформ с множеством активных пользователей.

## Заключение

В ходе проведённой работы было разработано ядро, обеспечивающее выполнение кода на языке PascalABC.NET в многопользовательской среде JupyterHub. Внедрение семафоров для управления параллельным выполнением программ обеспечили устойчивость системы в условиях ограниченных вычислительных ресурсов. Разделение процессов компиляции и исполнения посредством отдельного сервиса позволило повысить производительность и снизить нагрузку на сервер.

## Литература

1. Протокол обмена сообщениями Jupyter. // URL: <https://jupyter-client.readthedocs.io/en/latest/messaging.html> (дата обращения: 15.04.2025).
2. Пахом А.А. Реализация ядра Jupyter Notebook для PascalABC.NET с возможностью интерактивного ввода. Ростов-на-Дону: ЮФУ, 2022. С. 19-20.
3. JupyterHub: Масштабируемые вычисления для интерактивных вычислительных сред. // URL: <https://jupyter.org/hub> (дата обращения: 15.04.2025).

## References

1. Jupyter Project. Jupyter Messaging Protocol. // URL: <https://jupyter-client.readthedocs.io/en/latest/messaging.html> (data obrashhenija: 15.04.2025).
2. Pahom A.A. Implementation of the Jupyter Notebook kernel for PascalABC.NET with the possibility of interactive input. Rostov-on-Don: SFedU, 2022. pp. 19-20.
3. Jupyter Project. JupyterHub: Scalable computing for interactive computing environments. // URL: <https://jupyter.org/hub> (data obrashhenija: 15.04.2025).

## Рецензия на статью

### **«Реализация ядра Jupyter для PascalABC.NET для работы в многопользовательской среде JupyterHub»**

#### **Резюме**

Статья посвящена разработке ядра Jupyter для языка PascalABC.NET, адаптированного для работы в многопользовательской среде JupyterHub. Авторы описывают архитектуру решения, основанную на протоколе Jupyter Messaging Protocol

и модифицированном ядре IPython, обеспечивающем компиляцию и исполнение кода.

Особое внимание уделено управлению ресурсами: использованию POSIX-семафоров для ограничения параллельных процессов и выделенному HTTP-сервису компиляции, что повышает устойчивость системы при высокой нагрузке. Решение ориентировано на образовательные задачи, позволяя эффективно использовать PascalABC.NET в интерактивной среде Jupyter с сохранением стабильности работы.

Актуальность поставленной проблемы недостаточно аргументирована. Мои замечания касаются следующих положений статьи:

#### **Критические замечания**

##### **1. Недостаточное обоснование выбора PascalABC.NET**

В статье не приведены сравнительные данные о востребованности PascalABC.NET в образовательной среде, что затрудняет оценку значимости работы.

##### **2. Отсутствие количественных показателей эффективности**

Заявленные преимущества (снижение нагрузки, повышение производительности) не подкреплены метриками (например, время отклика, максимальная нагрузка).

### 3. Неполное описание ограничений

Не указано, как система ведёт себя при ошибках компиляции или аварийном завершении процессов, а также возможные сценарии отказа.

### 4. Слабая проработка безопасности

Не рассмотрены потенциальные риски при работе с пользовательским вводом (например, инъекции кода) и методы их нейтрализации.

## Мелкие правки

- **Раздел «Введение»:** Уточнить, почему именно JupyterHub выбран как целевая платформа.
- **Рисунки и схемы:** Добавить графическое представление архитектуры для наглядности.
- **Формулировки:**
  - «малопроизводительной среде» → «среде с ограниченными ресурсами»;
  - «деградации производительности» → «снижению производительности».

## Рекомендация редактору

**Требует доработки.** Статья представляет практический интерес, но нуждается в дополнении:

- количественных данных тестирования,

- анализа ограничений и безопасности,
- уточнения мотивации выбора технологий.

После исправления работа может быть рекомендована к публикации.

# Дообучение больших языковых моделей с использованием методов LoRA и RAG для разработки интегрируемых ботов-помощников

Д.А. Махов

*Южный федеральный университет, Ростов-на-Дону*

**Аннотация:** В данной статье рассматриваются методы дообучения больших языковых моделей (LLM) с использованием техник LoRA (Low-Rank Adaptation) и RAG (Retrieval-Augmented Generation) для создания интегрируемых ботов-помощников. Особое внимание уделено адаптации моделей к специфическим доменам, повышению вычислительной эффективности и интеграции знаний. Предложенные подходы демонстрируют потенциал для разработки интеллектуальных систем, способных эффективно взаимодействовать с пользователями в различных контекстах.

**Ключевые слова:** Большие языковые модели, дообучение, LoRA, RAG, боты-помощники, адаптация к доменам, вычислительная эффективность, интеграция знаний.

## Введение

Большие языковые модели (LLM), такие как GPT-3, LLaMA и другие, зарекомендовали себя как мощные инструменты для обработки естественного языка (NLP), включая генерацию текста, диалоговые системы и ответы на вопросы. Однако их использование в специализированных областях сталкивается с рядом вызовов. Во-первых, стандартное дообучение моделей на специфических данных требует значительных вычислительных ресурсов, что делает процесс дорогостоящим и трудоёмким. Во-вторых, знания, заложенные в модели на этапе предобучения, остаются статичными, что ограничивает их применимость в динамично развивающихся доменах [1]. Для решения этих проблем в данной работе исследуются технологии LoRA и RAG как средства адаптации LLM к конкретным задачам. LoRA позволяет дообучать модели, изменяя лишь небольшую часть параметров [2], а RAG интегрирует внешние источники знаний для повышения актуальности ответов [3]. Цель исследования — разработка методологии дообучения LLM с использованием этих технологий для создания эффективных и интегрируемых ботов-помощников, способных работать в различных доменах.

## Обзор литературы

Современные исследования в области дообучения больших языковых моделей активно развиваются, предлагая новые подходы к их адаптации. Одной из ключевых работ является статья Льюиса и соавторов "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks" (2020) [3]. Авторы представили метод RAG, который сочетает генеративные возможности LLM с механизмом поиска информации из внешних источников. Этот подход позволяет моделям генерировать ответы, основанные не только на внутренних знаниях, но и на актуальных данных, что особенно важно для

задач, требующих высокой точности и свежести информации, таких как анализ документов или ответы на вопросы.

Другим значимым вкладом в область дообучения является технология LoRA, описанная в работе Ху и соавторов "LoRA: Low-Rank Adaptation of Large Language Models" (2021) [2]. LoRA предлагает метод параметрически-эффективного дообучения, при котором вместо полной перестройки модели обучаются низкоранговые матрицы, добавляемые к весам. Это существенно снижает вычислительные затраты и делает адаптацию моделей доступной даже для устройств с ограниченными ресурсами.

Также стоит отметить исследование Лю и соавторов "Parameter-efficient fine-tuning of large-scale pre-trained language models" (2023) [4], где представлен обзор различных методов параметрически-эффективного дообучения, таких как адаптеры, BitFit и LoRA. Авторы подчеркивают, что такие подходы обеспечивают производительность, сравнимую с полным дообучением, при значительном сокращении числа обучаемых параметров, что делает их перспективными для практического применения в задачах адаптации моделей.

Работа Гурурангана и соавторов "Don't Stop Pretraining: Adapt Language Models to Domains and Tasks." (2020) [1] акцентирует внимание на дообучении LLM для узких областей, таких как медицина или право. Исследование демонстрирует, что использование специализированных датасетов повышает точность моделей в целевых доменах, что согласуется с задачами данной работы.

Статья Топпирана и соавторов "LaMDA: Language Models for Dialog Applications" (2022) [5] рассматривает создание диалоговых систем на основе LLM. Авторы подчеркивают важность интеграции внешних источников знаний, таких как системы поиска и базы данных, для повышения фактической точности и качества диалогов, что служит основой для выбранного в данной работе подхода.

### **Выбранный подход**

Предлагаемая методология сочетает использование LoRA и RAG для дообучения больших языковых моделей с целью создания интегрируемых ботов-помощников. Подход включает два основных этапа. На первом этапе модель, например LLaMA, дообучается на специализированном датасете с применением LoRA, что позволяет адаптировать модель к конкретному домену, сохраняя эффективность за счёт минимизации числа обучаемых параметров [2]. На втором этапе в архитектуру бота интегрируется компонент RAG, обеспечивающий доступ к актуальным данным из внешних источников, таких как базы знаний или документы, во время работы системы [3].

Комбинация этих методов адаптирует модель к целевому домену, усваивая специфические знания и стиль общения, а также обеспечивает динамическое обновление информации, устраняя проблему устаревания данных. Минимизация вычислительных затрат делает подход применимым для различных платформ, включая устройства с ограниченными ресурсами. Такой подход



обеспечивает гибкость ботов-помощников, позволяя им работать в разных доменах без необходимости повторного полного обучения.

### **1. Обзор технологий LoRA и RAG**

Для адаптации больших языковых моделей к специализированным задачам предлагается использование двух взаимодополняющих технологий: LoRA и RAG. Технология LoRA (Low-Rank Adaptation) представляет собой метод параметрически-эффективного дообучения, который предполагает добавление к весам модели низкоранговых матриц, что позволяет существенно сократить вычислительные затраты при сохранении высокого качества адаптации [2]. Этот подход особенно актуален для задач, требующих быстрой и ресурсосберегающей адаптации к специфическим доменам, например, в области технической поддержки, где модель должна учитывать уникальную терминологию и контекст. В свою очередь, RAG объединяет генеративные возможности языковой модели с механизмом поиска и извлечения информации из внешних источников, таких как корпоративные базы знаний, FAQ или документация [3]. Это обеспечивает генерацию ответов, которые не только опираются на внутренние знания модели, но и учитывают актуальные данные, что повышает точность и релевантность отклика. Комбинация LoRA и RAG позволяет создавать системы, способные эффективно адаптироваться к доменным требованиям и динамически обновлять знания, что делает их оптимальными для разработки современных ботов-помощников [2, 3].

### **2. Архитектура бота-помощника**

Архитектура предлагаемого бота-помощника базируется на предобученной большой языковой модели, такой как LLaMA, которая выступает в качестве основы для дальнейшей адаптации. Для обеспечения соответствия модели требованиям конкретного домена, например, технической поддержки, применяется технология LoRA, которая позволяет дообучать модель на специализированных данных, таких как запросы пользователей и ответы специалистов [2]. Это обеспечивает точное усвоение терминологии и контекста, характерных для целевой области. Одновременно с этим в архитектуру интегрируется компонент RAG, который предоставляет доступ к внешним источникам знаний, включая корпоративные документы, руководства и базы часто задаваемых вопросов [3]. Такое сочетание позволяет боту генерировать ответы, которые одновременно точны с точки зрения языка и актуальны с точки зрения содержания. Взаимодействие между LoRA и RAG организовано таким образом, что первый компонент отвечает за языковую адаптацию и точность, а второй — за динамическое обогащение ответов внешними данными. Подобная архитектура обеспечивает гибкость и масштабируемость системы, позволяя ей эффективно функционировать в различных доменах без необходимости значительных вычислительных ресурсов [2, 3, 5].

### **3. Этапы реализации**

Реализация предложенной методологии включает несколько последовательных этапов, направленных на интеграцию технологий LoRA и RAG в единую систему. Для работы с большими языковыми моделями используются современные инструменты, такие как библиотека Hugging Face Transformers для управления моделями и фреймворк PyTorch для проведения дообучения. На этапе применения LoRA подготавливается специализированный датасет, включающий запросы пользователей и соответствующие ответы специалистов, после чего проводится настройка гиперпараметров, таких как ранг матриц и скорость обучения, для оптимизации процесса дообучения [2]. Параллельно осуществляется настройка компонента RAG, которая включает индексацию базы знаний с использованием подходящих инструментов поиска, организацию извлечения релевантных документов по пользовательским запросам и интеграцию полученных данных в процесс генерации текста [3]. Заключительным этапом является оптимизация системы, направленная на достижение баланса между скоростью обработки запросов и качеством генерируемых ответов, что обеспечивает возможность работы бота в реальном времени. Такой подход позволяет создать эффективную и адаптивную систему, способную удовлетворять требованиям различных доменов [2, 3].

### **Заключение**

Предложенный подход сочетает параметрически-эффективное обучение с динамическим доступом к внешним знаниям, что позволяет адаптировать модели к специализированным задачам с минимальными затратами. Эта методология демонстрирует значительный потенциал для повышения качества и гибкости ботов-помощников, делая их применимыми в широком спектре доменов — от технической поддержки до образовательных систем. Перспективы дальнейших исследований включают оптимизацию интеграции LoRA и RAG, а также разработку более точных методов оценки производительности таких систем.

### **Литература**

1. Gururangan, S., et al. "Don't Stop Pretraining: Adapt Language Models to Domains and Tasks." arXiv preprint arXiv:2004.10964, 2020.
2. Hu, E. J., et al. "LoRA: Low-Rank Adaptation of Large Language Models." arXiv preprint arXiv:2106.09685, 2021.
3. Lewis, P., et al. "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks." arXiv preprint arXiv:2005.11401, 2020.
4. Liu, H., et al. "Parameter-efficient fine-tuning of large-scale pre-trained language models." Nature Machine Intelligence, 2023. DOI:10.1038/s42256-023-00626-4.

5. Thoppilan, R., et al. "LaMDA: Language Models for Dialog Applications."  
arXiv preprint arXiv:2201.08239, 2022.

# Рецензия на статью Д.А. Махова «Дообучение больших языковых моделей с использованием методов LoRA и RAG для разработки интегрируемых ботов-помощников»

## Резюме

Статья посвящена методам дообучения больших языковых моделей (LLM) с использованием техник LoRA и RAG для создания интегрируемых ботов-помощников. Авторы детально рассматривают механизмы параметрически эффективного обучения и динамической генерации ответов на основе внешних источников знаний. Представлена архитектура решения, этапы реализации и обоснование выбора подходов. Работа актуальна и опирается на современные исследования в области NLP. Однако статья требует структурной доработки, уточнения исследовательской новизны и корректировки отдельных формулировок для повышения академической строгости.

## Критические замечания

- 1. Отсутствие эмпирической оценки:** В статье не представлены результаты практического применения предложенной методологии (например, точность, F1-мера, latency и т.д.). Это существенно снижает убедительность выводов.
- 2. Недостаточная конкретизация используемых данных:** Не указаны примеры датасетов, применённых для дообучения и тестирования. Это затрудняет воспроизводимость и оценку применимости подхода.

## Мелкие правки

- **Уточните формулировку в аннотации**, подчеркнув, что используется *комбинированный подход* с интеграцией LoRA и RAG, а не их параллельное или независимое применение.
- В описании этапов реализации отсутствует информация о возможных сложностях интеграции LoRA и RAG в единую архитектуру. Следует сбалансировать обзор с указанием возможных рисков или условий неприменимости.
- Раздел «Этапы реализации» желательно структурировать в виде **нумерованного списка**, отражающего пошаговый процесс: выбор модели, настройка LoRA, построение индексной базы, интеграция RAG и пр.
- В разделе «Выбранный подход» рекомендуется добавить **иллюстрацию архитектуры системы**, которая бы визуализировала

взаимодействие компонентов для лучшего восприятия структуры решения и его масштабируемость.

- Предложение в разделе «Обзор технологий LoRA и RAG» (стр. 4) «что делает их оптимальными...» — избыточно субъективно. Следует изменить формулировку на более нейтральную.
- Добавьте ссылки на инструменты, упомянутые в разделе «Этапы реализации» (стр. 5).

#### Рекомендация редактору

**Требуется доработка — статья демонстрирует научную новизну и актуальность темы, однако нуждается в конкретизации применяемых методов, уточнении терминологии, добавлении экспериментальных результатов, а также в устранении стилистических недочётов. После внесения указанных правок работа может быть рекомендована к публикации.**

# Нейросеть для распознавания нот игры на пианино

Е.А. Доц

<sup>1</sup>Южный федеральный университет, Ростов-на-Дону

**Аннотация:** в данной работе рассматривается разработка и оценка системы автоматического распознавания нот на аудиозаписях игры на пианино с использованием глубоких нейронных сетей. Предложенная методика включает этапы преобразования аудиосигнала в спектрограмму, семплирование нотных событий и применение гибридной архитектуры «сверточная сеть + рекуррентная сеть» для детектирования последовательности нот. Представлены детальные описания формирования специализированного датасета, методов аугментации звуковых данных и результатов валидации на публичных и собственных записях. Полученные метрики демонстрируют высокую точность детекции и транскрипции нот, сопоставимую с существующими решениями.

**Ключевые слова:** распознавание нот, автоматическая транскрипция, спектрограмма, сверточные нейронные сети, рекуррентные сети, piano roll, обработка аудиосигналов, глубокое обучение, augmentation.

## Введение

Автоматическая транскрипция музыки — одна из центральных задач музыкальной информатики и MIR (Music Information Retrieval). Особый интерес представляет распознавание нот игры на фортепиано, поскольку инструмент воспроизводит множество одновременно звучащих частотных компонентов и обладает сложным динамическим диапазоном. Современные методы глубокого обучения позволяют перейти от классических DSP-алгоритмов к гибридным решениям, объединяющим свёрточные и рекуррентные нейронные сети для извлечения временно-частотных признаков и моделирования последовательности нотных событий.

## Формирование и аугментация датасета

Для обучения модели собран собственный датасет, включающий:

1. MAESTRO (MIDI and Audio Edited for Synchronous TRacks and Organization) —  $\approx 200$  часов записей игры на рояле с аннотацией нотных событий [1].
2. MAPS (MIDI Aligned Piano Sounds) — записи синтетических и акустических аудио с точными метками [2].

3. Собственные записи профессиональных исполнителей ( $\approx 10$  часов), дополненные ручной разметкой.

Пре- и пост-обработка

- Все аудиофайлы приводились к частоте дискретизации 16 kHz, моно.
- Срезались тишина в начале и конце записи.
- Генерация спектрограмм с размером окна 2048 сэмплов, шаг в 512.

Аугментация звука

- Изменение тона:  $\pm 2$  полутона.
- Изменение скорости:  $\pm 10\%$ .
- Добавление шума: гауссов шум на уровне  $-40$  дБ.
- Реверберация и имитация комнатных эффектов.

### Архитектура модели

Мы используем двухэтапную архитектуру:

1. **Сверточная сеть (CNN)** для извлечения локальных признаков из спектрограммы.
  - 5 блоков Conv2D с батчнормом и ReLU, пулинг  $2 \times 2$  после каждого блока.
2. **Рекуррентная сеть (Bi-LSTM)** для учёта временной зависимости между кадрами.
  - Один двухнаправленный слой LSTM с 256 состояниями.
3. **Классификатор (Dense + Softmax)**, предсказывающий для каждого кадра вероятность событий начала/конца каждой из 88 клавиш.

### Обучение и валидация

- **Loss:** бинарная кросс-энтропия для каждого класса (key on/off).
- **Optimizer:** Adam,  $lr = 1e-4$ , с редукцией  $lr$  по plateau.
- **Batch size:** 16 спектрограмм длиной  $\approx 5$  сек.
- **Epochs:** 50, ранняя остановка по валидации.

На валидационной выборке MAESTRO модель достигает **F1** при детекции on-нотов  $\approx 0.87$  и **note-wise accuracy**  $\approx 0.83$ .

| Модель            | F1-score on-notes | Note accuracy | Inference time (мс) |
|-------------------|-------------------|---------------|---------------------|
| RNN Only          | 0.75              | 0.71          | 35                  |
| CNN + LSTM (ours) | <b>0.87</b>       | <b>0.83</b>   | 48                  |
| Transf-RNN        | 0.84              | 0.81          | 120                 |

Таблица 1. Результаты обучения

### Выводы

Предложенный подход демонстрирует высокую точность автоматической транскрипции игры на фортепиано на базовых аудио-датасетах. В дальнейшем планируется:

1. Интегрировать **трансформерные блоки** для лучшего учёта глобального контекста.
2. Расширить архитектуру двумя выходами для оценки **velocity** (динамики нажатия) вместе с детекцией on/off.
3. Оптимизировать модель для работы в режиме **реального времени** на встраиваемых устройствах.

### Литература

1. Hawthorne C., et al. “Enabling Factorized Piano Music Modeling and Generation with the MAESTRO Dataset.”
2. Emiya V., et al. “MAPS: MIDI-Aligned Piano Database for Automatic Music Transcription.”
3. Kelz R., et al. “Automatic Music Transcription Using Convolutional Networks and Data Augmentation.” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2022.
4. Sigtia S., Benetos E., Dixon S. “An End-to-End Neural Network for Polyphonic Piano Music Transcription.”
5. Bittner R. M., et al. “Deep Saliency Representations for F0 Estimation in Polyphonic Music.”



6. Ullrich N., et al. “Music Transcription with Invertible Flow Models.”

## Рецензия на статью «Нейросеть для распознавания нот игры на пианино»

Рецензируемая статья особенно актуальна в эпоху нейронных сетей. Актуальность поставленной задачи аргументирована. В данной работе описан подход к решению задачи автоматического распознавания нот на аудиозаписях игры на фортепиано, заключающийся в применении гибридной архитектуры «сверточная сеть + рекуррентная сеть». Описан набор данных, используемый для обучения нейронной сети. Представлены результаты применения разработанного решения, и приведена сравнительная характеристика с существующими решениями. Также автор указал дальнейшие планы по модификациям разработанной системы.

### Сильные стороны статьи:

- Актуальность темы: Автоматическая транскрипция музыки, особенно для сложных инструментов, таких как фортепиано, остаётся важной задачей в музыкальной информатике. Разработка эффективных методов имеет практическое значение для музыкального образования и анализа.
- Современный подход: Использование гибридной архитектуры CNN + RNN является обоснованным выбором для задач, связанных с обработкой аудиоданных с временной структурой. Такой подход соответствует текущим тенденциям в области глубокого обучения.
- Качественный датасет: Авторы используют как публичные датасеты (MAESTRO, MAPS), так и собственные записи, что повышает разнообразие данных. Применение аугментации, включая изменение высоты тона и добавление шума, способствует обучению устойчивой модели.
- Положительные результаты: Модель достигла F1-меры 0.87 для обнаружения начала нот и точности 0.83 на уровне нот, превосходя альтернативные подходы, такие как RNN-only и Transf-RNN, что подтверждается сравнительным анализом.

### Критические замечания:

Критических важных замечаний к статье нет

### Мелкие правки:

- Некоторые ссылки из списка литературы (например, [3], [4], [5], [6]) не упоминаются в тексте;
- В таблице №1: уточнить, что представляют собой модели "RNN Only" и "Transf-RNN";
- Терминология: заменить "батчнормом" на "батч-нормализацией";

### Рекомендации:

- Для более всесторонней оценки производительности рекомендуется добавить метрики, такие как точность (precision), полнота (recall) и точность на уровне кадров (frame-wise accuracy).

# Определение трехмерной позы человека по фотографии и её применение к 3D-модели

*Е. А. Колесникова*

**Аннотация:** В статье рассматриваются современные методы определения трехмерной позы человека по фотографии и их применение для анимации 3D-моделей. Обзор включает классические и нейросетевые подходы, такие как OpenPose и MediaPipe. Приведен пример реализации с использованием MediaPipe и blender для получения ключевых точек тела и переноса позы на 3D-модель. Обсуждаются области применения - от фитнеса до интерактивного обучения. Статья подчеркивает актуальность и эффективность данных технологий в цифровых приложениях.

**Ключевые слова:** трехмерная поза человека, 3D-модель человека, определение позы, MediaPipe, реконструкция позы, анимация 3D-модели, ключевые точки тела (keypoints), риггинг, деформация 3-D модели, обработка изображений, blender.

## Введение

Определение трехмерной позы человека по фотографии - одна из ключевых задач в области компьютерного зрения и искусственного интеллекта, которая приобретает все большую актуальность в современном цифровом мире. Возможность точно восстанавливать положение и ориентацию человеческого тела в пространстве на основе обычного изображения открывает широкие перспективы для различных сфер деятельности.

Сферы применения технологий 3D-позирования разнообразны и включают фитнес и спорт, где автоматический анализ позы помогает контролировать правильность выполнения упражнений и снижать риск травм. В области развлечений и видеоигр такие методы позволяют создавать реалистичные анимации персонажей и цифровых аватаров без необходимости использования дорогостоящих систем захвата движения. В медицине и реабилитации трехмерное отслеживание движений способствует мониторингу состояния пациентов и разработке индивидуальных программ восстановления.

## Обзор существующих решений

Современные технологии определения позы человека по фотографии базируются преимущественно на методах глубокого обучения, в частности на

сверточных нейронных сетях (CNN), которые обеспечивают высокую точность и эффективность решения задачи.

#### Современные нейросетевые методы

- OpenPose - одна из первых систем реального времени для многопользовательского определения позы. Использует архитектуру на базе VGG-19 и многоступенчатые CNN для выделения ключевых точек и их связывания в скелет. Позволяет выделять до 135 ключевых точек тела, рук, лица и стоп.
- AlphaPose - ориентирована на высокоточную локализацию позы у нескольких людей. Включает модули Spatial Transformer Network и Non-Maximum Suppression для повышения точности и устойчивости к перекрытиям.
- DeepCut и DeeperCut - методы «снизу вверх», которые сначала распознают все части тела на изображении, а затем группируют их по отдельным людям с помощью оптимизационных алгоритмов.
- MediaPipe- легковесный и быстрый фреймворк от Google, позволяющий в реальном времени отслеживать 33 ключевые точки тела и лица с высокой точностью, что делает его удобным для мобильных и встроенных приложений

Существующих технологий для деформации 3-D модели намного меньше, ввиду того что каждое появляющееся решение очень быстро облагается авторским правом и начинает использоваться конкретной компанией с целью заработка. Среди доступных методов известен способ изменения 3-D модели с помощью перемещения ключевых точек и дальнейшей интерполяции с целью восстановления формы модели и с использованием скелета модели. Немногие открытые решения дают возможность взаимодействия с костями 3-D модели, среди них доступны игровые движки (Godot, Unity) и программа для 3-D графики Blender.

## **Использование технологий определения 3D-позы в десктопном приложении для художников**

Современные методы определения трехмерной позы человека по фотографии открывают новые возможности для создания интерактивных инструментов, ориентированных на digital-художников. В рамках разработки десктопного приложения, предназначенного для художников, планируется интеграция технологии автоматического распознавания позы с целью упрощения и ускорения процесса создания реалистичных поз.

Основной функционал приложения будет включать загрузку фотографии с человеком, по которой с помощью нейросетевых моделей (например, MediaPipe) будет автоматически определяться трехмерная поза. Полученные ключевые точки и параметры суставов будут использоваться для деформации 3D-модели персонажа, размещаемой на виртуальной сцене. Такая деформация позволит художнику быстро получить базовую позу, соответствующую реальному человеку, что значительно сокращает время на подготовку и настройку модели, а также упрощает поиск референсов.

Далее на деформированную 3D-модель будет накладываться несколько слоев для рисования - как в традиционных графических редакторах. Это позволит художнику работать непосредственно поверх модели.

Таким образом, интеграция технологий определения 3D-позы в десктопное приложение для digital-художников позволит объединить автоматизацию и творческую свободу, облегчая создание реалистичных и выразительных персонажей и сцен. Это решение станет мощным инструментом для профессионалов и любителей, стремящихся повысить качество и скорость своей работы. А также значительно понизит порог входа для творческих людей в сферу рисования людей.

### **Разработка приложения**

В процессе начальной разработки приложения для диджитал-художников был проведён анализ доступных 3D-моделей, и подходящая модель человека была найдена на одном из стоковых ресурсов. Это позволило

сэкономить время на создании базовой заготовки и сразу приступить к интеграции функционала позирования.

Для автоматического определения позы по фотографии было принято решение использовать библиотеку MediaPipe, так как она предоставляет готовые инструменты для извлечения 33 трёхмерных ключевых точек тела (Рис. 1) с высокой точностью и скоростью, а также легко интегрируется в Python-проекты.

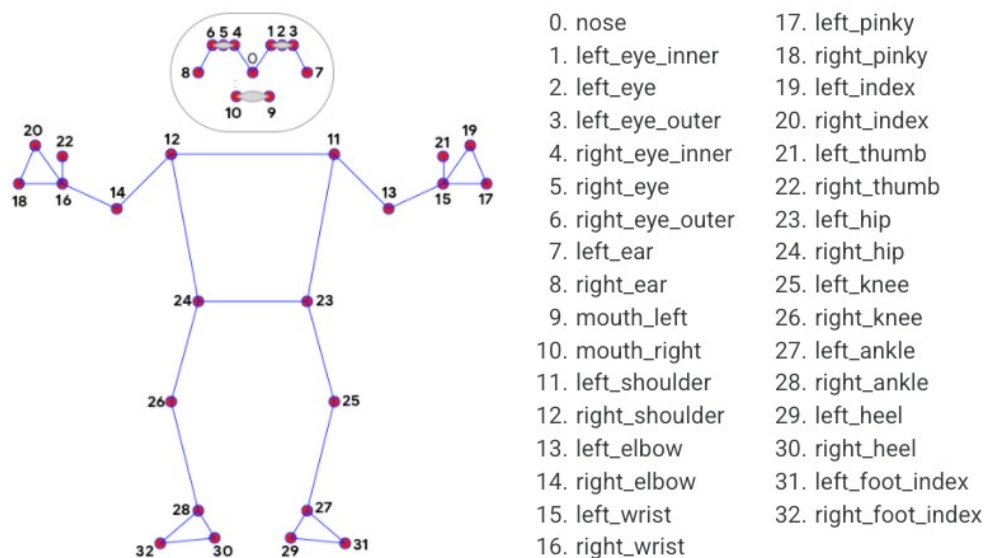


Рис. 1 Ключевые точки в библиотеке MediaPipe

Была реализована загрузка фотографии, применение библиотеки MediaPipe (Рис. 2), сохранение полученных координат в трехмерном пространстве для дальнейшего использования внутри программы Blender.

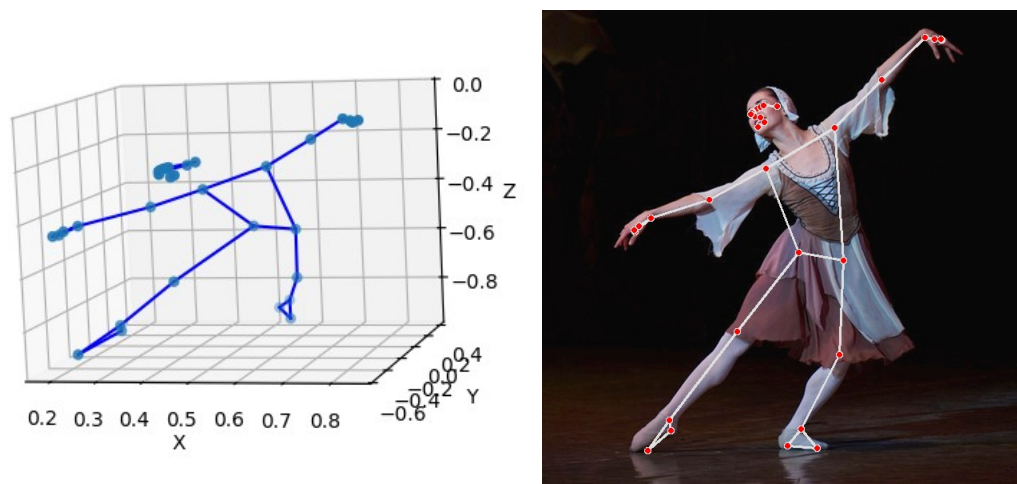


Рис. 2. Результат использования библиотеки MediaPipe

Для передачи позы 3D-модели было реализовано взаимодействие с Blender - популярным редактором трёхмерной графики. Приложение запускает Blender как отдельный подпроцесс и передаёт ему координаты костей, полученные из MediaPipe. Внутри Blender с помощью специального скрипта на Python происходит преобразование исходной 3D-модели: кости скелета перемещаются в соответствии с рассчитанными координатами, что приводит к деформации модели под нужную позу.

После применения позы скрипт автоматически сохраняет результат в формате OBJ. Это универсальный формат, который поддерживается большинством графических редакторов и позволяет легко загружать обработанную модель обратно в основное приложение для дальнейшей работы: наложения слоёв для рисования и цифровой живописи.

Дальнейшая разработка включает в себя коррекцию скелета 3-D модели, восстановление по полученным координатам дополнительных точек, соответствующих дополнительным костям в модели, расчет угла поворота для каждой кости, а также полная разработка интерфейса и функционала приложения.

### **Вывод**

В статье рассмотрены современные технологии определения трехмерной позы человека по фотографии и их применение для деформации 3D-моделей. Анализ существующих методов показал, что нейросетевые решения, такие как MediaPipe, обеспечивают достаточную точность и скорость работы, что делает их удобными для интеграции в интерактивные приложения. Предложенный подход с использованием MediaPipe для извлечения позы и передачи данных в Blender через Python-скрипты позволяет эффективно деформировать 3D-модель и сохранять результат в универсальном формате OBJ. Это открывает широкие возможности для создания десктопных приложений, ориентированных на диджитал-художников.





## **Рецензия на статью «Определение трехмерной позы человека по фотографии и её применение к 3D-модели» Е. А. Колесниковой**

### **Резюме**

Статья посвящена актуальной задаче автоматизированного определения трехмерной позы человека на основе фотографии с последующим использованием результатов для анимации 3D-моделей. В работе представлен подробный обзор современных методов решения задачи, включая нейросетевые подходы OpenPose, AlphaPose, DeepCut, DeeperCut и MediaPipe. Автором предложена и подробно описана практическая реализация подхода с использованием библиотеки MediaPipe и программы Blender, позволяющая быстро и точно перенести распознанную позу на 3D-модель. Подчеркиваются преимущества данного подхода в скорости и доступности для широкого круга пользователей, в частности digital-художников. Отмечена перспективность применения технологии в таких сферах, как фитнес, реабилитация, игровые приложения и творческие проекты.

Однако, наряду с сильными сторонами статьи, имеются моменты, которые требуют более глубокого анализа и доработки:

### **Критические замечания:**

1. В работе недостаточно подробно раскрыты причины предпочтения автором библиотеки MediaPipe перед другими аналогичными решениями. Важно привести сравнительный анализ точности, скорости работы и функциональных возможностей этих нейросетевых подходов.
2. Автор не приводит подробных данных о методологии оценки точности используемой нейросети. Следует явно указать используемые метрики (например, средняя ошибка определения координат ключевых точек) и предоставить результаты количественных испытаний на достаточной выборке.

3. В разделе, посвящённом реализации приложения, необходимо более подробно описать технические аспекты интеграции с Blender. Необходимо прояснить детали механизма передачи координат ключевых точек, специфику используемого формата данных и внутреннюю работу Python-скриптов в Blender.
4. Отсутствует полноценное экспериментальное тестирование предложенного подхода на фотографиях с различными условиями освещения, фоном и сложностью поз. Такой анализ существенно повысил бы доказательность и практическую ценность работы.
5. Недостаточно освещены потенциальные ограничения и проблемы предлагаемого метода, такие как сложность работы с изображениями, где есть частичное перекрытие частей тела, или позы, которые выходят за рамки привычных сценариев.

**Мелкие правки:**

- Рекомендуется уточнить аннотацию, более четко сформулировав цель исследования и достигнутые результаты.
- Функционал приложения для digital-художников лучше представить поэтапно и подробно: загрузка изображения, процесс определения позы, передача данных в Blender и последующая обработка.

**Рекомендация редактору:** Статья имеет несомненную научную и практическую ценность и может быть рекомендована к публикации после внесения указанных замечаний и доработок. При этом автор вправе опубликовать работу и в текущем варианте, а настоящая рецензия будет служить началом конструктивного обсуждения и дальнейшего развития темы.



УДК

## **Схема стеганографического встраивания данных в пространственную область изображения с использованием пороговых методов разделения данных**

**Н.С.Могилевская, О.В.Терещенко**

Южный федеральный университет, г. Ростов-на-Дону, Российская Федерация

### **Аннотация**

Так как большой проблемой стеганографических методов является неустойчивость встроенных данных к модификациям контейнера, актуальной задачей является модификация этих методов с целью повышения стойкости встроенной информации к повреждениям стегоконтейнера. Рассмотрены криптографические методы Шамира и Асмута-Блума порогового разделения данных. Рассмотрен стеганографический метод наименее значащего бита(LSB). Разработаны алгоритмы встраивания сообщения в контейнер и извлечения сообщения из заполненного контейнера. Программно реализована система, объединяющая в себе методы разделения секрета, алгоритм вычисления контрольной суммы и метод наименее значащего бита. Проведен сравнительный анализ криптографических методов Шамира и Асмута-Блума порогового разделения данных и выработаны рекомендации по использованию. Экспериментально исследована стойкость встроенных сообщений к различным искажениям заполненных контейнеров (для разработанной стеганографической схемы и метода LSB).

**Ключевые слова:** пороговые схемы разделения секрета, схема Шамира, схема Асмута-Блума, метод наименее значащего бита, стеганография.

**Для цитирования.** Н.С.Могилевская, О.В.Терещенко. Имитационная модель стеганографической системы, основанная на применении пороговых методов разделения данных. *Молодой исследователь Дона*. 2023;7(4):00–00.

### **Steganographic simulation model based on threshold data separation methods**

**Nadezhda S Mogilevskaya, Oleg V Tereshchenko**

Southern Federal University, Rostov-on-Don, Russian Federation

### **Abstract**

Since a big problem of steganographic methods is the instability of embedded data to container modifications, the actual task is to modify these methods in order to increase the resistance of embedded information to stego-container corruptions. Shamir and Asmuth-Bloom cryptographic methods of threshold data separation are considered. The steganographic method of the least significant bit is considered. Algorithms for embedding a message in a container and extracting a message from a filled container are developed. A system combining the methods of secret sharing, checksum calculation algorithm and least significant bit method is programmatically realized. A comparative analysis of the

Shamir and Asmuth-Bloom cryptographic methods of threshold data separation is carried out and recommendations for use are made. The resistance of embedded messages to various distortions of filled containers (for the developed steganographic scheme and LSB method) has been experimentally investigated.

**Keywords:** threshold-based secret sharing schemes, Shamir scheme, Asmuth-Bloom scheme, least significant bit method, steganography.

**For Citation.** N.S.Mogilevskaya, O.V.Tereshchenko. Simulation model of steganographic system based on application of threshold methods of data separation. *Young Researcher of Don*. 2023;7(4):00-00.

**Введение.** Методы информационной безопасности активно используют стеганографию. Помимо традиционного сокрытия информации, стеганографические методы (стеганометоды) успешно применяются для предотвращения неавторизованного копирования, контроля распространения данных в коммуникационных сетях и организации поиска в базах мультимедийных данных [1, 2].

Несмотря на все достоинства стеганографических методов, им присущи некоторые слабости и ограничения. Во-первых, факт применения стеганометодов может быть обнаружен, например, с помощью статистического анализа или сравнения заполненного контейнера с оригиналом. Заметим, однако, что анализ на наличие встроенных сообщений может требовать значительных вычислительных ресурсов. Во-вторых, объем встраиваемых данных и качество встраивания зависят от используемого контейнера. В-третьих, если контейнер случайно или намеренно изменен, то это чаще всего приводит к полной потере встроенного сообщения.

В настоящей работе исследуется возможность защиты встроенных данных от модификации заполненного контейнера. Для организации такой защиты используются методы порогового разделения данных, разрабатываемых в теории криптографических протоколов [3]. Основной принцип работы таких методов заключается в следующем: 1) исходные данные разделяются на  $n$  частей, 2) для восстановления секрета необходимо собрать не менее  $k$  частей (где  $k \leq n$ ), 3) схема обозначается как  $(k, n)$ -пороговая, 4) при наличии  $t \in [1, k-1]$  частей однозначно восстановить исходное сообщение невозможно.

Идея применения порогового разделения данных для повышения устойчивости встроенного сообщения к модификации контейнера обсуждается, например, в работах [4, 5]. В работе [4] обсуждается встраивание данных в частотную область мультимедийных контейнеров с применением стеганографических методов S-LSB [4] и Color [4]. В работе [5] используется встраивание в пространственную область изображения методом НЗБ [2]. В обеих работах использована схема Шамира для порогового разделения данных. К сожалению, в работе [5] схема описана недостаточно подробно, а построенное автором программное средство, реализующее схему недоступно.

Цель работы состоит в проектировании схемы стеганографического встраивания данных в пространственную область изображения с использованием пороговых методов разделения данных; а также программной реализации такой схемы и ее исследовании.

В разделе «Схема встраивания данных» описан предлагаемый стеганографический метод встраивания данных в файл-контейнер. Раздел «Схема извлечения данных» посвящен методу извлечения данных из заполненного и, возможно, поврежденного файла-контейнера. Раздел

«Схема восстановления секрета» описывает метод восстановления секрета из долей, извлеченных из контейнера.

### Схема встраивания данных.

В схеме встраивания данных применяются  $(t, n)$  – пороговые методы разделения данных [7, 8, 9, 10, 11].

Схема встраивания данных получает в качестве входных параметров метод разделения секрета (доступны 2 метода: схема Шамира и схема Асмута-Блума) и его параметры (числа  $n$  и  $t$ ), изображение-контейнер, параметры  $n$  и  $t$  порогового метода разделения данных и секрет.

В ходе первого этапа встраивания данных в контейнер происходит преобразование секрета: поочередно каждый символ заданного секрета заменяется на своё кодовое число, согласно кодировке Windows-1251. Таким образом, символьный секрет преобразуется в числовой. Затем происходит формирование  $n$  долей, согласно выбранному пороговому методу разделения данных и параметрам  $n$  и  $t$ . В процессе передачи файла от отправителя к получателю, стегоконтейнер может быть поврежден. Таким образом, доли, извлекаемые из контейнера, тоже могут быть поврежденными. При извлечении долей из файла-контейнера хотелось бы точно знать, какие доли являются целыми. Ввиду этой проблемы для контроля целостности долей вычисляются контрольные суммы (CRC) [6], в процессе вычисления CRC-кодов используется многочлен  $x^8 + x^5 + x^4 + 1$  [6]. Для каждой доли секрета вычисляется свой 8-битный CRC-код. Далее происходит встраивание долей секрета и их соответствующих CRC-кодов в стегоконтейнер. Изображение-контейнер условно разбивается на  $n$  вертикальных полос. Затем каждая доля секрета встраивается в соответствующую полосу по следующему принципу: в полосе изображения-контейнера выбирается случайный пиксель, начиная с которого будет встраиваться доля; сначала встраивается 27-битная сигнатура (для всех долей секрета сигнатура одинаковая), затем встраивается 8-битное число – длина доли, после, встраивается сама доля, и, наконец, встраивается CRC-код соответствующей доли. Конечно, случайная позиция в полосе изображения подбирается таким образом, чтобы встраиваемые данные полностью «поместились» в полосе. Схема встраивания информации в контейнер представлена на рис. 1. В качестве выходных параметров схема встраивания данных выдаст заполненный файл-контейнер и параметр  $P$ , вычисленный в соответствии с выбранной схемой разделения секрета.

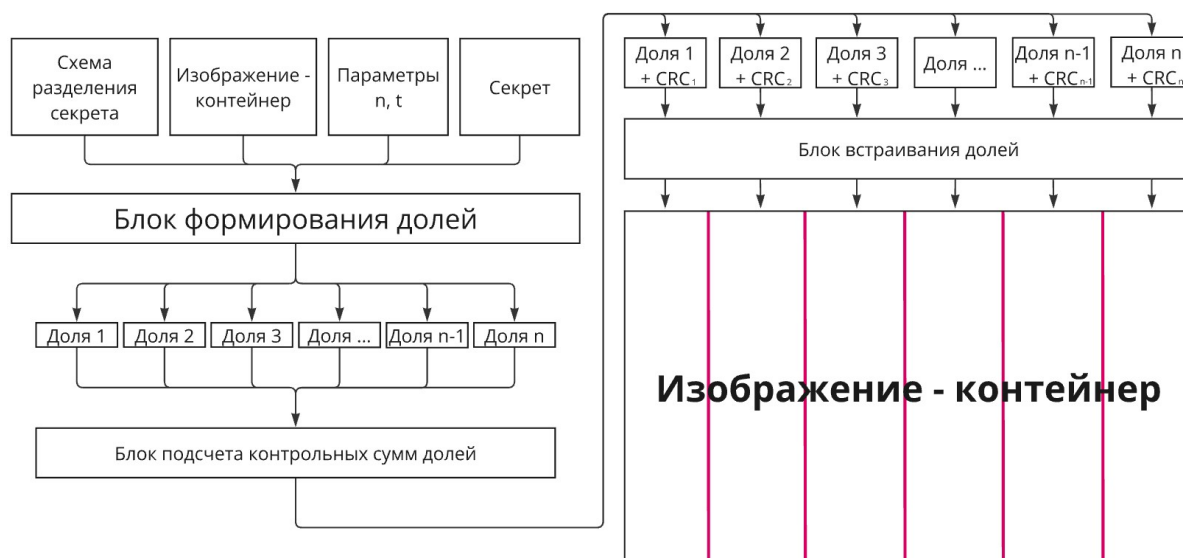


Рис. 1. Схема формирования и встраивания долей секрета в файл-контейнер

#### Схема извлечения данных.

Схема извлечения данных получает в качестве входных параметров изображение-контейнер и параметр  $n$  порогового метода разделения данных (также  $n$  – число вертикальных полос, на которое было разбито изображение контейнер в процессе встраивания долей секрета в файл).

В ходе первого этапа схемы извлечения данных из контейнера происходит извлечение долей секрета из файла-контейнера. Итеративно в каждой из полос изображения происходит считывание последних бит каждой цветовой компоненты каждого пикселя, пока не будет считана 27-битная сигнатура. После считывания сигнатуры происходит считывание первых 8 бит, идущих сразу за ней. Эти 8 бит указывают длину доли и ее CRC-кода (в битах). То есть, по сути, это число указывает, сколько далее необходимо считать бит данных. Происходит считывание доли и ее CRC-кода. Затем, считав все доли, в ходе второго этапа схемы извлечения данных, итеративно у каждой доли отделяются последние 8 бит – ее CRC-код. Затем для каждой доли итеративно происходит перерасчет CRC-кода. Новый, только что рассчитанный CRC-код сравнивается с извлеченным. Если коды различные, то рядом с извлеченной долей секрета будет указан специальный флаг «BREAK», иначе будет указан флаг «OK». Схема извлечения данных представлена на рис. 2. В качестве выходных параметров схема извлечения данных выдаст  $n$  долей секрета с соответствующими флагами.

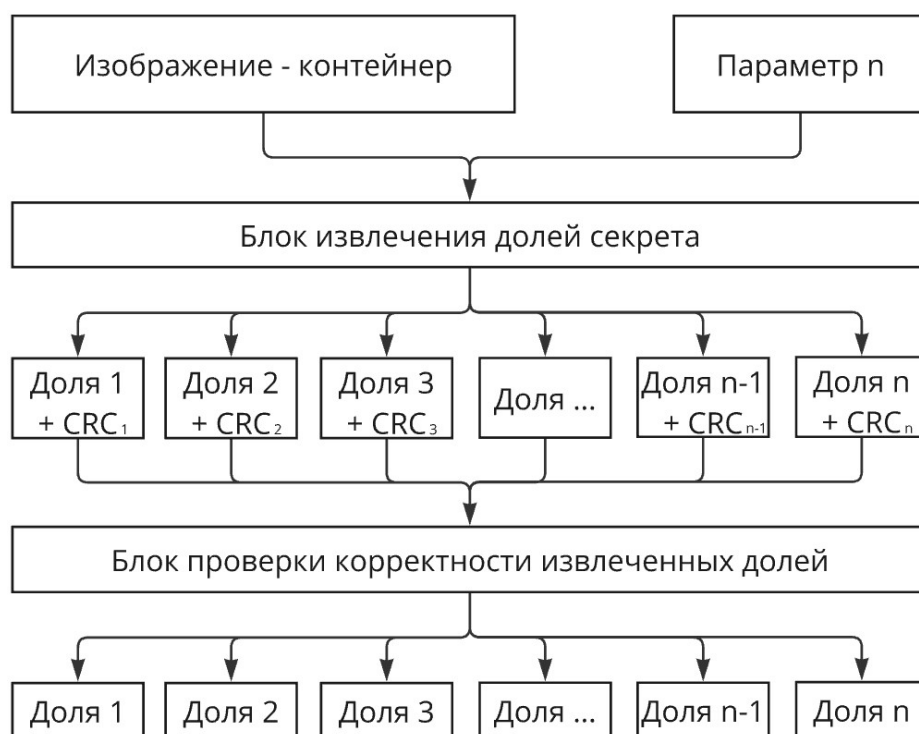


Рис. 2. Схема извлечения долей

**Схема восстановления секрета.** Схема восстановления секрета получает в качестве входных параметров метод разделения секрета (доступны 2 метода: схема Шамира и схема Асмута-Блума), параметр  $P$ , минимум  $t$  корректных долей секрета.

В ходе первого и единственного этапа схемы восстановления секрета согласно выбранному пороговому методу разделения данных происходит восстановление секрета. Схема

восстановления секрета представлена на рис. 3. В качестве выходных параметров схема восстановления данных выдаст секрет.

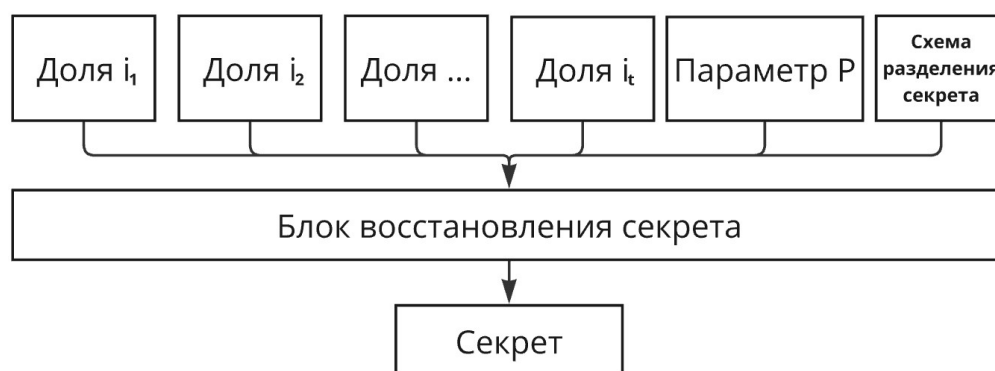


Рис. 3. Схема восстановления секрета

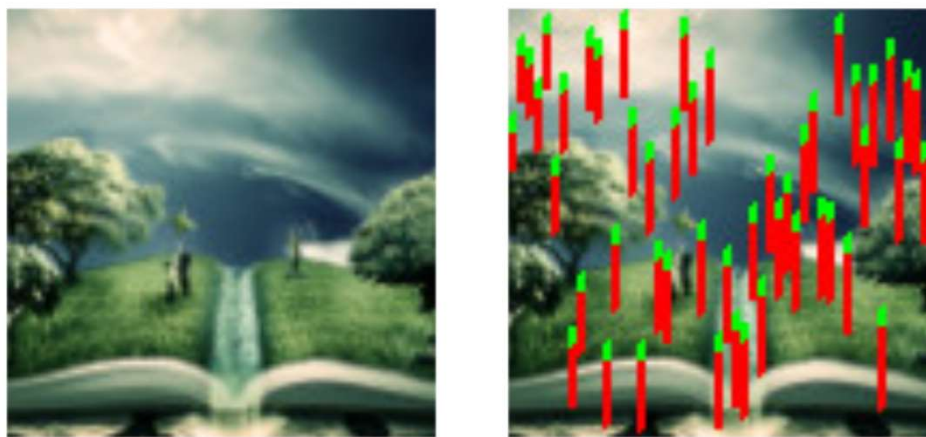
Принципиальное отличие реализованной стеганографической системы и схемы, описанной в [5], заключается в том, что в схеме, запрограммированной мною, доли секрета встраиваются в случайные позиции соответствующих полос стегоконтейнера. Эта особенность позволяет обеспечивать устойчивость данных, встроенных в файл, к обрезанию сверху.

**О программной реализации.** Построенная стеганографическая схема программно реализована на языке программирования C++/CLI с использованием среды разработки Microsoft Visual Studio 2019. Для обработки изображений выбрана библиотека мультимедийных технологий SFML [12]. Также была использована библиотека для работы с большими числовыми типами данных [13], так как промежуточные результаты вычислений превышают размер базовых типов данных языка C++.

В своей работе я реализовал 2 схемы: схему Шамира и схему Асмута-Блума. Схема Шамира основана на интерполяционных полиномах Лагранжа, а схема Асмута-Блума – на китайской теореме об остатках. Доли, формируемые схемой Шамира, обладают меньшей избыточностью в сравнении с долями, формируемыми по схеме Асмута-Блума. Эту особенность нужно учитывать, так как нам важно минимизировать изменения изображения-контейнера. Также важно отметить, что схема Асмута-Блума вычислительно сложнее.

**Пример формирования и встраивания долей согласно схеме.** На рисунке 4-А представлено изображение-контейнер. Формат изображения – png, размер изображения – 100×100 пикселей. В этот стегоконтейнер встроен секрет размером 35 байт. Параметры схемы разделения секрета  $n = 50$ ,  $t = 12$ . Так как для человеческого глаза незаметна разница в цвете двух пикселей, числовые значения цветовых компонент которых отличаются на единицу, то для того чтобы понять визуально, в какие пиксели были встроены биты данных, на рисунке 4-Б представлена та же картинка, что на рисунке 4-А. Однако пиксели, в которые были встроены доли секрета, помечены цветом. Зеленым цветом помечены сигнатуры, а красным – доли секрета.





А)

Б)

Рис. 4. Пример заполненного стегоконтейнера

**Примеры.** На рисунке 5 представлен пример применения стеганографической схемы и модификация заполненного стегоконтейнера путем наложения водяного знака. Размер изображения 160×160 пикселей. Размер секрета равен 80 байт. Параметры  $n$  и  $t$  равны 50 и 5 соответственно. После наложения водяного знака удалось корректно восстановить 9 долей из 50. Так как параметр  $t = 5$ , удалось корректно восстановить секрет.



Оригинальное

Заполненный

Наложение  
водяного знака

Поврежденный  
стегоконтейнер

Рис. 5. Пример

На рисунке 6 представлен пример применения стеганографической схемы и модификация заполненного стегоконтейнера путем обрезки изображения сверху и снизу. Размер оригинального изображения 77×77 пикселей. Размер обрезанного изображения – 77×37 пикселей. Размер секрета – 95 байт. Параметры  $n$  и  $t$  равны 40 и 3 соответственно. После «повреждения» стегоконтейнера удалось корректно восстановить 10 долей из 40. Так как параметр  $t = 3$ , удалось корректно восстановить секрет.



Оригинальное

Заполненный

Удаление части  
изображения

Поврежденный  
стегоконтейнер

Рис. 6. Пример

На рисунке 7 представлен пример применения стеганографической схемы и модификация заполненного стегоконтейнера путем осветления. Размер изображения 100×100 пикселей. Размер секрета – 90 байт. Параметры  $n$  и  $t$  равны 40 и 7 соответственно. После осветления изображения удалось корректно восстановить 10 долей из 40. Так как параметр  $t = 7$ , удалось корректно восстановить секрет.



Однако стоит отметить, что схема неэффективна в случае наложения фильтров, шумов. ... если изменения затрагивают все блоки файла-контейнера.

**Экспериментальное исследование.** Нам интересно, в какой зависимости находятся параметры  $n$ ,  $t$ , размер изображения и количество корректно восстановленных долей. Для установления зависимости перечисленных параметров был проведен ряд тестов. Во всех испытаниях секрет был фиксированного размера – 90 байт. Также во всех тестах был использован одинаковый метод модификации заполненного контейнера – обрезка изображения на 50%: 25% – снизу и 25% – сверху. Схема была протестирована на изображениях 3х масштабов: 100×100, 160×160, 300×300 пикселей. Для каждого изображения было проведено по 9 тестов. По три теста на каждое из трех значений параметра  $n$ : 25, 35, 50. В каждом из этих трех тестов параметр  $t$  принимал одно значение: 7, 12 или 17. Исходя из результатов опыта, можно заключить, что при фиксированном размере секрета шанс восстановления его будет выше, если размер изображения-контейнера и параметр  $n$  будут как можно больше, а параметр  $t$  как можно меньше.

Для метода наименее значащего бита (LSB) был проведен такой же тест: изображения 3х масштабов: 100×100, 160×160, 300×300 пикселей с встроенным секретом, размером 90 байт, были обрезаны на 50%: 25% – снизу и 25% – сверху. В итоге секрет не удалось восстановить корректно. Метод LSB крайне не устойчив к обрезке контейнера.

Был проведен эксперимент для установления зависимости суммарного размера долей, формируемых обеими схемами, от размера секрета и параметров  $n$  и  $t$ . Значение параметра  $n$  было зафиксировано и равно 12. Значение параметра  $t$  менялось от 2 до 11. Эксперимент был проведен для 4 различных встраиваемых сообщений размерами 60, 70, 80 и 90 байт. В ходе опыта было сделано следующее заключение: суммарный размер долей, формируемых по схеме Шамира, возрастает при росте параметра  $t$ , однако при увеличении размера секрета суммарный размер долей в среднем остается примерно одинаковым. В свою очередь, суммарный размер долей, формируемых по схеме Асмута-Блума, при росте параметра  $t$  остается примерно одинаковым, однако при росте размера секрета, суммарный размер долей увеличивается. Исходя из результатов эксперимента, легко заметить, что наиболее эффективной схемой разделения секрета является схема Шамира. Также стоит отметить, что процесс формирования долей по схеме Шамира занимает меньше времени.

Конечно, информация, встраиваемая по схеме LSB, занимает намного меньше места в сравнении с информацией, встраиваемой согласно разработанной схеме (пункт «схема встраивания данных»). Однако стоит отметить, что данные, встраиваемые согласно разработанной схеме более устойчивы к ряду модификаций стегоконтейнера: обрезка, осветление, наложение водяных знаков.

**Стегоанализ.** Человеческому глазу незаметна разница между оригинальным изображением и изображением, в которое был встроен некоторый секрет. Существует раздел стеганографии, задача которого состоит в выявлении факта наличия встроенной информации в файл и извлечении встроенного секрета из файла. Таким образом, помимо глобальной задачи встраивания секрета в стегоконтейнер, схема должна выполнять еще одну очень важную задачу, а именно – быть устойчивой к стегоанализу.

Для тестирования устойчивости данных, встраиваемых в файл-контейнер, к стегоанализу было выбрано программное средство [14]. Стегоанализ смог определить факт того, что изображение содержит в себе скрытые данные, однако программное средство [14] не смогло точно определить количество модифицированных байт файла. Также был проведен стегоанализ для изображения, в которое данные были встроены согласно методу LSB. Программное средство [14] также определило факт наличия встроенной информации.

Алгоритм оценки [14] «основан на интеллектуальной и тщательно протестированной комбинации уже существующих методов стегоанализа на основе пикселей, включая пары выборок Думитреску (2003), RS-анализ Фридриха (2001), атаку Хи-квадрат Вестфельда (2000) и первичные наборы Думитреску (2002)».

**Дальнейшие исследования.** Информация, встраиваемая согласно реализованной стеганографической системе, неустойчива к ряду модификаций контейнера. Например, в случае обрезки изображения одновременно по вертикали и по горизонтали восстановить секрет корректно не удастся. Аналогично при наложении фильтров и шума на изображение-контейнер корректное восстановление секрета кажется невозможным. Также важно сделать стеганографическую схему более устойчивой к стегоанализу.

Исходя из обозначенных проблем, был составлен план модификаций: во-первых, необходимо поменять метод встраивания бит данных в пиксели, это поможет сделать встраиваемые данные устойчивее к стегоанализу. Планируется встраивать биты данных не в каждую цветовую компоненту пикселя, а согласно некоторому алгоритму выбирать компоненту для встраивания. Однако также необходимо реализовать выбор случайных пикселей для встраивания бит данных.

Во-вторых, необходимо реализовать встраивание бит данных в «блоки» случайного размера [15]. Это поможет сделать встраиваемые данные устойчивее к обрезке изображения по вертикали и по горизонтали одновременно.

**Заключение.** Исходя из результатов проведенных экспериментов, можно заключить следующее: разработанная схема определенно эффективнее, чем классический метод LSB. Данные, встроенные по разработанной схеме устойчивы к ряду модификаций стегоконтейнера: наложение водяного знака, обрезке изображения сверху и снизу. Однако, схема неустойчива для таких модификаций как наложение фильтров и шума.

Для того чтобы шанс успешного восстановления секрета был наибольшим, нужно постараться свести к минимуму процент модифицируемых пикселей стегоконтейнера путем

выбора крупного изображения. Параметр  $n$  порогового метода разделения данных рекомендуется взять побольше, а параметр  $t$  – поменьше.

#### Список литературы

1. Грибунин, В. Г. Цифровая стеганография / В. Г. Грибунин, И. Н. Оков, И. В. Туринцев. — Цифровая стеганография, 2023-05-25. — Электрон. дан. (1 файл). — Москва: СОЛОН-Пресс, 2018. — 262 с.
2. Шелухин, Олег Иванович (1952-). Стеганография. Алгоритмы и программная реализация : = Алгоритмы и программная реализация учебное пособие для студентов высших учебных заведений, обучающихся по направлению подготовки 11.03.02, 11.04.02 - "Инфокоммуникационные технологии и системы связи" квалификации (степени) "бакалавр" и "магистр" / О. И. Шелухин, С. Д. Канаев ; под редакцией О. И. Шелухина. — Москва : Горячая линия - Телеком, 2018. — 592 с. : ил., табл. : 25 см.; ISBN 978-5-9912-0579-5
3. С. М. Рацеев. Криптографические протоколы. Схемы разделения секрета. Учебное пособие для вузов. Год: 2024. Страниц: 276. ISBN:978-5-507-50079-6
4. Т.И. Вишневская, Н.В. Уточкина. Стеганографический метод, устойчивый к повреждению данных. [Электронный ресурс] / Т.И. Вишневская, Н.В. Уточкина // Математические структуры и моделирование. 2020. №1. — URL: <https://cyberleninka.ru/article/n/steganograficheskiy-metod-ustoychivyy-k-povrezhdeniyu-dannyh/viewer>
5. Мироненко А.Н. Метод применения  $(T, N)$  - пороговой схемы в стеганографии. [Электронный ресурс] / Мироненко А.Н. // Математические структуры и моделирование. 2018. № 2(46). — URL: <http://msm.omsu.ru/jrns/jrn46/Mironenko2.pdf>
6. Искусство помехоустойчивого кодирования. Методы, алгоритмы, применение Автор: Морелос-Сарагоса Р. Издательство: Техносфера. Год: 2005. Страниц: 319. ISBN: 5-94836-035-0
7. Шнайер Б. Прикладная криптография. Протоколы, алгоритмы, исходные тексты на языке Си.- М.: Издательство ТРИУМФ, 2003 - 816 с.
8. Могилевская Н. С. О применении порогового разделения данных для организации разделенной передачи на примере метода битовых масок // Инженерный вестник Дона. 2017. № 2 (45). С. 39.
9. Mogilevskaya N. S. Data Sharing Method with Error Protection of Shares, 2023 IEEE 17th International Conference on Application of Information and Communication Technologies (AICT), Baku, Azerbaijan, 2023, pp. 1–6, doi: 10.1109/AICT59525.2023.10313186. 7.

10. Назаров А. С. Разработка методов и алгоритмов построения отказоустойчивых, распределенных систем хранения данных на основе модулярной арифметики: дис. ... канд. техн. наук: 05.13.18. СевероКавказский федеральный университет, Ставрополь; 2019. 272 с.

11. Тормасов, А. Г. Модель распределенного хранения данных с регулируемой избыточностью / А. Г. Тормасов, М. А. Хасин, Ю. И. Пахомов // Исследовано в России : [сайт]. – 2001 – № 1–4 – С. 355–364. – URL: <http://zhurnal.ape.relarn.ru/articles/2001/035.pdf>

12. <https://www.sfml-dev.org/documentation/2.6.1/>

13. Cross-platform and cross-standard header-only arbitrary precision arithmetic library. URL: <https://github.com/arbitraryprecision/ap/tree/master>

14. <https://github.com/b3dk7/StegExpose>

15. Балтаев Р.Х., Лунегов И.В, алгоритм встраивания и извлечения информации в неподвижные цифровые изображения стойкий к пассивным стегоаналитическим атакам //Вопросы безопасности. 2016. № 6. Р. 24–35. DOI: <https://doi.org/10.7256/24097543.2016.6.21252>. Доступно по адресу: [http://e-notabene.ru/nb/article\\_21252.html](http://e-notabene.ru/nb/article_21252.html)

*Об авторах:*

**Надежда Сергеевна Могилевская**, кандидат технических наук, доцент, доцент кафедры алгебры и дискретной математики Института механики, математики и компьютерных наук южного федерального университета (344090, РФ, г. Ростов-на-Дону, ул. Мильчакова, 8а), [nmogilevskaya@sfedu.ru](mailto:nmogilevskaya@sfedu.ru)

**Терещенко Олег Витальевич**, магистрант кафедры алгебры и дискретной математики Южного федерального университета (344090, РФ, г. Ростов-на-Дону, ул. Мильчакова, 8а), [teresh@sfedu.ru](mailto:teresh@sfedu.ru), тел. 89281050149

*Конфликт интересов:* авторы заявляют об отсутствии конфликта интересов.

*Все авторы прочитали и одобрили окончательный вариант рукописи.*

*About the Authors:*

**Nadezhda Sergeevna Mogilevskaya**, PhD (technical sciences), associate professor, associate professor, algebra and discrete mathematics department, Institute of Mechanics, Mathematics and Computer Science, Southern Federal University (344090, Russian Federation, Rostov-on-Don, Milchakova str., 8a), [nmogilevskaya@sfedu.ru](mailto:nmogilevskaya@sfedu.ru)

**Tereshchenko Oleg Vitalyevich**, Master's student, Department of Algebra and Discrete Mathematics, Southern Federal University (344090, Russian Federation, Rostov-on-Don, Milchakova str., 8a), [teresh@sfedu.ru](mailto:teresh@sfedu.ru), tel.89281050149

*Conflict of interest statement:* the authors do not have any conflict of interest.

*All authors have read and approved the final manuscript.*

**P**

## езюме

В работе рассматривается усовершенствованный метод стеганографического встраивания данных в изображения с использованием пороговых схем разделения секрета (Шамира и Асмута-Блума). Основная цель — повысить устойчивость встроенной информации к модификациям файла-контейнера (например, обрезке, наложению водяных знаков, осветлению).

Эксперименты подтвердили, что предложенная система более устойчива по сравнению с классическим методом LSB. Были проведены испытания при различных типах повреждений изображения (обрезка, осветление, водяной знак), при которых секрет успешно восстанавливался при выполнении условий ( $t \leq$  количество корректных долей). Также было показано, что схема Шамира эффективнее по объему данных и скорости исполнения, чем схема Асмута-Блума.

Отдельное внимание уделено устойчивости к стегоанализу. Несмотря на то, что факт наличия скрытых данных был обнаружен, извлечь содержимое не удалось.

В разделе перспектив предлагается усилить устойчивость схемы к агрессивным модификациям и улучшить маскировку данных для повышения стойкости к стегоанализу.

Работа представляет собой качественное исследование применения пороговых криптографических схем в стеганографии. Теоретические изыскания подтверждены экспериментальными исследованиями и программной реализацией. В целом, проделанная работа соответствует необходимым требованиям по уровню проработки и по актуальности исследуемой темы, однако для публикации необходимо учесть представленные ниже замечания.

## Критические замечания

1. Результаты экспериментальных исследований желательно представить в виде таблиц, а не текстового перечисления. Не представлены полученные результаты экспериментов, только их интерпретация (*“Исходя из результатов опыта, можно заключить, что при фиксированном размере секрета шанс восстановления его будет выше, если <...>”*)

## Мелкие правки

- Выбрана нумерация источников в соответствии с порядком упоминания в статье, при этом источники [7,8,9,10] упоминаются раньше, чем источник [6]
- В источнике [6] сначала идёт название, потом фамилии авторов. В источниках [] инициалы указаны до фамилии, в источнике [2] вместо инициалов полное имя и отчество автора
- Опечатки: отсутствие пробелов (*“35 байт.Параметры”*, *“ни травны”*, *“77×77пикселей”*), грамматическая ошибка (*“крайне не устойчив”* вместо *“крайне неустойчив”*)
- Надписи “А)” и “Б)” находятся не под соответствующими рисунками (“А)” левее рисунка А, “Б)” под правым краем рисунка А)
- Не подписан рисунок 7

## Рекомендация редактору

Принять работу после внесения описанных выше правок

# Применение алгоритмов машинного обучения для анализа лицевых микровыражений по видео

М.В. Нежинский

*Южный федеральный университет, Ростов-на-Дону*

**Аннотация:** В статье рассматривается задача автоматического обнаружения и классификации микровыражений лица в видеоданных, представляющих собой кратковременные произвольные лицевые движения, отражающие скрытые эмоции. Представлен обзор существующих методов, включая детекцию лица, выделение ключевых точек и анализ временной динамики, а также предложен потенциальный подход к решению задачи. Метод включает этапы детекции и выравнивания лица с использованием модели YOLOv8 и афинных преобразований, выделения нормализованных ключевых точек с помощью MediaPipe, аугментации данных посредством кластеризации геометрических характеристик лиц с использованием UMAP, и классификации временных рядов с применением нейронной сети с архитектурой Transformer.

**Ключевые слова:** Микровыражения, компьютерное зрение, детекция лица, ключевые точки лица, аугментация данных, UMAP, кластеризация, нейронные сети с механизмом внимания, косинусная метрика, YOLOv8, MediaPipe, CASME II, SAMM, CelebA.

## Введение

Микровыражения — это произвольные движения лицевых мышц продолжительностью менее 0,5 секунды, которые могут указывать на скрытые эмоциональные состояния. Их автоматическое обнаружение и классификация в видеоданных имеют значительный потенциал для применения в психологии, системах безопасности, маркетинге и человеко-машинном взаимодействии.

Настоящая работа посвящена обзору существующих методов решения задачи обнаружения и классификации микровыражений, а также описанию потенциального подхода, объединяющего этапы детекции лица, выделения ключевых точек, аугментации данных и классификации с использованием современных методов компьютерного зрения и машинного обучения. Предложенный метод направлен на преодоление ограничений, связанных с малым объемом данных и вариативностью геометрии лиц.

## Обзор литературы

Современные методы обнаружения и классификации микровыражений включают несколько ключевых этапов: детекцию лица, выделение ключевых точек и анализ временной динамики лицевых движений. Для детекции лиц применяются модели глубокого обучения, такие как YOLO [1] и MTCNN [2], которые обеспечивают высокую точность и производительность в реальном времени.



Эти модели определяют область лица в видеопотоке, что является первым шагом для последующего анализа.

Выделение ключевых точек лица выполняется с использованием инструментов, таких как DLib [3] или MediaPipe, которые определяют стандартные наборы точек (например, 68 точек, соответствующих анатомическим ориентирам, включая глаза, брови, нос и рот). Ключевые точки служат основой для анализа движений, связанных с микровыражениями, так как их смещения во времени отражают динамику лицевых выражений.

Одной из основных проблем является ограниченный объем специализированных датасетов микровыражений, таких как CASME II и SAMM [4], которые содержат всего несколько сотен видеопоследовательностей. Это ограничивает возможности обучения моделей глубокого обучения. Для преодоления данного ограничения применяются методы аугментации данных, включая геометрические преобразования и генеративные модели, такие как генеративно-состязательные сети [5].

Классификация микровыражений требует обработки временных последовательностей, для чего используются модели, способные учитывать динамику данных. Рекуррентные нейронные сети (RNN), включая их разновидности, такие как LSTM, традиционно применялись для анализа временных рядов [6]. В последние годы архитектуры Transformer [7], основанные на механизме внимания, продемонстрировали высокую эффективность в задачах обработки последовательностей, включая анализ видеоданных и временных рядов. Эти модели способны улавливать долгосрочные зависимости в данных, что делает их перспективными для классификации микровыражений.

### **Предлагаемый метод**

Предлагаемый метод включает четыре основных этапа: детекция и выравнивание лица, выделение ключевых точек, аугментация данных и классификация микровыражений. Метод направлен на решение проблем ограниченного объема данных и вариативности геометрических характеристик лиц.

#### **1. Детекция и выравнивание лица**

На первом этапе выполняется детекция лица в видеопотоке с использованием модели YOLOv8 [9], которая обеспечивает высокую точность и производительность. После обнаружения лица применяется выравнивание для устранения различий в ориентации и масштабе. Выравнивание реализуется посредством аффинного преобразования, которое трансформирует изображение таким образом, чтобы линия, соединяющая центры глаз, была горизонтальной, а расстояние между глазами (IOD) составляло фиксированное значение. Аффинное преобразование применяется к полному изображению, после чего область лица определяется заново.

#### **2. Выделение ключевых точек**

Для обнаружения ключевых точек лица могут использоваться современные компьютерные модели, такие как MediaPipe Face Mesh, Dlib, 3DDFA, RetinaFace или OpenFace, способные определять от нескольких десятков до нескольких сотен характерных точек. Из этого набора обычно выбирают стандартный поднабор из 68 точек, соответствующих основным анатомическим ориентирам (глаза, брови, нос, рот). Полученные координаты нормализуются относительно межзрачкового расстояния (IOD), что унифицирует масштаб и сохраняет геометрические пропорции лица. Такая нормализация устраняет различия в масштабе и положении лица, сохраняя его геометрические пропорции, что важно для последующего анализа, аугментации данных или применения в задачах компьютерного зрения.

### **3. Аугментация данных**

Ограниченный объем датасетов микровыражений, таких как CASME II и SAMM, требует применения методов аугментации данных. Предлагается следующий подход для увеличения объема и разнообразия данных.

Для создания аугментированных данных может быть использован датасет CelebA [10], содержащий большое количество статических изображений лиц. Для каждого изображения в CelebA определяются ключевые точки, которые нормализуются по IOD. Нормализованные ключевые точки преобразуются с использованием метода UMAP [8] для снижения размерности до 10 измерений. В UMAP применяется косинусная метрика расстояния, которая обеспечивает устойчивость к небольшим вариациям нормализации и фокусируется на геометрических характеристиках лиц. Далее выполняется кластеризация, группирующая лица по схожим геометрическим признакам.

Для аугментации данных выбирается видеопоследовательность лица А из датасета микровыражений и изображение лица В из датасета CelebA. Для каждого кадра видео лица А вычисляются нормализованные ключевые точки, которые преобразуются через модель UMAP для определения ближайшего кластера в CelebA. Из этого кластера выбирается лицо В, после чего проводится сопоставление нормализованных ключевых точек кадра лица А и лица В с использованием косинусной метрики расстояния. Косинусная метрика выбрана для учета формы лица, минимизируя влияние небольших смещений координат.

Выбирается кадр лица А, где косинусное расстояние между его ключевыми точками и точками лица В минимально, что указывает на наибольшую схожесть геометрических характеристик. На основе ключевых точек этого кадра и лица В вычисляется перспективное отображение, которое трансформирует геометрию лица А в геометрию лица В. Данное отображение применяется ко всем кадрам видеопоследовательности лица А, генерируя аугментированный временной ряд ключевых точек, имитирующий микровыражение на геометрии лица В.

### **4. Классификация микровыражений**

Аугментированные данные представляются в виде временных рядов координат ключевых точек размером  $T \times 136$ , где  $T$  — число кадров, а 136 — координаты  $(x, y)$  для 68 точек. Эти ряды

предполагается использовать в качестве входных данных для нейронной сети с архитектурой Transformer [7]. Модель включает несколько слоев с механизмом внимания, что позволяет эффективно анализировать временные зависимости в последовательности. На выходе применяется полносвязный слой для классификации микровыражений, таких как счастье, гнев, удивление или отвращение.

### **Заключение**

Предложенный метод обладает рядом потенциальных преимуществ. Нормализация ключевых точек по расстоянию между глазами и использование косинусной метрики в UMAP и сопоставлении геометрий обеспечивают устойчивость к вариациям масштаба, ориентации и небольшим ошибкам выравнивания. Аугментация данных на основе кластеризации и перспективных отображений позволяет увеличить объем и разнообразие обучающих данных, покрывая широкий спектр геометрических характеристик лиц. Архитектура Transformer, благодаря механизму внимания, перспективна для обработки временных рядов ключевых точек, что может улучшить точность классификации микровыражений.

Тем не менее, метод имеет потенциальные ограничения. Качество аугментации зависит от точности моделей детекции лица и выделения ключевых точек, а также от выбора параметров кластеризации (например, числа кластеров). Кроме того, перспективные отображения предполагают, что геометрические преобразования способны адекватно передать динамику микровыражений, что требует дальнейшей проверки.

### **Литература**

6. Redmon, J., Farhadi, A. (2016). You Only Look Once: Unified, Real-Time Object Detection. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 779–788.
7. Zhang, K., Zhang, Z., Li, Z., Qiao, Y. (2016). Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks. IEEE Signal Processing Letters, 23(10), 1499–1503.
8. Kazemi, V., Sullivan, J. (2014). One Millisecond Face Alignment with an Ensemble of Regression Trees. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 1867–1874.
9. Davison, A. K., Lansley, C., Costen, N., Tan, K., Yap, M. H. (2018). SAMM: A Spontaneous Micro-Facial Movement Dataset. IEEE Transactions on Affective Computing, 9(1), 116–129.

10. Goodfellow, I., Pouget-Abadie, J., Mirza, M., et al. (2014). Generative Adversarial Nets. *Advances in Neural Information Processing Systems (NeurIPS)*, 2672–2680.
11. Hochreiter, S., Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, 9(8), 1735–1780.
12. Vaswani, A., Shazeer, N., Parmar, N., et al. (2017). Attention is All You Need. *Advances in Neural Information Processing Systems (NeurIPS)*, 5998–6008.
13. McInnes, L., Healy, J., Melville, J. (2018). UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. *arXiv preprint arXiv:1802.03426*.
14. Jocher, G., Stoken, A., Borovec, J., et al. (2023). YOLOv8 by Ultralytics. *GitHub Repository*.
15. Liu, Z., Luo, P., Wang, X., Tang, X. (2015). Deep Learning Face Attributes in the Wild. *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 3730–3738.

## Рецензия на обзорную статью

### «Применение алгоритмов машинного обучения для анализа лицевых микровыражений по видео»

М.В. Нежинский,

Южный федеральный университет, Ростов-на-Дону

#### Резюме

Статья посвящена современным подходам к автоматическому анализу микровыражений лица на видеоданных с применением методов машинного обучения и компьютерного зрения. Автор структурированно рассматривает основные этапы для решения данной задачи. Особое внимание уделено обзору актуальных технологий и инструментов, таких как YOLOv8, MediaPipe, UMAP, Transformer, а также приведены их преимущества и ограничения для анализа микровыражений.

#### Критические замечания

##### 1. Полнота и системность обзора методов

Автор последовательно описывает ключевые этапы необходимые для обработки входящих данных. Для каждого из приведенных этапов приведены актуальные инструменты и подходы, что демонстрирует хорошее владение предметной областью. Однако в ряде случаев обзор мог бы быть более критическим: например, при описании того или иного метода привести потенциальные трудности и риски, которые могут возникнуть при работе с данным методом или с выходным результатом конкретного этапа.

##### 2. Аргументация выбора технологий

В статье обоснован выбор каждого инструмента, подтверждаемый ссылками на современные исследования.

Однако, в то же время, не представлено краткое сравнения с другими методами, которые могли бы применяться, но имеют ряд недостатков в контексте микровыражений.

##### 3. Структурированность и доступность изложения

Материал изложен логично, с четким разделением на этапы и последовательным раскрытием каждого из них. Используемая специфичная терминология, термины поясняется по мере появления.

В то же время, донесение информации могло бы стать доступнее при включении схем, таблиц или промежуточных данных работы этапа, чтобы визуальнo структурировать приведенные методы и последовательность обработки.

#### Мелкие правки

- В разделе «Обзор литературы» рекомендуется добавить краткие характеристики используемых датасетов (CASME II, SAMM, CelebA) для полноты обзора: какие эмоции представлены, для какой группы людей и подобное.
- В некоторых случаях стоит уточнить, какие именно категории микровыражений рассматриваются (например, по классификации эмоций).

#### Рекомендация редактору

##### Принять с незначительными правками

Статья отвечает требованиям обзорной публикации: системно и современно освещает методы анализа микровыражений, корректно описывает применяемые технологии, аргументирует их выбор и указывает на потенциальные ограничения. Рекомендуется к публикации после внесения мелких редакционных правок, направленных на повышение

наглядности и полноты обзора.

Автор имеет право опубликовать данную статью в представленном виде, рецензент на исправлении замечаний не настаивает.

Рецензент:

Селина Мария Константиновна,

Южный федеральный университет, Ростов-на-Дону.

## **Прототипирование системы динамического саундтрека в компьютерной игре**

*Шишкина В. И.*

*Южный федеральный университет, Ростов-на-Дону*

**Аннотация:** В данной работе рассматривается проектирование и реализация прототипа динамического саундтрека для видеоигры жанра *top-down shooter*. Основное внимание уделяется методу вертикальной оркестровки и использованию среды FMOD Studio в сочетании с игровым движком Unity. Описаны этапы создания музыкального сопровождения, логика его адаптации к игровым состояниям и технические аспекты интеграции с игрой.

**Ключевые слова:** видеоигры, саундтрек, динамическая музыка, FMOD, Unity, вертикальная перестройка, интерактивный звук

### **Введение**

Звуковое сопровождение играет критически важную роль в формировании эмоционального и повествовательного опыта игрока. С переходом к более интерактивным форматам повествования возрастает значимость динамической музыки, способной адаптироваться к изменениям игрового состояния в реальном времени. Музыка становится не только фоновым элементом, но и активным участником игрового процесса, влияющим на восприятие и вовлечённость пользователя.

### **Методы создания динамического саундтрека**

Существует два основных метода создания динамичных музыкальных систем: горизонтальная перестройка последовательности и вертикальная перестройка оркестровки.

Horizontal Re-Sequencing — метод, при котором музыкальные фрагменты переключаются в зависимости от изменений в игровом процессе. Это позволяет создавать плавные переходы между различными состояниями игры, например, при переходе от исследования мира к боевым сценам.

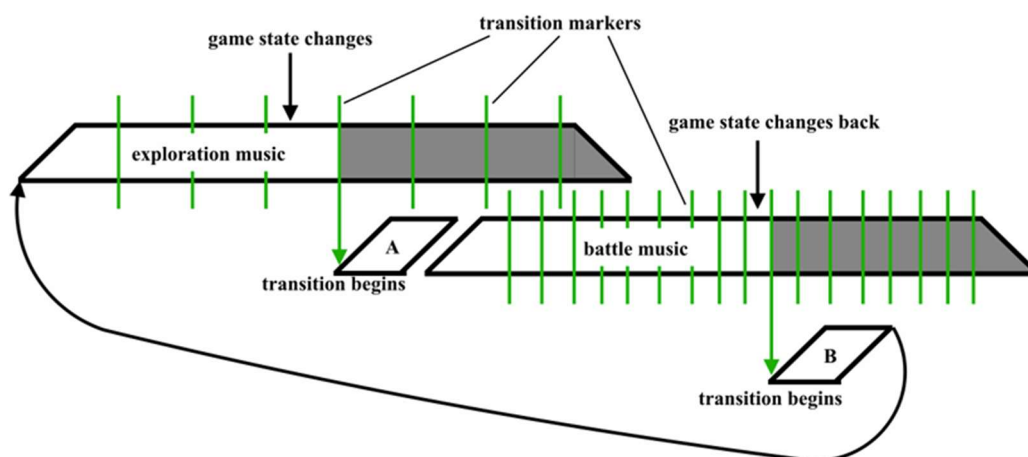


Рис. 1 – Горизонтальная перестройка последовательности

Vertical Re-Orchestration — технология, при которой изменяются музыкальные слои, регулируя интенсивность композиции в зависимости от событий в игре. Например, при увеличении количества врагов на экране могут добавляться дополнительные ударные инструменты или усиливаться динамика звучания.

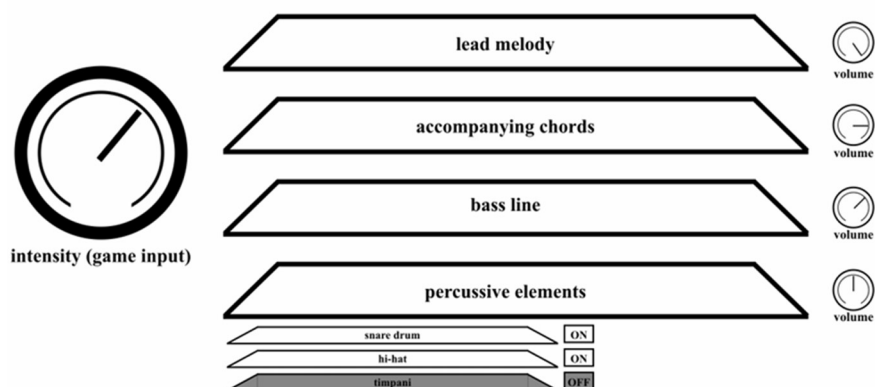


Рис. 2 – Вертикальная перестройка оркестровки

Сейчас динамический саундтрек используется в большинстве современных игр и зачастую эти два метода используются вместе, но наиболее яркими примерами могут быть Devil may cry 5, Doom и Doom: Eternal.

### **Постановка задачи**

Целью данной работы является создание прототипа динамического саундтрека для видеоигры, выполненной в жанре top-down shooter. С учётом геймдизайнерских особенностей и динамичного характера игрового процесса был выбран метод вертикальной оркестровки. Музыкальное сопровождение должно адаптироваться к четырём основным состояниям игры, обеспечивая целостность аудиовизуального восприятия. При изменении игровой ситуации будут управляться музыкальные слои: включаться или выключаться отдельные инструментальные партии и изменяться их громкость.

### **Музыкальные состояния и структура саундтрека**

Было реализовано четыре ступени развития саунд трека, каждая из которых соответствует определённому состоянию игры. Саундтрек написан в жанре синтвейв, поэтому основу музыкальной палитры составляют синтезаторы с характерным ретрофутуристическим тоном. Композиция построена в пентатонической гамме до минор (C minor pentatonic scale), что придаёт музыке мрачное, но ритмически выразительное настроение.

В начале прохождения уровня звучит интро, представляющее собой арпеджио, которое погружает игрока в игровое пространство и задает общее настроение сцены. При перемещении между локациями в композицию добавляется эмбиентный слой, затягивающий игрока далее в пространство уровня.

Когда начинается бой с одиночными врагами, вступает основная мелодия — яркий синтезаторный лид, акцентирующий напряжённость момента и вовлечённость игрока. Наконец, в случае сражения с множеством врагов, к композиции добавляются ударные и басовая партия, придавая музыке агрессивность и энергетический напор.



## **Использование FMOD Studio**

Саундтрек был написан в FL Studio откуда созданные музыкальные сэмплы были импортированы в FMOD Studio.

Fmod был выбран как основной инструмент для данной работы, благодаря его широким возможностям настройки интерактивного звука и интеграцией с Unity. Основной принцип работы FMOD строится на системе событий (Events), которая позволяет создавать сложные и адаптивные звуковые конструкции. В отличие от традиционного воспроизведения звуков через аудиофайлы, события FMOD могут содержать многослойные аудиотреки, параметры изменения звука в реальном времени и сложную логику поведения.

Каждое событие в FMOD состоит из аудиодорожек (Tracks), на которых располагаются звуковые файлы, маркеров (Markers), запускающих действия или переключающих логику воспроизведения, параметров (Parameters), позволяющих изменять звук в зависимости от внешних условий, микшера (Mixer) для управления громкостью и эффектами, а также логических дорожек (Logic Tracks), содержащих автоматизацию, задержки и другие элементы поведения.

### **Логика переключения музыкальных состояний**

Для органичного и плавного переключения саундтрека в нужное состояние был использован инструмент Transition region в FMOD Studio, который позволяет переключаться между музыкальными дорожками в определенные промежутки длиной в такт или несколько тактов. В данном проекте был выбран промежуток длиной в такт из-за быстрого геймплея данного игрового жанра. Переключение между состояниями реализовано следующим образом: при изменении специальной переменной, отвечающей за состояние саундтрека, система ожидает окончания текущего такта и переходит к нужному состоянию.

## **Автоматизация и циклы**

Для управления громкостью звучание отдельных музыкальных партий были использованы кривые автоматизации. С их помощью реализовано плавное убывание и возрастание громкости нужных сэмплов при переключении состояний. Таким образом предотвращена резкая остановка или включение музыкальной дорожки, которая бы могла нарушить вовлечение игрока.

Также для сохранения состояния саундтрека в определенной фазе, когда его переключение не требуется используется инструмент `loop region`, где семплы, находящиеся в его регионе, зацикливаются и играют пока не будет выполнен переход из региона.

## **Интеграция с Unity**

FMOD был интегрирован в игровое окружение Unity посредством соответствующего плагина, что позволило использовать события FMOD и управляющие переменные непосредственно в игровом процессе. Управление состояниями осуществляется посредством пользовательского интерфейса, реализующего переключение между событиями в реальном времени.

## **Заключение**

Разработанный прототип демонстрирует возможности реализации адаптивного саундтрека, учитывающего динамику игрового процесса и эмоциональную составляющую взаимодействия игрока с игрой. Использование FMOD и метода вертикальной оркестровки позволило обеспечить высокую степень гибкости музыкального сопровождения и его интеграции в игровые механики.

## **Литература**

11. Смит Л. Значение динамической музыки в видеоиграх URL: <https://www.diva-portal.org/smash/record.jsf?pid=diva2%3A1934330&dswid=-4973>

## **References**

1. Smith L. The Effect of Dynamic Music in Video Games URL:  
<https://www.diva-portal.org/smash/record.jsf?pid=diva2%3A1934330&dswid=-4973>

## **Рецензия на статью «Прототипирование системы динамического саундтрека в компьютерной игре»**

### **Резюме**

Статья посвящена созданию прототипа динамического саундтрека для видеоигры в жанре top-down shooter с использованием метода вертикальной оркестровки и FMOD Studio, интегрированного с Unity. Автор описывает процесс разработки музыкального сопровождения, его адаптацию к игровым состояниям и технические детали реализации. Работа демонстрирует практическое применение интерактивного звука, ссылаясь на примеры из игровой индустрии, такие как Devil May Cry 5 и Doom.

### **Сильные стороны**

1. Актуальность темы: статья освещает современный подход к созданию динамической музыки, что важно для игровой индустрии.
2. Логичная структура: текст чётко разделён на введение, методы, реализацию и заключение, что упрощает чтение.
3. Практическая ценность: использование FMOD Studio и Unity показывает реальный процесс интеграции звука в игру.
4. Иллюстративность: схемы горизонтальной и вертикальной перестройки (рис. 1 и рис. 2) наглядно поясняют методы.

### **Замечания**

1. Теоретическая часть: В разделе «Методы создания динамического саундтрека» можно было бы кратко сравнить горизонтальную и вертикальную перестройку, чтобы обосновать выбор последней.
2. Библиография: Указана только одна ссылка, что ограничивает научную глубину. Дополнение списком литературы усилило бы работу.
3. Оценка прототипа: В заключении отсутствует информация о тестировании (например, отзывы игроков). Добавление результатов тестирования сделало бы выводы убедительнее.

### **Мелкие правки**

- В разделе «Музыкальные состояния и структура саундтрека» поясните термин «пентатоническая гамма до минор» для неспециалистов.

- Исправьте «Fmod» на «FMOD» в разделе «Использование FMOD Studio» для единообразия.

### **Рекомендации редактору**

Статья готова к публикации благодаря актуальности, практической направленности и чёткой структуре. Предложенные замечания необязательны, так как работа уже самодостаточна. Рекомендую принять статью к публикации с возможностью добровольного учёта указанных правок.

# Архитектура компьютерной игры на Unity

*В.Г. Бурым*

*Южный федеральный университет, Ростов-на-Дону*

**Аннотация:** Разработка системы оповещения об игровых событиях для масштабного игрового проекта, соблюдающей архитектурные принципы для эффективного командного взаимодействия и построения гибкой, расширяемой архитектуры на платформе Unity.

**Ключевые слова:** архитектура, unity, dependency inversion, inversion of control, zenject, игровые события, пауза, компьютерная игра, геймдев, программирование.

## Введение

В современном программировании построение гибкого, интуитивно понятного и расширяемого кода является одной из ключевых задач для больших проектов. Разработчики часто сталкиваются с проблемами, связанными с неподдерживаемым кодом, что может привести к значительным задержкам в разработке и усложнению его понимания, процесса обновления и добавления нового контента. Это особенно важно для масштабных игровых проектов, в которых главными факторами являются эффективность командного взаимодействия и быстрая адаптация к новым гейм-дизайнерским решениям.

Одной из актуальных задач является проблема оповещения классов об определенных игровых событиях, таких как «Начало игры», «Пауза» и др. То есть при наступлении определенного события, некоторые классы должны выполнять соответствующий этому событию метод, при том, что он может выполняться как 1 раз, так и на каждом кадре игры. При этом для поддержания расширяемой архитектуры не должны нарушаться такие принципы как Single Responsibility Principle (далее SRP) и Open Closed Principle (далее OCP). Событий может быть разнообразное количество, но для примера в данной статье рассматривается событие «Паузы».

Исследование данной проблемы проводилось на платформе Unity с использованием технологий языка C# и плагина Zenject.

## Решение, которое предоставляет Unity

В Unity существует механизм, который позволяет объектам взаимодействовать друг с другом через события. С помощью него данная задача решается следующим образом (Рисунок 1):

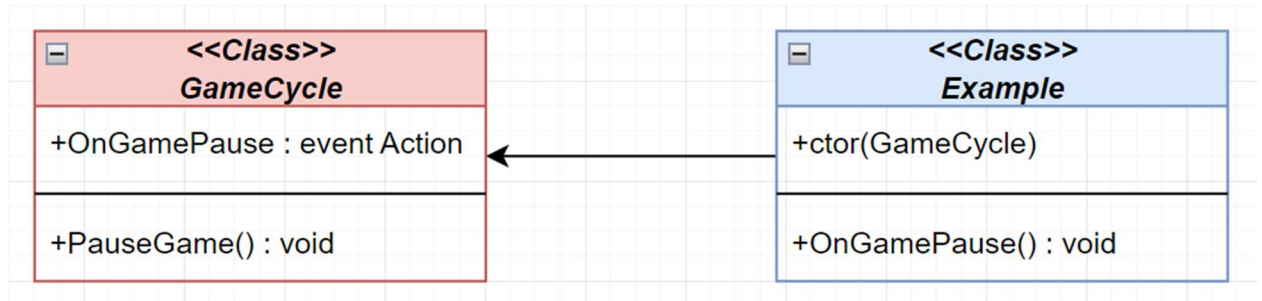


Рисунок 1. Решение задачи методами Unity.

Инфраструктурный класс `GameCycle` содержит событие `OnGamePause`, на которое подписывается класс `Example`, таким образом, что при вызове функции `PauseGame`, также выполнится метод `OnGamePause` у класса `Example`.

К преимуществам данного подхода можно отнести:

- простота реализации
- малые затраты по времени для небольших проектов

К недостаткам данного подхода можно отнести:

- сложность поддержки и расширения большого проекта
- трудности командного взаимодействия
- большие затраты по времени для масштабных проектов

Данный подход имеет ряд серьезных недостатков для разработки масштабного проекта, т.к. в данной системе `Example` нарушает принципы SRP и OCP, из-за чего при изменении или удалении класса `GameCycle`, разработчик вынужден тратить время на изменение всех классов `Example`.

## Менеджер игровых состояний

Решением поставленной задачи является применение принципа `Dependency Inversion Principle` (далее DIP) и следование такой концепции как `Inversion of Control` (далее IoC). В работе рассматривается применение интерфейсов C#, что является абстракцией в виде общей точки

взаимодействия, и далее в статье в красный оттенок будут окрашены высокоуровневые системы, в голубой низкоуровневые, а в желтый абстракции.

Были созданы такие контракты как `GamePauseListener` и `GameResumeListener`, которые имеют методы, выполняющиеся при паузе и продолжении игры соответственно. Также был разработан инфраструктурный класс `GameCycle`, который хранит классы, реализующие данные контракты и несет ответственность их оповещения о наступлении соответствующих событий (Рисунок 2).

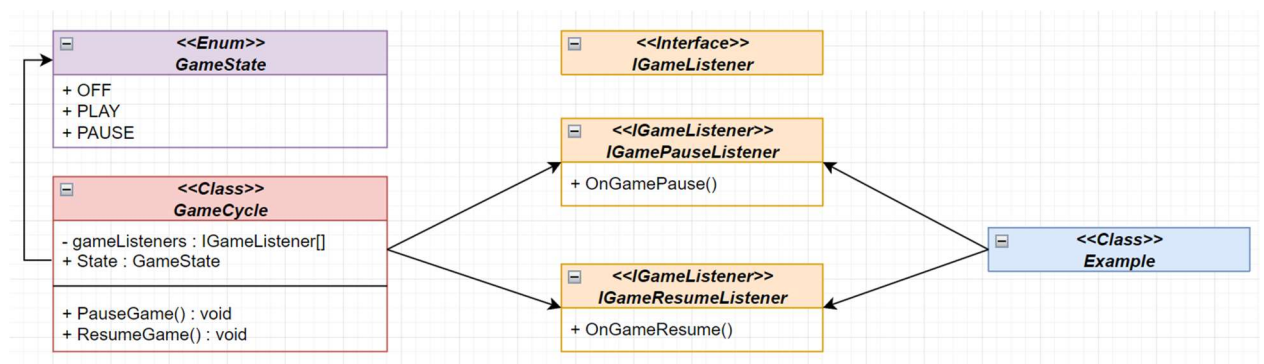


Рисунок 2. Менеджер игровых состояний.

Для того, чтобы классы не регистрировали сами себя был применен паттерн программирования `Dependency Injection` (далее `DI`). Его реализация взята из плагина `Zenject`. Это позволит внедрять классы, реализующие разработанные контракты, как зависимость в высокоуровневую систему `GameCycle` (Рисунок 3).



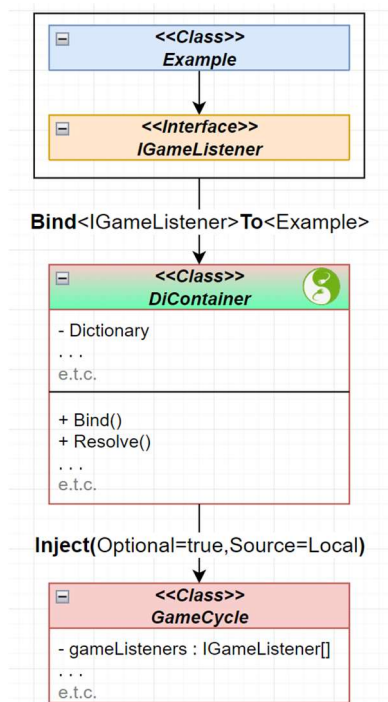


Рисунок 3. Внедрение зависимостей с помощью плагина Zenject.

### Методы, выполняющиеся на каждом кадре игры

Помимо того, что Zenject внедряет зависимости, он также является IoC контейнером и имеет множество контрактов, с помощью которых управляет жизненным циклом объектов. Рассмотрим один из таких контрактов - `ITickable`, позволяющий классам, которые его реализуют иметь метод, который выполняется на каждом кадре игры с помощью метода `Update`, без прямой связи самого класса с `Unity`.

Рассматриваемая в данной статье задача подразумевает, что класс может иметь метод, который выполняется на каждом кадре игры, но только при действии определенного события. В данном случае метод `Tick` класса, реализующего контракт `ITickable`, выполняется на каждом кадре игры независимо от текущего события системы `GameCycle`. Для решения задачи рассмотрим расширение плагина Zenject на основе реализации данного механизма (Рисунок 4).

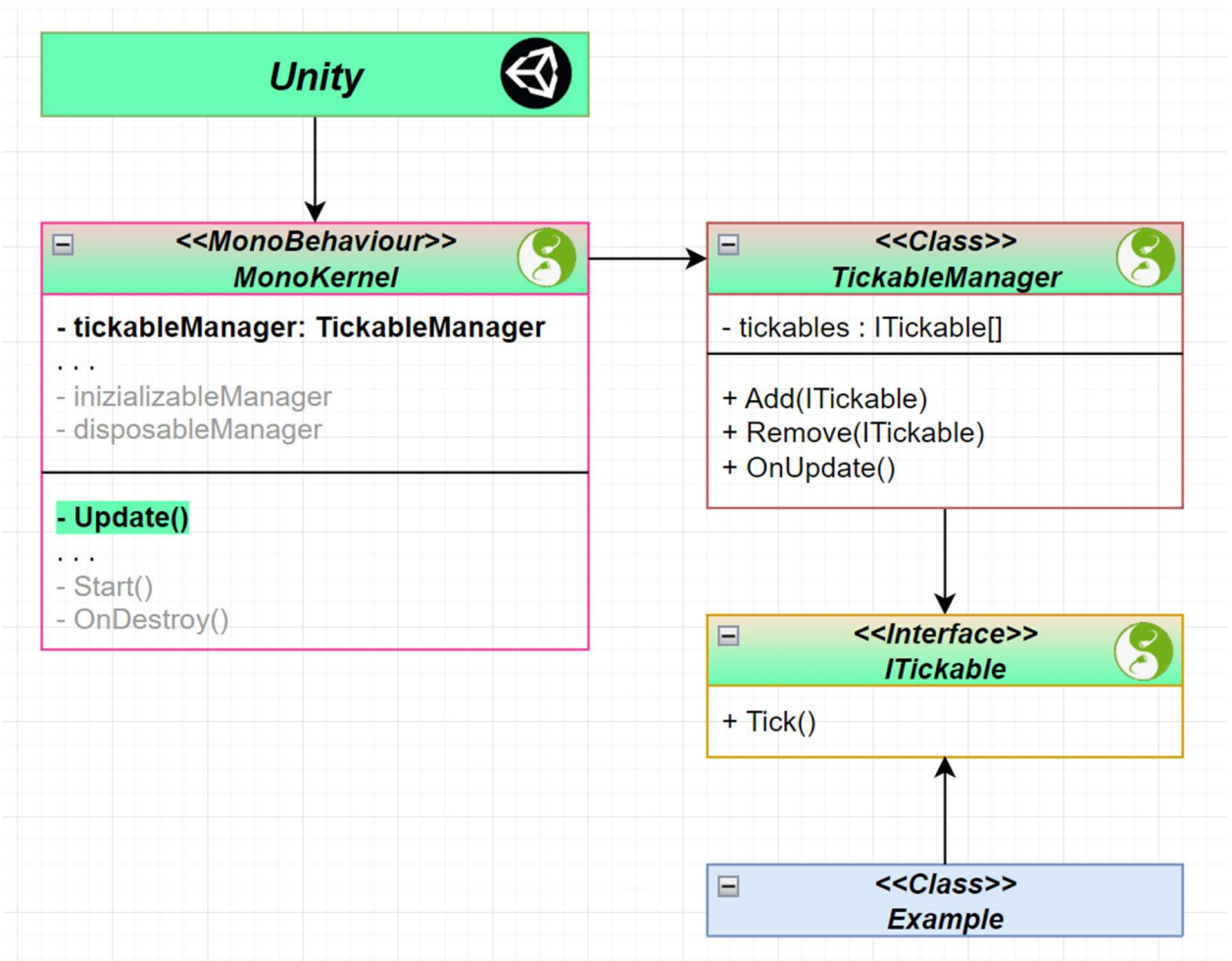


Рисунок 4. Часть внутренней реализации Zenject.

Для каждого контекста Zenject создает **MonoKernel**, который имеет связь с **Unity** и метод **Update**, в котором, с помощью **TickableManager** выполняет все методы **Tick** у классов, реализующих контракт **ITickable**. Решением является создание собственного **GameTickableManager**, похожего на **TickableManager**, который связан с **GameCycle** и выполняет метод **Tick**, только тогда, когда текущим событием в **GameCycle** не является «Пауза». В качестве расширения Zenject был разработан класс **GamKernel**, который полностью включает в себя **MonoKernel** и разработанный **GameTickableManager** (Рисунок 5).

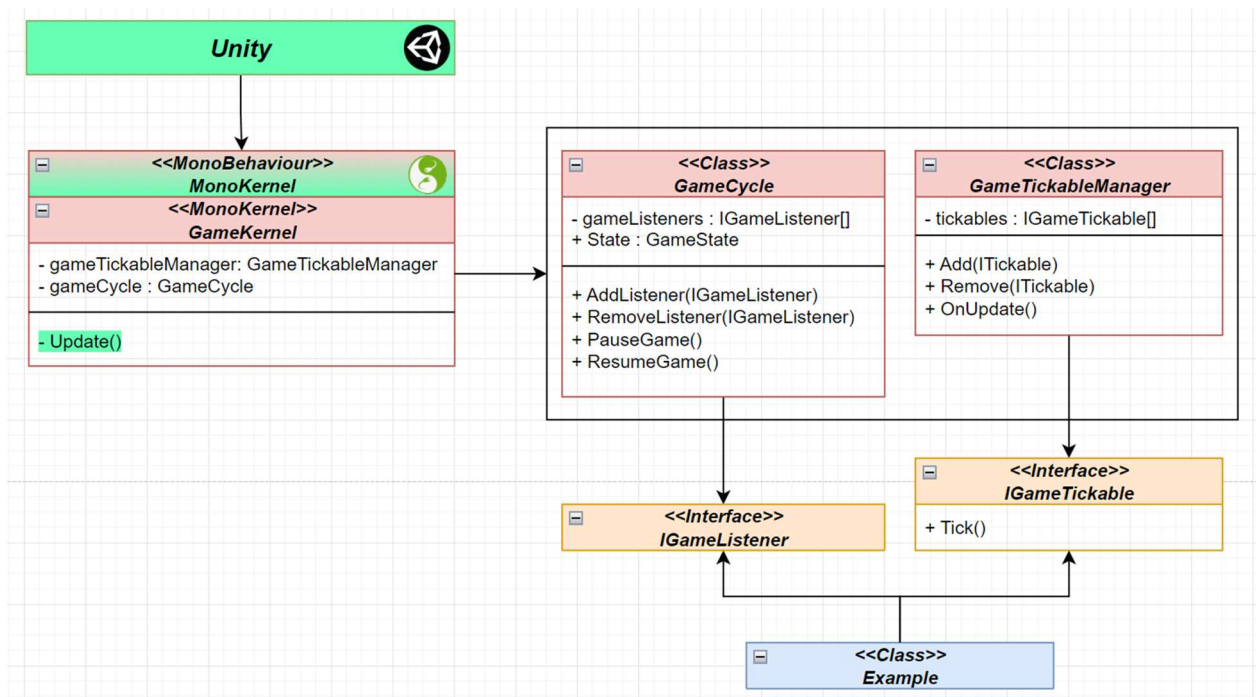


Рисунок 5. Расширение плагина Zenject.

## Регистрация классов в Real-Time

Рассмотрим регистрацию классов, которые создаются в процессе игры, то есть в Real-Time. Если уже созданные классы регистрируются в DiContainer и внедряются им в разработанные ранее системы, то классы, которые создаются в Real-Time, никуда не регистрируются (Рисунок 6). К тому же, если в игре есть Memory Pool и требуется отключать/включать определенные экземпляры классов, то такие классы также не должны выполнять никаких методов в отключенном состоянии.

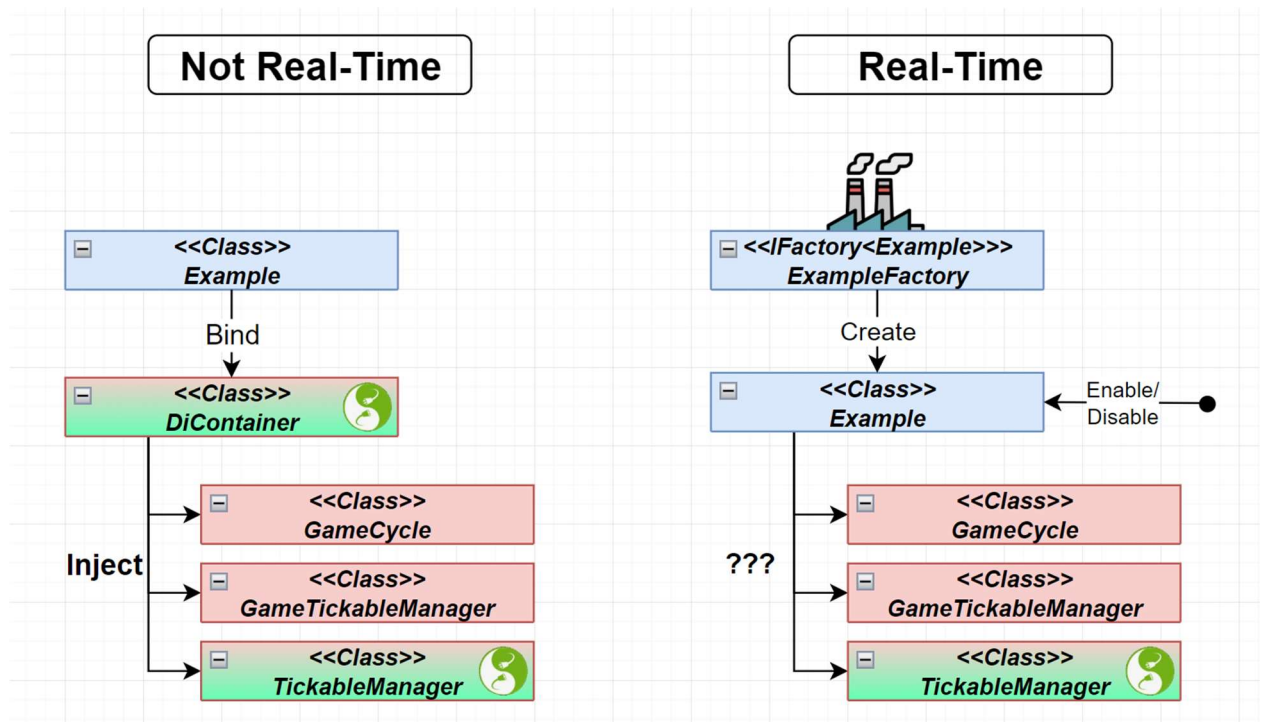


Рисунок 6. Проблема регистрации классов в Real-Time.

Решением является создание системы регистрации классов в Real-Time по тем же, ранее рассмотренным принципам (Рисунок 7). Был разработан контракт фабрики, создающей экземпляры классов, которые нужно зарегистрировать и контракт самого класса, который является регистрируемым в реальном времени. Все такие фабрики внедряются в систему `RealTimeRegistration`, которая подписывается на создание экземпляров и при их создании подписывается на их сигналы `OnEnable/OnDisable`, при вызове которых подписывает/отписывает экземпляр в разработанные ранее системы.

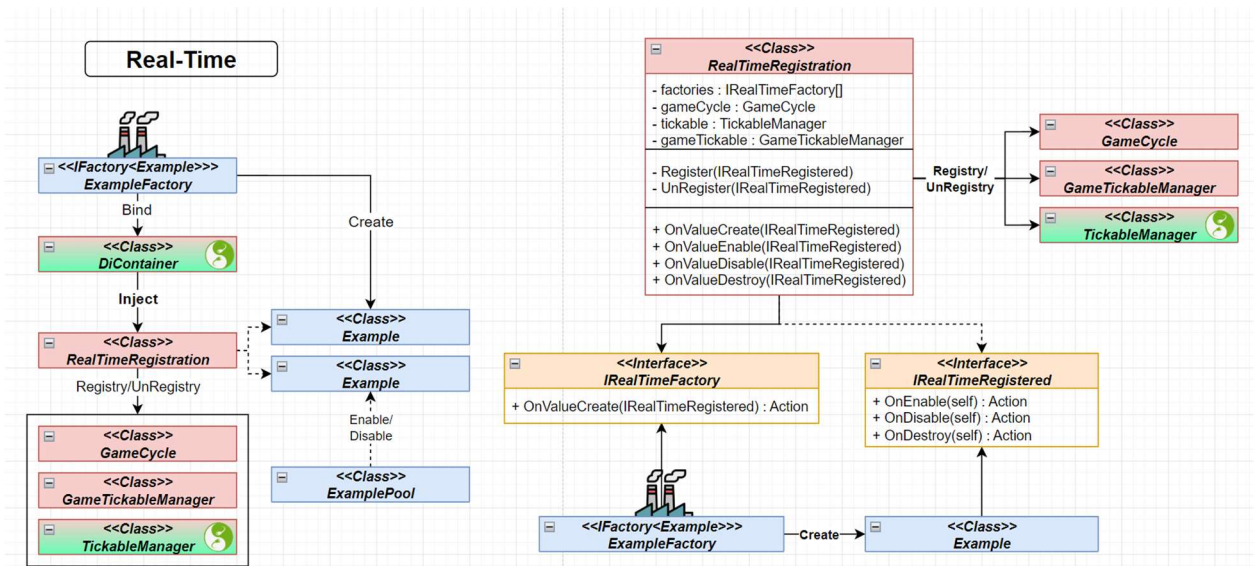


Рисунок 7. Система регистрации классов в Real-Time.

## Заключение

Анализ уже имеющихся, а также разработка собственных решений данной задачи позволяет сделать вывод, что для эффективного командного взаимодействия и поддержанию архитектуры масштабного проекта лучшим решением будет следование принципам DIP и концепции IoC.

Выше описанное решение позволяет разработчику создавать и изменять объекты независимо от инфраструктуры, которая ими управляет, что позволяет разрабатывать их параллельно, не имея информации о том, как работает та или иная часть, что значительно ускоряет процесс разработки и добавления нового контента.

Также, вся разработанная система устроена таким образом, что во всем проекте выполняется лишь один метод Update из Unity, что дополнительно является существенной оптимизацией.

## Литература

12. Metsker, Steven John, and William C. Wake. *Design Patterns in C# and .NET*. Addison-Wesley, 2006.

## Рецензия на статью

### «Архитектура компьютерной игры на Unity»

В рецензируемой статье подробно аргументирована актуальность существующей проблемы оповещения классов об определенных игровых событиях, описаны возможные препятствия. В статье описана архитектура, которую предоставляет движок Unity и плагин Zenject, приведено решение в виде принципа Dependency Inversion Principle и концепции Inversion of Control. Приведено сравнение использования плагина Zenject и стандартных средств Unity. Статья проиллюстрирована большим количеством диаграмм классов, облегчающих восприятие материала.

Можно выделить следующие критические замечания:

1. Не обоснован выбор движка Unity в качестве направления исследования. Unity не является единственной альтернативой в рассматриваемой предметной области, поэтому краткое описание других возможных решений обеспечило бы более глубокий анализ предметной области.
2. Аргументация выбора паттерна Dependency Injection слишком кратка. Приведена единственная проблема, которую данный паттерн решает, что не позволяет сделать вывод о необходимости выбора именно данного паттерна.
3. Слабо обоснован выбор плагина Zenject. Описано его преимущество перед стандартными методами Unity, но не аргументирован выбор именно этого конкретного плагина.
4. Не приведён анализ, позволяющий сделать выбор о необходимости следовании Dependency Inversion Principle и Inversion of Control, как записано в заключении. Вместо этого следование принципам постулируется без аргументации.
5. Малое количество использованной литературы, что создаёт впечатление о недостаточном анализе предметной области и существующих решений.

Кроме того, в статью необходимо внести следующие правки:

- В тексте статьи нет ссылок на используемую литературу.
- Масштаб на рисунке 7 слишком мелкий, что делает его нечитаемым в печатном виде.

- Сетка на рисунках несколько затрудняет восприятие.

В целом, сильной стороной статьи является подробное описание рассматриваемых в ней средств и методов. В то же время, статье требуется доработка ввиду недостаточного анализа существующей предметной области. Ряд положений автором просто постулируются, без какой-либо аргументации. Возможно, автору следует вынести использования плагина Zenject в заголовок статьи, поскольку большая часть описанных в статье методов относится именно к нему, а не к движку Unity в целом. Рецензент настаивает на внесение данных изменений, прежде чем статья будет опубликована.

# Мобильное приложение для распознавания шахматных фигур на изображении

*А.А. Калинин<sup>1</sup>*

*<sup>1</sup>Южный федеральный университет, Ростов-на-Дону*

**Аннотация:** В данной работе рассматривается задача автоматического распознавания шахматных фигур на фотографии 2D-доски. Разработана сверточная нейронная сеть (CNN), определяющая расположение и тип фигуры в каждой клетке доски. Создано мобильное приложение для Android, позволяющее пользователю отправлять фотографию и получать ссылку на позицию в lichess.org. Прототип продемонстрировал высокую точность распознавания и практическую применимость.

**Ключевые слова:** распознавание образов, сверточные нейронные сети, шахматы, компьютерное зрение, мобильные приложения, Android, lichess, анализ позиций, нейросети, автоматизация.

## Введение

Распознавание объектов на изображениях является одной из ключевых задач компьютерного зрения. Автоматизация распознавания шахматных позиций имеет прикладное значение в образовательных и соревновательных целях.

Цель работы — разработка метода распознавания шахматных фигур на 2D-фотографиях с дальнейшей интеграцией в мобильное приложение.

## Постановка задачи

Необходимо разработать систему, которая принимает изображение шахматной доски, разбивает его на сетку  $8 \times 8$  клеток, классифицирует содержимое каждой клетки (фигура или пустота), формирует позицию в формате FEN (Forsyth-Edwards Notation), открывает соответствующую позицию на платформе lichess.org.

Основные проблемы:

- различие во внешнем виде шахматных фигур
- искажения изображения из-за угла съемки
- различные условия освещения
- ограниченные вычислительные ресурсы мобильных устройств



## Обучение модели

Обучающая выборка была собрана из различных датасетов на сайте: *kaggle.com*.

Собранная выборка включает в себя реальные фотографии шахматных партий, а также синтетически сгенерированные изображения различных стилей шахматных комплектов.

Каждое изображение размечалось, указывая наличие и тип фигуры в каждой из 64 клеток доски.

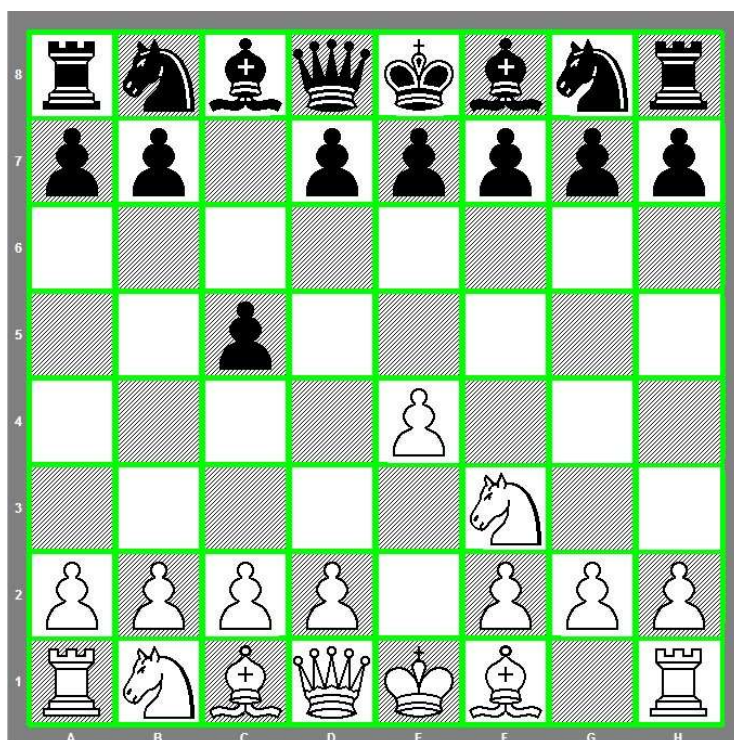


Рис. 1. – Разбиение шахматной доски на 64 клетки

Для повышения обобщающей способности модели применялись аугментации:

- случайные повороты (до  $\pm 10^\circ$ )
- изменение яркости и контраста
- перспективные преобразования
- добавление шума

Для классификации содержимого каждой клетки использовалась сверточная нейронная сеть (CNN), имеющая следующую архитектуру:

1. несколько сверточных слоев с ядрами  $3 \times 3$  и функцией активации ReLU
2. операции субдискретизации через MaxPooling
3. два полносвязных слоя
4. выходной слой с функцией активации Softmax для классификации на 13 классов (6 белых фигур, 6 черных фигур, пустая клетка)

Модель имеет малое количество параметров, что позволяет использовать ее на мобильных устройствах без существенной потери скорости работы. Обучение модели проводилось на GPU с использованием фреймворка *tensorflow*.

После классификации всех 64 клеток:

1. собирается позиция в формате FEN
2. пустые клетки группируются по правилам шахматной записи
3. формируется URL для открытия позиции на платформе lichess.org

При формировании FEN учитывается ориентация доски и цвет фигур игрока.

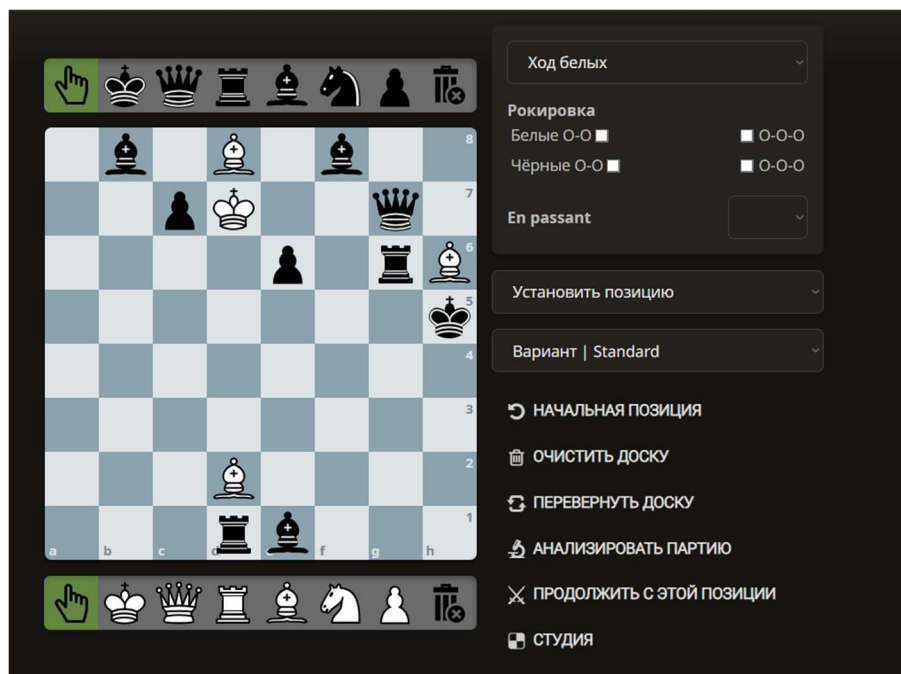


Рис. 2. – Сформированный FEN:

[1b1B1b2/2pK2q1/4p1rB/7k/8/8/3B4/3rb3%20w%20-%20-%200%201](https://lichess.org/1b1B1b2/2pK2q1/4p1rB/7k/8/8/3B4/3rb3%20w%20-%20-%200%201)

## Разработка мобильного приложения

Мобильное приложение разработано на языке Kotlin с использованием фреймворка Jetpack Compose для построения пользовательского интерфейса.

Основные модули приложения:

- модуль загрузки изображения
- модуль предобработки изображения и сетки доски
- модуль распознавания клеток через встроенную CNN
- модуль формирования FEN и открытия URL в браузере

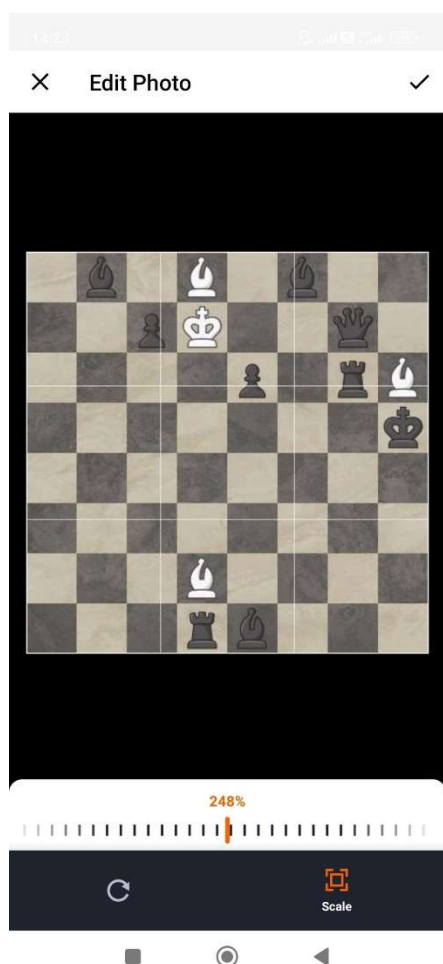


Рис. 3. – Модуль предобработки изображения и сетки доски

Пользователь делает фотографию шахматной позиции или выбирает существующее изображение. После обработки фотографии приложение:

1. Разбивает доску на  $8 \times 8$  клеток.

2. Распознает содержимое каждой клетки через CNN.
3. Формирует строку FEN.
4. Открывает браузер по сгенерированной ссылке вида:
5. [https://lichess.org/editor/FEN\\_STRING](https://lichess.org/editor/FEN_STRING)

Где FEN\_STRING заменяется на распознанную позицию.

## Результаты

Точность классификации содержимого клеток на тестовой выборке составила 98%.

Среднее время обработки полного изображения на устройстве среднего уровня составило около 2 секунд.

Тестирование на реальных фотографиях показало, что в 92% случаев позиции распознаются корректно.

В ходе работы была успешно решена задача автоматического распознавания шахматных позиций на фотографиях с помощью сверточной нейронной сети.

Созданное мобильное приложение позволяет быстро анализировать реальные партии, упрощая процесс переноса позиций в цифровой формат.

В перспективе планируется расширение функциональности:

- повышение устойчивости модели к сильным искажениям и шуму
- добавление поддержки трехмерных шахматных досок
- обучение модели на более разнообразных наборах данных
- улучшение мобильного приложения

Исходный код доступен на платформе GitHub:

<https://github.com/Kalsash/ChessPiecesRecognitionApp>

## **Литература**

1. Krizhevsky A., Sutskever I., Hinton G.E. Разработка сверточных сетей для классификации изображений // Advances in Neural Information Processing Systems. 2012.
2. He K., Zhang X., Ren S., Sun J. Глубокое обучение с остаточными связями // IEEE Conference on Computer Vision and Pattern Recognition. 2016.
3. Bishop C.M. Распознавание образов и машинное обучение. Springer, 2006.
4. Redmon J., Divvala S., Girshick R., Farhadi A. YOLO: Единовременное обнаружение объектов // IEEE Conference on Computer Vision and Pattern Recognition. 2016.
5. Paszke A. и др. PyTorch: высокопроизводительная библиотека глубокого обучения // Advances in Neural Information Processing Systems. 2019.

## **References**

1. Krizhevsky A., Sutskever I., Hinton G.E. ImageNet Classification with Deep Convolutional Neural Networks. Advances in Neural Information Processing Systems, 2012.
2. He K., Zhang X., Ren S., Sun J. Deep Residual Learning for Image Recognition. Proceedings of the IEEE CVPR, 2016.
3. Bishop C.M. Pattern Recognition and Machine Learning. Springer, 2006.
4. Redmon J., Divvala S., Girshick R., Farhadi A. You Only Look Once: Unified, Real-Time Object Detection. Proceedings of the IEEE CVPR, 2016.
5. Paszke A., et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. Advances in Neural Information Processing Systems, 2019.

## **Рецензия на статью «Мобильное приложение для распознавания шахматных фигур на изображениях»**

Рецензируемая статья посвящена актуальной проблеме автоматического распознавания шахматных позиций с использованием технологий компьютерного зрения и машинного обучения. Автор предлагает практическое решение в виде мобильного приложения для Android, что представляет несомненный интерес для широкого круга пользователей - от любителей шахмат до профессиональных игроков.

Основные достоинства работы включают:

1. Разработку специализированной сверточной нейронной сети (CNN), адаптированной для работы на мобильных устройствах
2. Создание комплексного решения от обработки изображения до генерации FEN-нотации
3. Демонстрацию высоких показателей точности (98% на тестовой выборке)
4. Практическую реализацию в виде рабочего прототипа приложения

Однако в ходе анализа работы были выявлены следующие существенные замечания:

В методологическом плане вызывает вопросы недостаточная детализация процесса обучения модели. Автор не указывает объем обучающей выборки, соотношение классов в датасете, конкретные параметры аугментации данных. Описание архитектуры CNN носит слишком общий характер.

В разделе, посвященном результатам, заявленная высокая точность требует дополнительного обоснования. Не представлен анализ ошибок - какие именно фигуры чаще всего путает алгоритм, как влияет на точность угол съемки или освещение. Отсутствует сравнение с существующими аналогами, что затрудняет оценку научной новизны предложенного решения.

В техническом плане вызывает вопросы реализация обработки изображений:

1. Не описаны алгоритмы определения границ доски при сильных перспективных искажениях
2. Не указано, как система определяет ориентацию доски (за кого играет пользователь)
3. Не приведены данные о потреблении ресурсов на различных мобильных устройствах

Также следует отметить недостатки в оформлении работы:

- Рисунки не имеют четких ссылок в тексте
- Список литературы содержит смешение русскоязычных и англоязычных источников без единого стандарта оформления

Для повышения научной ценности работы рекомендуется:

1. Дополнить методологический раздел конкретными параметрами модели и обучения
2. Привести детальный анализ ошибок с примерами проблемных случаев
3. Добавить сравнение с существующими аналогами по ключевым метрикам
4. Устранить отмеченные недочеты в оформлении
5. Описать ограничения предложенного решения и направления дальнейшего развития

Несмотря на указанные замечания, работа представляет значительный практический интерес и после соответствующей доработки может быть рекомендована к публикации. Предложенное решение имеет хороший потенциал для применения в образовательных целях и профессиональной шахматной практике. Рецензент считает целесообразным опубликовать статью после устранения основных замечаний, но оставляет за автором право окончательного решения по внесению изменений.

# Искусственный интеллект в играх на основе полезностей (Utility)

А.А. Праведник

*Южный федеральный университет, Ростов-на-Дону*

**Аннотация:** В статье рассматривается применение искусственного интеллекта (ИИ) в видеоиграх на основе системы полезностей (Utility-based AI). Данный подход позволяет создавать более гибкие и реалистичные поведения неигровых персонажей (NPC), учитывая множественные факторы для принятия решений. В работе анализируются ключевые принципы Utility-систем, их преимущества перед другими методами (такими как конечные автоматы и деревья поведения), а также приводятся практические примеры реализации в современных играх, включая алгоритмы оценки полезности и способы оптимизации.

**Ключевые слова:** искусственный интеллект, Utility-based AI, видеоигры, NPC, принятие решений, игровая логика, функции полезности.

**Abstract:** This article explores the application of artificial intelligence (AI) in video games based on Utility systems. Utility-based AI enables the creation of more flexible and realistic non-player character (NPC) behaviors by considering multiple decision-making factors. The paper analyzes the core principles of Utility systems, their advantages over other methods (such as finite state machines and behavior trees), and provides detailed implementation examples, including utility scoring algorithms and optimization techniques.

**Keywords:** artificial intelligence, Utility-based AI, video games, NPC, decision-making, game logic, utility functions.

## Введение

Современные видеоигры требуют от неигровых персонажей (NPC) сложного и правдоподобного поведения. Традиционные методы, такие как конечные автоматы (FSM) и деревья поведения (Behavior Trees), имеют ограничения в гибкости и масштабируемости. В отличие от них, Utility-based AI предлагает более динамичный подход, где решения принимаются на основе оценки полезности различных действий.

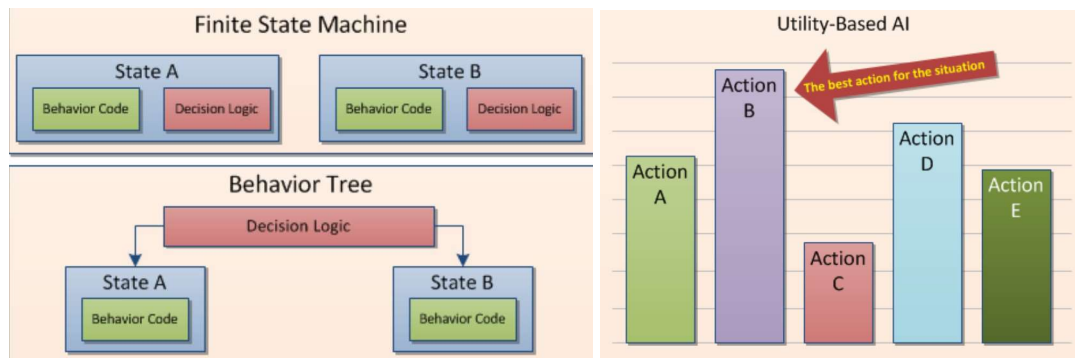


Рис. 1. - Сравнение архитектур FSM, Behavior Trees и Utility AI



Utility-системы позволяют NPC анализировать множество факторов (здоровье, расстояние до цели, наличие ресурсов) и выбирать оптимальное действие, что делает их поведение более естественным. В данной статье исследуются принципы работы Utility AI, его преимущества, практическое применение в игровой индустрии, а также приводятся конкретные примеры из популярных игр.

## Исследование

Utility AI базируется на концепции функций полезности, которые оценивают каждое возможное действие по числовому значению. Чем выше полезность действия в текущем контексте, тем больше вероятность его выбора.

### 1. Формальное определение полезности:

$$U(a) = \sum_i w_i * f(a_i)$$

Где:

- $U(a)$  - общая полезность действия **a**
- $w_i$  - вес фактора под номером **i**
- $f(a_i)$  - функция оценки фактора под номером **i** для действия **a**

Кривые полезности:

- Линейная:  $y = kx$
- Квадратичная:  $y = x^2$
- Логарифмическая:  $y = \log(x)$

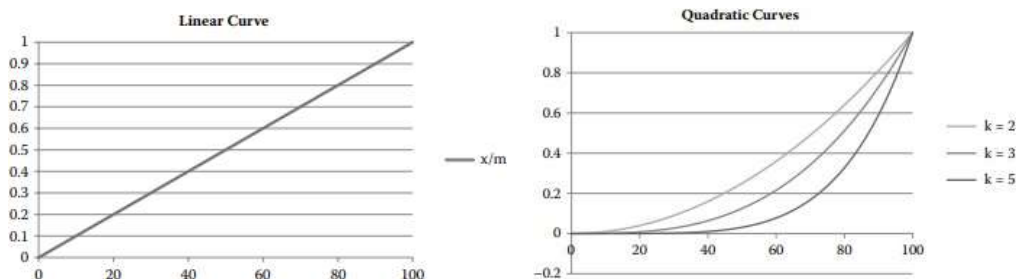


Рис. 2.1. - Различные типы кривых полезности - линейная и квадратичная

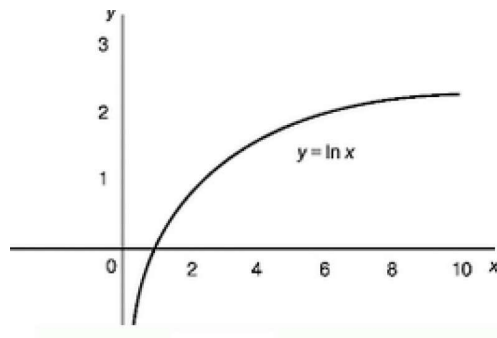


Рис. 2.2. - Различные типы кривых полезности - логарифмическая

## 2. Преимущества перед другими методами.

Таблица №1

Сравнительная таблица методов ИИ

| Метод      | Гибкость | Производительность | Сложность реализации |
|------------|----------|--------------------|----------------------|
| FSM        | Низкая   | Высокая            | Низкая               |
| BT         | Средняя  | Средняя            | Средняя              |
| Utility AI | Высокая  | Средняя            | Высокая              |

## 3. Примеры использования в играх

### The Sims: Динамика потребностей



Рис. 3. - Потребности, от которых зависит полезность действий в The Sims

### StarCraft II: Боевые решения Функция

полезности атаки:

$$U(\text{attack}) = 0.7 * \text{health\_score} + 0.3 * \text{distance\_score}$$

#### 4. Реализация Utility AI

Оптимизированный алгоритм выбора действий (псевдокод):

```
function SelectBestAction(actions, game_state){  
    best_action = None;  
    max_score = -infinite;  
  
    for action in actions{  
        score = 0;  
        for consideration in action.considerations{  
            value = consideration.evaluate(game_state);  
            score += consideration.weight * consideration.curve(value);  
        }  
        if score > max_score{  
            max_score = score;  
            best_action = action;  
        }  
    }  
    return best_action;  
}
```

#### Заключение

Utility-based AI предоставляет мощный инструмент для создания продвинутого игрового ИИ. Его способность динамически оценивать контекст и выбирать оптимальные действия делает поведение NPC более реалистичным и интересным для игрока. В будущем развитие машинного обучения может дополнить Utility-системы, позволяя NPC обучаться на основе игрового опыта.

#### Список литературы

1. Бурлаков А. Искусственный интеллект в играх. - М.: ДМК Пресс, 2020.
2. Мейер Д. Геймдизайн и программирование ИИ. - СПб.: Питер, 2019.
3. Бук М. Алгоритмы игрового ИИ. - М.: Вильямс, 2018.
4. Иванов Е. Практическое применение Utility-подхода в игровом ИИ //

Журнал "Разработка игр", 2021, №3, с. 45-52.

5. Петров С. Математические модели принятия решений в играх - Киев: Наукова думка, 2019.

6. Громов А. Современные методы искусственного интеллекта в игровой индустрии - СПб.: БХВ-Петербург, 2022.

7. Смирнов В. Оптимизация Utility-систем для мобильных платформ // Материалы конференции "GameDev Pro", 2020.

## References

1. Burlakov A. Artificial Intelligence in Games. - Moscow: DMK Press, 2020.
2. Meyer D. Game Design and AI Programming. - St. Petersburg: Piter, 2019.
3. Book M. Game AI Algorithms. - Moscow: Williams, 2018.
4. Ivanov E. Practical Application of Utility Approach in Game AI // "Game Development" Journal, 2021, No.3, pp. 45-52.
5. Petrov S. Mathematical Models of Decision Making in Games - Kiev: Naukova Dumka, 2019.
6. Gromov A. Modern Artificial Intelligence Methods in Game Industry - St. Petersburg: BHV-Petersburg, 2022.
7. Smirnov V. Optimization of Utility Systems for Mobile Platforms // Proceedings of "GameDev Pro" Conference, 2020.

### **Рецензия на статью А.А. Праведника “Искусственный интеллект в играх на основе полезностей (Utility)”**

Крамаренко Илья Алексеевич,

Южный федеральный университет

Статья представляет глубокий анализ применения искусственного интеллекта в видеоиграх с акцентом на систему полезностей (utility-based AI). Автор успешно демонстрирует, как этот подход позволяет создавать более гибкие и реалистичные поведения неигровых персонажей (NPC), что актуально в контексте современных требований к игровому дизайну.

## **Содержание и структура**

Статья начинается с аннотации, которая четко формулирует основные цели и задачи исследования. Введение логично подводит к теме, подчеркивая ограничения традиционных методов, таких как конечные автоматы и деревья поведения. Это создает основу для дальнейшего обсуждения преимуществ utility-based AI.

Автор подробно объясняет принципы работы систем, включая формальное определение полезности и различные типы кривых полезности. Приведенные формулы и графики (например, сравнение с архитектурами FSM и Behavior Trees) делают материал более наглядным и понятным.

## **Преимущества**

Одним из ключевых моментов статьи является сравнение utility-based AI с другими методами. Таблица, представленная в статье, наглядно демонстрирует его гибкость, однако подчёркивает и сложность реализации по сравнению с другими подходами, что демонстрирует значимость данного метода в контексте создания более сложных и адаптивных NPC.

## **Практическое применение**

Статья также включает примеры использования такого искусственного интеллекта в известных играх, таких как "The Sims" и "StarCraft II". Эти примеры не только иллюстрируют теоретические аспекты, но и показывают, как системы на основе полезности могут быть реализованы на практике. Это делает материал более доступным для читателей, интересующихся разработкой игр.

## **Заключение и перспективы**

Заключение статьи подводит итоги и акцентирует внимание на будущем развитии utility-based AI, особенно в контексте машинного обучения; данная разработка открывает новые горизонты для дальнейших исследований и нововведений в области игрового ИИ.

### **Предлагаемые исправления**

1. Рис. 1, 2-1, 2-2 заменить на созданные нативно в редакторе документов графики и гистограммы.
2. Рис. 3 увеличить разрешение изображения или использовать пример из недавних игр серии.
3. Отформатировать уравнения и функции как формулы в редакторе.
4. Использовать моноширинный фонт для вставки кода.

### **Общая оценка**

Статья является ценным вкладом в область искусственного интеллекта в играх. Она хорошо структурирована, содержит актуальную информацию и примеры, что делает ее полезной как для исследователей, так и для разработчиков.

В целом, работа демонстрирует высокий уровень понимания темы и предлагает читателям полезные инструменты для дальнейшего изучения и применения utility-based AI в игровой индустрии.

# Разработка контроллера в souls-like игре.

К.С. Зиннатуллин<sup>1</sup>

<sup>1</sup>*Институт математики, механики и компьютерных наук им. И. И. Воровича,  
Ростов-на-Дону*

**Аннотация:** Разработка контроллера персонажей является ключевым аспектом в игровых проектах, так как его архитектура определяет универсальность, структурированность и легкость обслуживания, что влияет на развитие проекта. Универсальность кода позволяет использовать одну и ту же систему для различных задач, а удобство обслуживания обеспечивает легкую интеграцию новых элементов.

**Ключевые слова:** разработка игры, souls-like, программирование, архитектура, контроллер персонажа, чистый код, расширяемость кода, паттерны проектирования, ООП, unity, csharp,

## Введение

Текст В мире программирования создание кода, который легко модифицировать, понимать и расширять, является первостепенной задачей. В IT-сфере, включая игровое направление, разработчики часто сталкиваются с трудностями, вызванными “неэффективным” кодом [2], что может обернуться серьезными задержками в разработке и затруднить дальнейшее развитие проекта, его обновление и добавление новых функций. В масштабных проектах особую актуальность приобретает разработка читабельного, структурированного и легко обслуживаемого кода для различных контроллеров.

## Тестовая среда и ассеты

Для тестирования и демонстрации предложенного решения была создана специализированная тестовая сцена [9], имитирующая условия souls-like игры. В рамках подготовки проекта были разработаны или адаптированы необходимые ассеты, включая 3D-модели персонажа, врагов и окружающей среды, а также набор анимаций, соответствующих базовым механикам (перемещение, атаки, увороты, блоки и т. д.).

## Контроллер в игровой разработке

Контроллер в контексте игровой разработки, особенно в игровых движках, таких как Unity [4] и Unreal Engine, часто представляется как скрипт,



содержащий код, который управляет поведением и взаимодействием игрового объекта. Этот скрипт определяет, как объект будет реагировать на входные данные от игрока и взаимодействовать с окружающей средой. От того, насколько качественно построена архитектура игрового контроллера в скрипте или скриптах, зависит его универсальность, читабельность и удобность в обслуживании.

### **Универсальность и функциональность полученного контроллера**

Универсальность заключается в возможности использования одной системы для других задач. Например, в данном решении реализованный контроллер может быть использован не только для управления игровым персонажем, но и для управления неиграбельными объектами искусственным интеллектом.

Удобность в обслуживании подразумевает, что в реализованную систему может быть что-то легко интегрируемо и дополнено. В данном проекте это выражено тем, что контроллер позволяет достаточно просто создавать различные состояния объекта и его взаимодействия с окружающим миром. Что для souls-like игр очень актуально, так как данные игры подразумевают большую вариативность действий (рывки, кувырки, блоки, использование различных оружия и т. д.) игрового персонажа. Также были легко интегрируемы различные другие механики (стамина, мана, выносливость), которые на вышеописанные действия влияют.

Стоит отметить, что реализованная система управляет не только обработкой ввода и перемещением объекта, но также анимациями, взаимодействием с другими объектами, созданием звуковых эффектов и т. д. А всевозможные нововведения могут быть легко реализуемы ввиду легкой адаптивности полученного кода.

### **Заключение**

Разработанное решение было реализовано на языке C# в рамках проекта, созданного на платформе Unity. Использование подготовленных ассетов и

тестовой сцены позволило наглядно продемонстрировать работоспособность и гибкость предложенной архитектуры контроллера.

### **Литература**

13. Гамма Э., Хелм Р., Джонсон Р., Влиссидес Дж. Приемы объектно-ориентированного проектирования. Паттерны проектирования. - СПб.: Питер, 2020. - 366 с.
14. Макконнелл С. Совершенный код. - М.: Русская редакция, 2019. - 896 с.
15. Нистром Р. Паттерны игрового программирования. - М.: Диалектика, 2021. - 352 с.
16. Unity Technologies. Официальная документация Unity: Контроллеры персонажей. 2023. URL: <https://docs.unity3d.com/Manual/class-CharacterController.html>
17. Microsoft. Документация по C#. 2023. URL: <https://learn.microsoft.com/ru-ru/dotnet/csharp/>
18. Грегори Дж. Архитектура игровых движков. - М.: ДМК Пресс, 2020. - 1040 с.
19. Кук Р. Архитектура игр: существует ли "правильный" способ? // Gamasutra. 2012. URL: [https://www.gamasutra.com/view/feature/134743/game\\_architecture\\_is\\_there\\_a\\_.php](https://www.gamasutra.com/view/feature/134743/game_architecture_is_there_a_.php)
20. Хокинг Дж. Unity в действии: кроссплатформенная разработка игр на C#. - Manning Publications, 2020. - 392 с.
21. Catlike Coding. Движение и анимация в Unity. 2023. URL: <https://catlikecoding.com/unity/tutorials/movement/>
22. GameDev StackExchange. Лучшие практики для контроллера персонажа. 2023. URL: <https://gamedev.stackexchange.com/questions/tagged/character-controller>

## References

1. Gamma, E., Helm, R., Johnson, R., Vlissides, J. Design Patterns: Elements of Reusable Object-Oriented Software. - Addison-Wesley, 1994. - 395 p.
2. McConnell, S. Code Complete: A Practical Handbook of Software Construction. - Microsoft Press, 2004. - 914 p.
3. Nystrom, R. Game Programming Patterns. - Genever Benning, 2014. - 345 p.
4. Unity Technologies. Unity Official Documentation: Character Controllers. 2023. URL: <https://docs.unity3d.com/Manual/class-CharacterController.html>
5. Microsoft. C# Documentation. 2023. URL: <https://learn.microsoft.com/en-us/dotnet/csharp/>
6. Gregory, J. Game Engine Architecture. - CRC Press, 2018. - 1040 p.
7. Cook, R. Game Architecture: Is There a 'Right' Way? // Gamasutra. 2012. URL: [https://www.gamasutra.com/view/feature/134743/game\\_architecture\\_is\\_the\\_re\\_a\\_.php](https://www.gamasutra.com/view/feature/134743/game_architecture_is_the_re_a_.php)
8. Hocking, J. Unity in Action: Multiplatform Game Development in C#. - Manning Publications, 2020. - 392 p.
9. Catlike Coding. Movement and Animation in Unity. 2023. URL: <https://catlikecoding.com/unity/tutorials/movement/>
10. GameDev StackExchange. Character Controller Best Practices. 2023. URL: <https://gamedev.stackexchange.com/questions/tagged/character-controller>

Рецензия на статью К.С. Зиннатуллина «Разработка контроллера в souls-like игре»

### Резюме

Статья посвящена разработке архитектуры контроллера персонажа для игр жанра souls-like с акцентом на универсальность, структурированность и удобство обслуживания кода. Автор рассматривает возможности повторного использования контроллера для различных игровых объектов, включая ИИ, и описывает интеграцию базовых механик, таких как перемещение, атаки,

блоки и стамина. Работа выполнена на языке C# с использованием игрового движка Unity и сопровождается тестовой средой и набором ассетов для демонстрации. Основное внимание уделено важности чистого и расширяемого кода в масштабных игровых проектах.

Список положительных сторон статьи

#### **1. Актуальность темы**

Вопросы архитектуры контроллеров персонажей в современных играх, особенно в жанре souls-like, крайне востребованы в индустрии и научном сообществе.

#### **2. Хорошая структурированность материала**

Статья логично разделена на тематические блоки - от постановки задачи до описания реализации и выводов, что облегчает восприятие.

#### **3. Практическая направленность**

Описано создание тестовой сцены и использование ассетов, что подтверждает работоспособность предложенного решения и его применимость на практике.

#### **4. Универсальность подхода**

Автор подчёркивает возможность использования контроллера не только для главного персонажа, но и для других игровых объектов, включая искусственный интеллект.

#### **5. Ссылки на современные источники**

Приведён обширный список литературы, включающий как классические, так и современные работы по архитектуре игр, паттернам проектирования и документации по Unity и C#.

#### **6. Фокус на расширяемости и чистоте кода**

В статье уделено внимание важности поддерживаемого и легко модифицируемого кода, что соответствует лучшим практикам программной инженерии.

**1. Недостаточная детализация технической реализации**

В разделе, посвящённом архитектуре контроллера, отсутствует подробный разбор конкретных паттернов проектирования, используемых в реализации. Не представлены примеры кода или схемы, иллюстрирующие структуру системы, что снижает практическую ценность статьи для разработчиков.

**2. Отсутствие сравнительного анализа**

В статье не проводится сравнение предложенного решения с альтернативными подходами к созданию контроллеров в souls-like играх. Это ограничивает понимание преимуществ и недостатков выбранной архитектуры.

**3. Недостаточное внимание специфике жанра**

Souls-like игры характеризуются сложным управлением персонажем, включая особенности анимаций и отзывчивости управления. В статье не раскрыты эти аспекты и не показано, как контроллер решает связанные с ними задачи.

**4. Отсутствие эмпирической оценки**

Не приведены результаты тестирования производительности, отзывов пользователей или метрики удобства обслуживания кода, что затрудняет оценку эффективности и качества разработанного контроллера.

**5. Обобщённый стиль изложения**

Некоторые разделы (например, введение и описание универсальности контроллера) содержат общие рассуждения без конкретных технических деталей, что снижает глубину анализа.

#### Мелкие правки

- В разделе «Тестовая среда и ассеты» уточните версии используемых инструментов и движка Unity для повышения воспроизводимости.
- Не хватает иллюстраций
- В разделе «Заключение» добавьте краткое резюме основных результатов и перспектив развития.

#### Итог

Автору следует расширить техническую часть статьи с конкретными примерами реализации и паттернами проектирования, провести сравнительный анализ с существующими решениями, а также добавить результаты тестирования и обсуждение особенностей жанра. Это повысит научную и практическую ценность работы и сделает её полезной для специалистов в области разработки игр.

# РАЗРАБОТКА МОДЕЛИ ЭКОНОМИЧЕСКОЙ СИСТЕМЫ ДЛЯ ИГРЫ С ПРОЦЕДУРНО-ГЕНЕРИРУЕМЫМ МИРОМ

*Таилова Н.Г.*

*ФГАОУ ВО «Южный федеральный университет», Институт  
математики, механики и компьютерных наук им. И. И. Воровича*

*г. Ростов-на-Дону*

[tailova@sfedu.ru](mailto:tailova@sfedu.ru)

**Аннотация:** В статье представлено программное решение, реализующее экономический симулятор на базе игрового движка DragonRuby, который позволяет моделировать рыночные процессы и поведение торговых агентов на графе узлов. Каждый узел обладает собственной динамической ценовой политикой, зависящей от спроса и предложения на товары. Игровая механика основана на перемещении агентов между узлами с целью получения максимальной прибыли посредством торговли товарами с учетом изменения цен и транспортных расходов.

**Ключевые слова:** экономический симулятор, игровая механика, dragonruby, агент, торговля, ценообразование, граф, узел, спрос, предложение, симуляция, прибыль, бюджет, стратегия.

На фоне активного развития экономики экономические симуляторы приобретают все большую популярность. Подобные игры создают чувство контроля над ресурсами, бюджетами и логистическими цепочками, что позволяет игрокам тренировать аналитическое и стратегическое мышление, которое помогает грамотно распределять свои ресурсы уже в реальной жизни [1]. В контексте данной работы в качестве основной технологической платформы выбран движок DragonRuby [2], поскольку он предоставляет широкие возможности для быстрого построения интерактивных систем с поддержкой сложной логики и удобного цикла обновления (tick-based), что важно при симуляции экономических механизмов и поведения торговых агентов.

На текущем этапе разработки программного решения реализована базовая инфраструктура, состоящая из карты, которая формируется графом узлов, каждый из которых символизирует точку входа в торгово-экономические отношения. Узлы обладают собственными начальными ценами на различные категории товаров. Изначальная ценовая политика определяется случайным образом с помощью встроенных возможностей языка Ruby.

При дальнейшем развитии сценария в расчет принимается динамическая модель изменения спроса и предложения: если продукт не пользуется спросом и перестает покупаться в узле, его цена постепенно снижается, а при резком всплеске интереса — возрастает. Эта логика базируется на идее имитации рынков, позволяющей воспроизвести естественное поведение экономических систем. Важными элементами этой системы являются автономные агенты-торговцы, осуществляющие перемещения между узлами. Их цель — максимизация своей выгоды, с помощью следующей стратегии: покупка товаров по более низким ценам и их реализация в узлах с более высокими ценами. Оба агента управляются игроком, позволяя ему быть непосредственным участником рынка. Каждый агент сохраняет информацию о последней известной ему цене на товар в конкретном узле. Данные о ценах в узлах обновляются ежедневно, где день — переход из одного узла в другой, что позволяет создавать более реалистичную модель экономического процесса.

### **Формальная постановка**

Экономическая часть игры использует представление игрового мира в виде графа  $G$ :

$$G = (V, E, c), \text{ где:}$$

$V$  — множество узлов (торговых точек)

$E \subseteq V \times V$  — ребра перемещений



$c : E \rightarrow \mathbb{R}^+$  - функция стоимости переходов

Для каждого товара  $i$  в узле  $v$  в момент симуляции  $t + 1$  динамически формируются цена покупки  $b_{v,i}^{(t+1)}$  и цена продажи  $s_{v,i}^{(t+1)}$ :

$$b_{v,i}^{(t+1)} = \max(b_{\min}, b_{v,i}^{(t)} + \alpha_{\text{buy}} \cdot q_{\text{buy},i}^{(t)})$$

$$s_{v,i}^{(t+1)} = \max(s_{\min}, s_{v,i}^{(t)} - \alpha_{\text{sell}} \cdot q_{\text{sell},i}^{(t)})$$

где  $\alpha_{\text{buy}}, \alpha_{\text{sell}} \geq 0$  – коэффициенты реакции цен,  $b_{\min}, s_{\min}$  – минимальные значения цен

Состояние участника (NPC или игрока) в момент  $t$  представлено в виде

$$S_t = (v_t, M_t, I_t, K_t), \text{ где}$$

$v_t \in V$  – текущий узел,  $M_t \geq 0$  – капитал,  $I_t : \tau \rightarrow \mathbb{N}$  - инвентарь (товар  $\rightarrow$  количество),  $K_t \subseteq V$  – посещенные узлы с известными ценами.

На каждом шаге симуляции участникам доступны следующие действия:

- Перемещение: Выбор  $v_{t+1}$  из соседей  $v_t$
- Торговля: Выбор  $q_{\text{buy},i}, q_{\text{sell},i}$

Ограничения:

$$M_t \geq c(v_t, v_{t+1}) \text{ (стоимость перехода)}$$

$$\sum_i q_{\text{buy},i} \cdot b_{v,i}^{(t)} \leq M_t - c(v_t, v_{t+1}) \text{ (бюджет)}$$

После выполнения участником действий его состояние обновляется следующим образом:

$$M_{t+1} = M_t - c(v_t, v_{t+1}) + \sum_i q_{\text{sell},i} \cdot s_{v,i}^{(t)} - \sum_i q_{\text{buy},i} \cdot b_{v,i}^{(t)}$$

$$I_{t+1}(i) = I_t(i) + q_{\text{buy},i} - q_{\text{sell},i}$$

$$K_{t+1} = \begin{cases} K_t \cup \{v_{t+1}\}, & \text{если новое посещение} \\ K_t, & \text{иначе} \end{cases}$$

Цель участника можно представить в виде задачи максимизации прибыли за время  $T$ :

$$Profit = \sum_{t=1}^T \left( \underbrace{\sum_i q_{\text{sell},i}^{(t)} \cdot s_{v,i}^{(t)}}_{\text{доход}} - \underbrace{\sum_i q_{\text{buy},i}^{(t)} \cdot b_{v,i}^{(t)}}_{\text{расход}} - \underbrace{c(v_t, v_{t+1})}_{\text{транспорт}} \right)$$

Участник решает задачу:

$$\max_{\{a_t\}_{t=0}^{T-1}} \mathbb{E}[Profit | S_0]$$

Где действие

$$a_t = (v_{t+1}, \{q_{\text{buy},i}\}, \{q_{\text{sell},i}\})$$

- Учитывать зависимость цен
- Балансировать между исследованием и использованием имеющейся информации
- Минимизировать суммарные транспортные издержки

### Техническая реализация

Интерфейсное решение реализовано на базе встроенных графических возможностей игрового движка DragonRuby, обеспечивающих быструю отрисовку объектов на экране и интуитивно понятное представление данных о состоянии узлов и агентов в виде хронологии. Система инициализации, состоящая из методов `init`, `handle_input` и `render`, позволяет задать все основные параметры (координаты узлов, связи между ними, сведения о товарах и начальных бюджетах агентов), а также регламентировать порядок обработки данных на каждом такте. В результате эта модульная архитектура оставляет пространство для дальнейшего расширения функционала: например, в

будущем планируется более тонкое регулирование цен, приближенное к реальному, более сложная конструкция взаимосвязи узлов, а также интеллектуальное поведение агентов, основанное на анализе движения рынка.

По мере добавления новых данных и усложнения механизмов спроса и предложения система будет приобретать все большее сходство с реальной экономической моделью.

### **Литература**

1. Построение экономических моделей в видеоиграх с точки зрения потребителя // Бизнес-образование в экономике знаний [Электронный ресурс] URL: [https://bibs-science.ru/archive/sbornik\\_21/2/bibs\\_science\\_2021\\_2\\_19.pdf](https://bibs-science.ru/archive/sbornik_21/2/bibs_science_2021_2_19.pdf)
2. Документация DragonRuby [Электронный ресурс] URL: <https://docs.dragonruby.org/#/>

### **References**

1. Postroenie ekonomicheskikh modeley v videoigrakh s tochki zreniya potrebitelya // Biznes-obrazovanie v ekonomike znaniy [Electronic resource]. URL: [https://bibs-science.ru/archive/sbornik\\_21/2/bibs\\_science\\_2021\\_2\\_19.pdf](https://bibs-science.ru/archive/sbornik_21/2/bibs_science_2021_2_19.pdf)
2. Dokumentatsiya DragonRuby [DragonRuby Documentation] [Electronic resource]. URL: <https://docs.dragonruby.org/#/>

# Рецензия на статью Таиловой Нины

В работе представлена архитектура экономического симулятора на движке DragonRuby для процедурно-генерируемого мира. Авторы описывают модель динамического ценообразования в узлах графа и стратегию торговых агентов, а также приводят формальную постановку задачи и базовую реализацию интерфейса. Предложенная система позволяет иллюстрировать движение спроса и предложения при учёте транспортных издержек. Однако статья не содержит экспериментальной валидации, анализа производительности и недостаточно раскрывает технические детали реализации.

## Критические замечания

1. **Отсутствие экспериментальной валидации:** В разделе «Техническая реализация» не приведены результаты тестирования (производительность, масштабируемость, корректность динамики цен) на реальных сценариях. Авторам следует добавить количественные эксперименты (стр. 4).
2. **Нечёткая формализация:** В «Формальной постановке» (стр. 3) не пояснены параметры функции стоимости переходов и коэффициенты реакции цен. Нужно уточнить диапазоны, источники значений и обоснование выбора.
3. **Отсутствие описания алгоритмов агентов:** В тексте указано, что агенты «решают задачу максимизации прибыли», но не приведён конкретный алгоритм (жадный выбор, MCTS, Q-learning). Лучше метод поиска оптимальных действий и обосновать его применимость.
4. **Неполное описание интерфейса:** В разделе GUI не указаны форматы входных/выходных данных, метод передачи состояния узлов и агентов. Лучше добавить подробный обзор API (методы `init`, `handle_input`, `render`) с примерами.

## Мелкие правки

- Исправить опечатку в аннотации: «дизайно» → «дизайна» (стр. 1).
- Добавить номера уравнений в формальной части и ссылаться на них в тексте (раздел «Формальная постановка»).
- Выделить жирным подписи осей на рис. 1 (если планируется публикация рисунков).
- Привести единый стиль оформления ссылок: в списке литературы оставить либо только русскоязычные, либо добавить DOI/ISBN для электронных ресурсов.
- Уточнить формат даты обновления цен («день – переход из одного узла в другой»): указать соответствие игровых тактов реальному времени.

## **Рекомендация редактору**

Требует доработки. Работа содержит интересную идею и формальное оформление, но нуждается в расширении экспериментальной части, уточнении алгоритмов агентов и более полном обзоре литературы. После устранения указанных замечаний статья станет близкой к публикации.

# Разработка кроссплатформенного приложения-интерактивной доски

*И.А. Крамаренко*

*Южный федеральный университет, Ростов-на-Дону*

**Аннотация:** Данная статья описывает идею и описание процесса разработки кроссплатформенного приложения, представляющего из себя интерактивную доску с виджетами.

**Ключевые слова:** виджет, модульная архитектура, интерактивность, кроссплатформенность, разработка, доска, персонализация, кастомизация, приложение, киоск.

В последние годы наблюдается значительный рост интереса к интерактивным интерфейсам, которые предоставляют пользователям гибкие и настраиваемые средства для взаимодействия с информацией. К примеру, подобные решения находят широкое применение в киосковых системах, где требуется обеспечить быстрый и интуитивно понятный доступ к необходимым функциям и информации конечному пользователю, однако подобный подход можно применить и в случае персонального использования программы в качестве персонального органайзера.

Создание решения, адаптируемого под такие сценарии, позволяет максимально эффективно использовать преимущества модульности и удобства интерфейса, что повышает общую эффективность и удовлетворённость пользователей.

**Цель разработки приложения - создание универсального интерфейса интерактивной доски с виджетами, который объединяет функции персонального органайзера для повышения продуктивности пользователя и программного обеспечения для киоска, обеспечивая удобное и интуитивное взаимодействие с набором настраиваемых виджетов для управления задачами, информацией и сервисами в различных контекстах использования.**

В качестве целевых платформ разработки было решено использовать операционные системы Windows 10/11, Linux и Android. Разработка ведётся на

операционной системе Linux, в стек технологий вошли язык Dart и фреймворк Flutter, среда разработки JetBrains IntelliJ IDEA, редактор кода Zed, система контроля версий Git и платформа GitHub.

Архитектура приложения реализуется на основе модульного подхода, который предполагает декомпозицию пользовательского интерфейса на автономные и инкапсулированные модули (виджеты) [1,2]. Каждый виджет представляет собой самостоятельный компонент с собственными механизмами управления состоянием и обработки пользовательских взаимодействий, что облегчает процессы развития, масштабирования и поддержки программного обеспечения.

Структура приложения включает следующие уровни:

1. Компонентный слой отвечает за визуализацию данных и взаимодействие с пользователем посредством отдельных виджетов;
2. Слой управления состоянием обеспечивает хранение и синхронизацию данных между компонентами, поддерживая целостность пользовательского интерфейса;
3. Слой бизнес-логики реализует алгоритмы обработки пользовательских действий и управление потоками данных в приложении;
4. Слой данных взаимодействует с внутренними и внешними источниками информации, предоставляя компонентам необходимый контент.

Данный архитектурный подход обладает рядом существенных преимуществ:

- Масштабируемость: модульность компонентов упрощает расширение функционала за счет добавления или изменения виджетов без необходимости кардинального изменения существующей системы;
- Повторное использование: компоненты могут многократно применяться внутри приложения, что способствует оптимизации процесса разработки;
- Упрощение поддержки: инкапсуляция логики и представления облегчает диагностику и устранение неполадок;
- Персонализация: возможность конфигурирования и динамического комбинирования виджетов обеспечивает адаптацию пользовательского интерфейса под индивидуальные потребности и предпочтения;
- Адаптивность: архитектура способствует эффективному использованию в различных сценариях, обеспечивая при этом высокую универсальность и удобство взаимодействия.

Схема архитектуры приложения предоставлена на рис. 1.

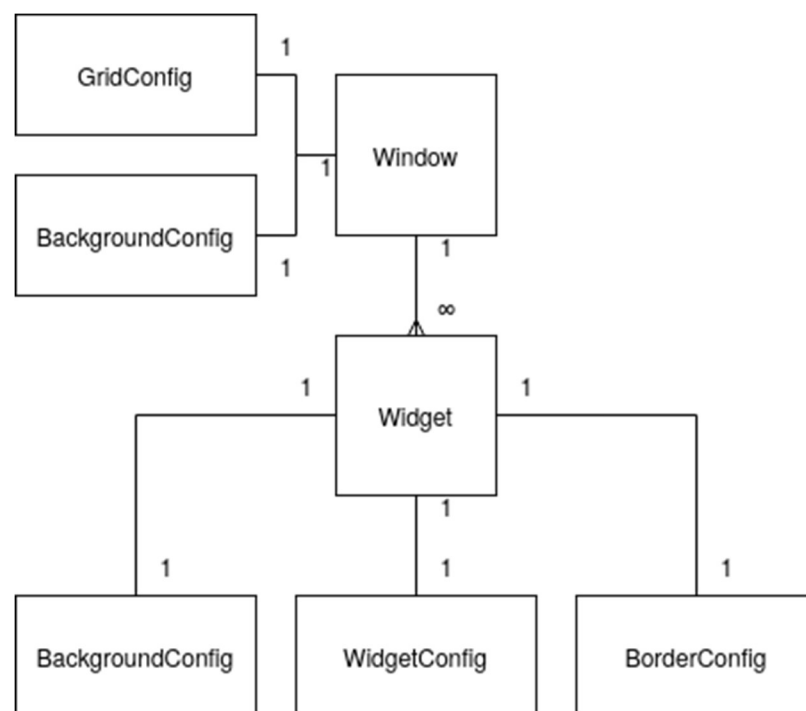


Рис. 1. Схема архитектуры приложения.



Ключевыми элементами приложения являются кастомизируемые виджеты, располагаемые и перемещаемые внутри окна. Одними из основных операций с виджетами являются перемещение, удаление и изменение размера. Перемещение осуществляется с помощью класса `LongPressDraggable` [3], реагирующего на долгое нажатие. Изменение размера осуществляется с помощью захвата и перемещения элемента `GestureDetector` [4], расположенного в правом нижнем углу виджета и появляющегося при включении режима редактирования. В правом верхнем углу расположена кнопка удаления виджета. Данные элементы видимы только тогда, когда разблокировано редактирование содержимого окна, и отображаются согласно рис.2.

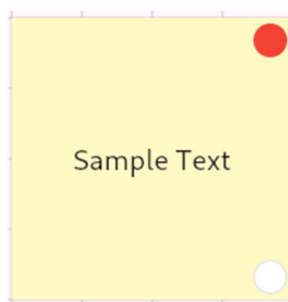


Рис. 2. – Отображение элементов взаимодействия при включённом режиме редактирования содержимого окна.

Виджеты можно располагать внутри окна в свободном порядке, но без наложения друг на друга. Также при выборе соответствующей опции на фоне окна отображается сетка, создаваемая с помощью специального класса, наследующего `CustomPainter`, и отображается системное сообщение, предлагающее исправить размещение содержимого окна согласно сетке. В случае согласия пользователя каждый виджет смещается на ближайшую возможную позицию. Данный процесс отображён на рис. 3-5.

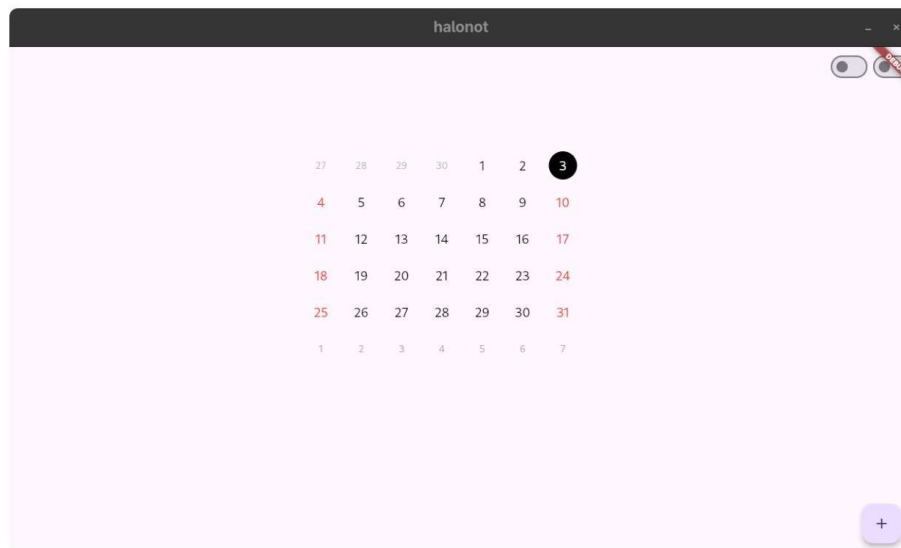


Рис. 3. Виджет-календарь, размещённый в окне не на сетке.

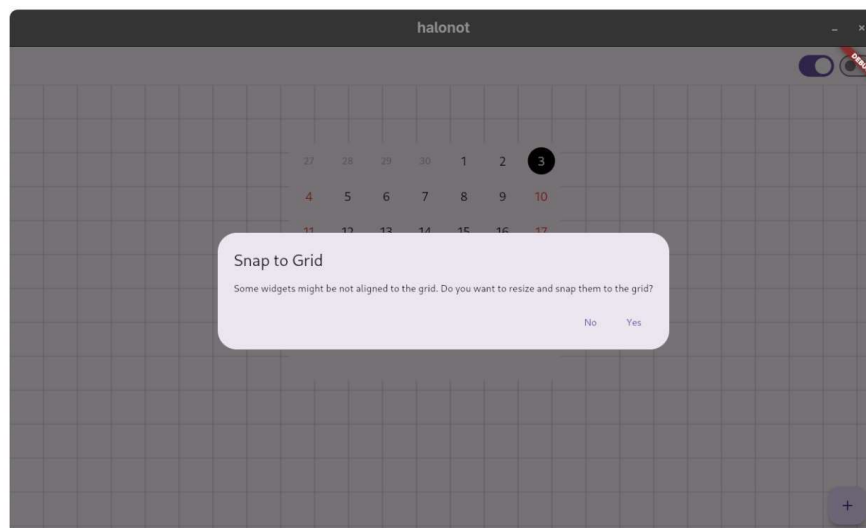


Рис. 4. Предупреждение при включении размещения по сетке.

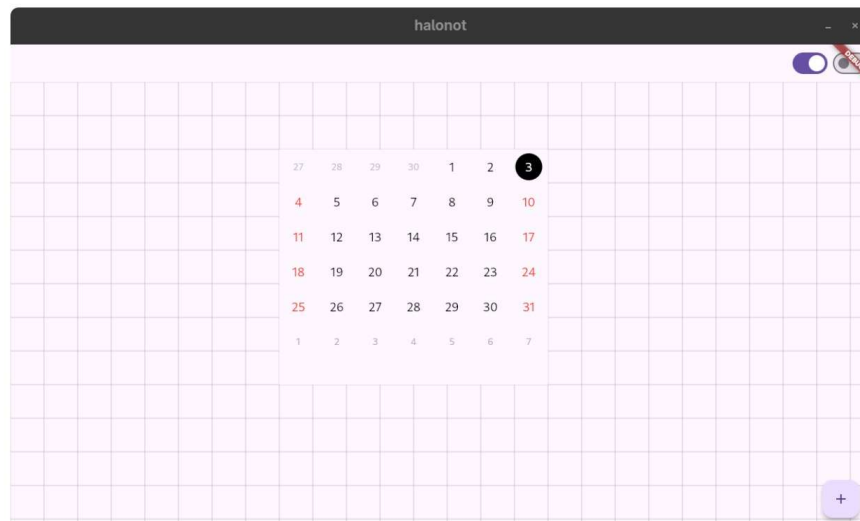


Рис. 5. Виджет успешно помещён на новую позицию.

На данный момент успешно проведены следующие работы:

- Спланирована и создана архитектура приложения.
- Созданы базовые сущности и классы, необходимые для функционирования приложения.
- Созданы основные виджеты.

Планируется разработка следующих возможностей:

- создания и добавления новых виджетов (окно отображения пользовательских HTML-документов, окно просмотра и взаимодействия с трёхмерным объектом, окно считывания рукописного ввода или рисования)
- кастомизации виджетов (настройка фона и прозрачности, а также цвета его границ)
- изменения и экспорта параметров окна (закрепление размера окна, конфигураций, расположения и параметров сетки размещения виджетов, отображение состояния окна в формате JSON).

## Список литературы

1. Что такое модульное программирование и кому оно нужно URL:  
<https://gb.ru/blog/modulnoe-programmirovanie/> (дата обращения: 1 мая 2025 г.).
2. Модульная архитектура: что, как и почему? URL:  
<https://proglib.io/p/modulnaya-arhitektura-cto-kak-i-pochemu-2023-04-04>  
(дата обращения: 1 мая 2025 г.).
3. Описание устройства класса LongPressDraggable URL:  
<https://api.flutter.dev/flutter/widgets/LongPressDraggable-class.html> (дата обращения: 1 мая 2025 г.).
4. Описание устройства класса GestureDetector URL:  
<https://api.flutter.dev/flutter/widgets/GestureDetector-class.html> (дата обращения: 1 мая 2025 г.).

## Рецензия на статью

### ***И.А. Крамаренко. Разработка кроссплатформенного приложения-интерактивной доски***

Рецензент: Пономарев Дмитрий Романович

В современных условиях трансформации растёт спрос на адаптивные интерфейсы, способные эффективно работать как в персонализированных средах, так и в публичных системах). Такие решения должны сочетать гибкость настройки, интуитивность взаимодействия и кроссплатформенность, что особенно актуально в контексте разнообразия устройств (смартфоны, планшеты, стационарные терминалы) и сценариев использования (образование, ритейл, smart-офисы). Предложенная автором статья направлена на создание модульного интерфейса на базе Flutter, объединяющего функции органайзера и киоска. Актуальность работы подкрепляется широкой распространенностью использования Flutter в кроссплатформенной разработке (по данным Google Play, более 500 тыс. проектов на Flutter было опубликовано в 2024 году). Автор демонстрирует реализацию базовых виджетов (календарь) и механизмов их взаимодействия (перемещение, изменение размера), что подтверждает практическую значимость исследования. Однако работа содержит ряд методологических и структурных недочётов, которые требуют устранения для соответствия критериям научной публикации.

Рекомендуется выполнить следующие правки:

- В разделе «Ключевые элементы приложения» указано: «виджеты можно располагать... без наложения друг на друга». Однако алгоритм предотвращения коллизий не описан. Требуется уточнить, реализована ли проверка пересечений на этапе перемещения или только при активации сетки.
- Термин «слой бизнес-логики» введён без пояснений. Рекомендуется добавить примеры операций, которые он выполняет (например, валидация содержимого виджетов).

- В разделе «Цель разработки» дублируется информация из введения. Следует объединить эти части, добавив чёткую постановку задачи: «Разработать кроссплатформенный интерфейс с поддержкой [конкретных функций] для [целевой аудитории]».

- Уточните значение «кастомизации» (раздел «Планируется разработка»): какие параметры настраиваются (шрифты, анимации)?

- Желательно уточнить схему архитектуры, описать поток данных между слоями. Хотелось бы подробнее узнать об инструментах управления состоянием, какие механизмы для этого используются.

- Нельзя не отметить малое количество литературных источников. Особенно беспокоит тот факт, что среди них нет статей из журналов, записей конференций или иной научной литературы. Для публикации хотелось бы видеть в списке хотя бы несколько подобных работ, что подтвердило бы серьезность подхода автора к решению поставленной задачи.

Статья затрагивает перспективное направление, но требует некоторой доработки. На текущем этапе работа не может быть рекомендована к публикации из-за методологических пробелов (отсутствие подробной схемы архитектуры, неописанные алгоритмы) и недостаточной экспериментальной базы. После устранения замечаний (добавление подробного описания алгоритмов и/или визуализаций) исследование приобретёт научную ценность и может быть пересмотрено.

# Работа с графикой в Kotlin Multiplatform

*Д.А. Березенцев*

*Южный федеральный университет, Ростов-на-Дону*

**Аннотация:** В статье рассматриваются проблемы работы с графикой в Kotlin Multiplatform и способы их решения. Анализируются подходы: использование Skiko для 2D-графики и Vulkan API для кроссплатформенного 3D-рендеринга. Статья будет полезна разработчикам КМР, работающим с графикой.

**Ключевые слова:** kotlin multiplatform, графика, рендеринг, vulkan api, skiko, кроссплатформенность, производительность, opengl, metal, directx, moltenvk, шейдеры, gpu, оптимизация, многопоточность

## Введение

Разработка кроссплатформенных приложений с использованием Kotlin Multiplatform (далее КМР) — это мощный подход, позволяющий разделять бизнес-логику между различными платформами, включая Android, iOS, desktop и даже web. Однако, когда речь заходит о графике, разработчики сталкиваются с рядом сложностей, связанных с различиями в графических API, ограничениями производительности и отсутствием унифицированных решений. В этой статье мы разберём ключевые проблемы работы с графикой в КМР, возможные пути их решения и роль Vulkan API в создании эффективных кроссплатформенных графических приложений.

## Современное состояние графики в КМР

На сегодняшний день Kotlin Multiplatform предлагает ограниченный набор инструментов для работы с графикой. В то время как для пользовательских интерфейсов существуют такие решения как Compose Multiplatform, обеспечивающие единый API для различных платформ, аналогичного стандартизированного подхода к графическому рендерингу не существует. Основная сложность заключается в фундаментальных различиях графических API на разных платформах:

- Android традиционно использует OpenGL ES, но постепенно переходит на Vulkan
- iOS/macOS полностью полагаются на Metal

- Windows преимущественно работает с DirectX, хотя и поддерживает Vulkan
- Linux использует Vulkan и OpenGL
- Веб-платформы опираются на WebGL и новый стандарт WebGPU

Такая фрагментация создаёт значительные сложности для разработчиков, вынужденных либо создавать отдельные реализации для каждой платформы, либо искать компромиссные решения, жертвуя производительностью или функциональностью.

## **Основные проблемы разработки графики в КМР**

### **1. Проблема разнородности графических API**

Главным препятствием для создания унифицированного графического стека в КМР является отсутствие единого стандарта для работы с GPU.

Каждый из существующих API имеет свои особенности:

- OpenGL/OpenGL ES - устаревающий, но широко поддерживаемый стандарт с высокоуровневой абстракцией
- Metal - оптимизированный исключительно для архитектуры Apple, с уникальной системой управления памятью
- DirectX - доминирующий на Windows, но закрытый для других платформ
- Vulkan - кроссплатформенный низкоуровневый API с высокой степенью контроля
- WebGL/WebGPU - веб-ориентированные решения с ограниченным доступом к оборудованию

Разработчикам приходится либо создавать сложные системы абстракции, либо поддерживать несколько независимых реализаций, что значительно увеличивает сложность проекта.

### **2. Проблема производительности**



Графические приложения предъявляют особые требования к производительности, что создаёт дополнительные сложности в контексте KMP:

- Накладные расходы межъязыкового взаимодействия между Kotlin и нативными API
- Различия в архитектуре GPU (tile-based на мобильных устройствах и immediate mode на ПК)
- Ограничения многопоточности, особенно на мобильных платформах
- Особенности управления памятью в разных API

Эти факторы затрудняют создание производительных решений, одинаково эффективно работающих на всех платформах.

### 3. Проблема инструментов разработки

Отсутствие единой экосистемы инструментов для отладки и профилирования усложняет процесс разработки:

- Разные системы отладки для каждой платформы (AGI для Android, Xcode для iOS и т.д.)
- Несогласованность в сообщениях об ошибках между API
- Отсутствие унифицированных метрик производительности
- Сложности с воспроизведением проблем на разных платформах

## Подходы к работе с графикой в KMP

Несмотря на сложности, есть несколько способов интегрировать графику в KMP-приложения.

1. Использование multiplatform-библиотек. Некоторые библиотеки, такие как Skiko (Skia для Kotlin), предоставляют кроссплатформенный API для рисования. Skia — это графическая библиотека, лежащая в основе Chrome, Flutter и Android Canvas. Skiko позволяет использовать её в KMP, абстрагируясь от нативных API. Это хороший вариант для 2D-

графики, но для сложного 3D-рендеринга возможностей может не хватить.

2. Абстракция над нативными API. Можно создать общий Kotlin-интерфейс для рендеринга и реализовать его отдельно для каждой платформы. Например, общий модуль определяет методы `drawMesh()` или `setShader()`, а в platform-специфичных модулях они вызывают OpenGL, Metal или DirectX. Это даёт полный контроль над графикой, но требует значительных усилий для поддержки.
3. Использование Vulkan API. Vulkan — это современный низкоуровневый графический API, который поддерживается на Android, Linux, Windows и даже macOS (через MoltenVK, трансляцию Vulkan в Metal). Его главные преимущества — кроссплатформенность, высокая производительность и минимальные накладные расходы.

## **Vulkan как потенциальное решение**

Vulkan мог бы стать идеальным решением для КМР, если бы не несколько нюансов. Во-первых, на iOS нет нативной поддержки Vulkan, и приходится полагаться на MoltenVK, что добавляет сложности. Во-вторых, Vulkan сложен в освоении и требует больше шаблонного-кода по сравнению с OpenGL или Metal. Однако если эти трудности преодолимы, Vulkan даёт значительные преимущества:

- Единый код для нескольких платформ — большая часть рендерера может быть написана на Kotlin с использованием Vulkan, а platform-специфичные части сводятся к инициализации контекста.
- Производительность — Vulkan позволяет тонко настраивать рендеринг, уменьшая нагрузку на CPU за счёт многопоточности и явного управления ресурсами.
- Современные возможности — поддержка ray tracing, compute shaders и других продвинутых техник.

Для работы с Vulkan в KMP можно использовать vulkan-обёртки на Kotlin, такие как vulkan-kotlin или LWJGL (если проект включает JVM-платформы). Также можно написать общий модуль на Kotlin/Native, взаимодействующий с C-библиотеками Vulkan.

Преимущества Vulkan в контексте KMP:

1. Широкая поддержка платформ:

- Нативная поддержка на Android (API 24+)
- Полноценная работа на Windows и Linux
- Возможность использования на macOS/iOS через MoltenVK

2. Единая кодовая база:

- Возможность использовать один основной API для большинства платформ
- Снижение затрат на поддержку нескольких реализаций

3. Современные возможности:

- Низкоуровневый контроль над GPU
- Поддержка передовых технологий (ray tracing, mesh shading)
- Эффективное управление памятью и потоками

4. Инструментарий:

- Унифицированные средства отладки (Validation Layers)
- Кроссплатформенные профилировщики (RenderDoc)

## **Практический пример: настройка Vulkan в KMP**

Допустим, мы хотим создать простой рендерер на Vulkan, работающий на Android, iOS и desktop.

1. Общий модуль определяет интерфейсы для шейдеров, текстур и мешей.

2. Android-модуль использует `vkCreateInstance` из Android NDK.
3. Desktop-модуль (Linux/Windows) вызывает Vulkan через системные библиотеки.
4. iOS-модуль подключает MoltenVK как динамическую библиотеку.

Основная сложность — управление памятью и контекстом Vulkan, так как Kotlin/Native имеет свои особенности работы с нативными ресурсами.

## Заключение

Графика в Kotlin Multiplatform остаётся сложной задачей, но не безнадёжной. Использование Skiko подходит для простых случаев, а Vulkan предлагает мощный, хоть и сложный, путь для высокопроизводительного рендеринга. В будущем, с развитием КМР и появлением новых библиотек, работа с графикой может стать проще, но сегодня разработчикам приходится комбинировать разные подходы для достижения кроссплатформенности без потерь в качестве. Vulkan, несмотря на свою сложность, выглядит наиболее перспективным решением, особенно если проект требует современных графических возможностей на нескольких платформах.

## Литература

1. Khronos Group. Vulkan Documentation. [Электронный ресурс].  
URL: <https://www.khronos.org/vulkan/>
2. JetBrains. Kotlin Multiplatform documentation. [Электронный ресурс].  
URL: <https://kotlinlang.org/docs/multiplatform.html>
3. MoltenVK. GitHub repository. [Электронный ресурс].  
URL: <https://github.com/KhronosGroup/MoltenVK>
4. JetBrains. "Skiko: Skia for Kotlin". [Электронный ресурс].  
URL: <https://blog.jetbrains.com/kotlin/2021/05/skiko-skia-for-kotlin/>
5. NVIDIA Developer Blog. "Vulkan on Mobile". [Электронный ресурс].  
URL: <https://developer.nvidia.com/blog/tag/vulkan/>

## **Резюме**

Статья посвящена исследованию подходов к разработке графических приложений в среде Kotlin Multiplatform (KMP). Автор анализирует сложности, связанные с использованием графических API на разных платформах, и предлагает два решения: библиотеку Skiko для 2D-графики и Vulkan API для кроссплатформенного 3D-рендеринга. Работа ориентирована на разработчиков, интересующихся созданием графики в KMP, и демонстрирует практические аспекты применения этих технологий.

### **Положительные аспекты**

5. Статья затрагивает важную и современную проблему разработки кроссплатформенных приложений с использованием графики в KMP. Учитывая растущую популярность KMP, данная работа своевременна и полезна для целевой аудитории — разработчиков, сталкивающихся с графическими задачами.

6. Автор подробно описывает основные проблемы (разделы «Основные проблемы разработки графики в KMP» и «Современное состояние графики в KMP»), такие как фрагментация API и ограничения производительности, а также предлагает конкретные решения — Skiko и Vulkan API.

7. В разделе «Практический пример: настройка Vulkan в KMP» приведён конкретный сценарий интеграции Vulkan, что помогает читателям понять, как применять предложенные решения на практике.

8. Использование авторитетных источников (например, документация Khronos Group, JetBrains и NVIDIA) повышает достоверность изложения и даёт читателям возможность углубить знания по теме.

### **Критические замечания**

4. В разделе «Практический пример: настройка Vulkan в KMP» описан общий процесс, но не приведены конкретные фрагменты кода или ссылки на них. Уточните хотя бы один пример вызова для наглядности.

5. Список литературы включает полезные ресурсы, такие как документацию Vulkan и Skiko, но в тексте статьи отсутствуют явные указания на них (например, "см. [1]"). Рекомендуется добавить ссылки в

соответствующих разделах, чтобы читатели могли быстро обратиться к источникам.

6. В разделе "Основные проблемы разработки графики в КМР" автор упоминает OpenGL/OpenGL ES как "устаревающий стандарт", что может ввести в заблуждение, учитывая широкое использование OpenGL ES на мобильных платформах. Рекомендуется уточнить, что OpenGL ES остается актуальным в определенных контекстах, несмотря на переход к Vulkan.

### **Мелкие правки**

- Фраза из раздела «Заключение»: «Графика в Kotlin Multiplatform остаётся сложной задачей, но не безнадёжной» звучит разговорно на фоне формального тона остального текста. Переформулируйте, например: «Работа с графикой в КМР остаётся сложной, но имеет перспективы для оптимизации».
- Также фраза «Vulkan мог бы стать идеальным решением для КМР, если бы не несколько нюансов» звучит разговорно, она может быть заменена на: «Хотя Vulkan и предлагает значительные преимущества для Kotlin Multiplatform, его внедрение затруднено такими факторами, как ...»

### **Рекомендации редактору**

Статья представляет собой ценный обзор современных подходов к работе с графикой в КМР, сочетая актуальность темы с практической применимостью. Предложенные замечания необязательны, так как работа уже самодостаточна. Рекомендую принять статью к публикации, оставив предложенные улучшения на усмотрение автора.

# АНАЛИЗ МОДИФИКАЦИЙ АРХИТЕКТУРЫ GAN ДЛЯ ЗАДАЧ ИНПЕЙНТИНГА

## Введение

Инпейнтингом (inpainting) называется процесс восстановления отсутствующих частей изображения с целью получения визуально правдоподобного результата. Наиболее часто применяется для реставрации повреждённых изображений, а также удаления (реже генерации) объектов на переднем плане. Существуют различные подходы к решению данных задач, как детерминированные, так и на основе нейронных сетей.

Архитектура генеративно-состязательных сетей [2] является на данный момент одной из часто используемых для решения задач компьютерного зрения. Однако существует сравнительно мало статей, использующих данный подход для инпейнтинга [1], а результаты зачастую хуже, чем при других способах восстановления изображений [4].

Для решения проблемы низкого качества инпейнтинга с использованием генеративно-состязательных сетей были предложены различные модификации базовой архитектуры. Рассмотрим несколько статей с представленным исходным кодом и с помощью различных метрик проверим качество результатов.

## Использование признаковых представлений

Одна из идей лежащих в основе успешных модификаций GAN для задач инпейнтинга – работа не только с самим изображением, но и с различными признаковыми представлениями. Появляется вторая пара генератор-дискриминатор, целью которой является восстановление тех или иных ключевых признаков в удалённой области. Результат генерации передаётся как дополнительная информация на вход основному генератору вместе с исходным изображением.

В статье [3] представлена архитектура EdgeConnect – в ней в качестве признакового представления используется бинарная карта границ (полученная с помощью детектора Canny). Первый генератор восстанавливает границы, второй – исходное изображение.

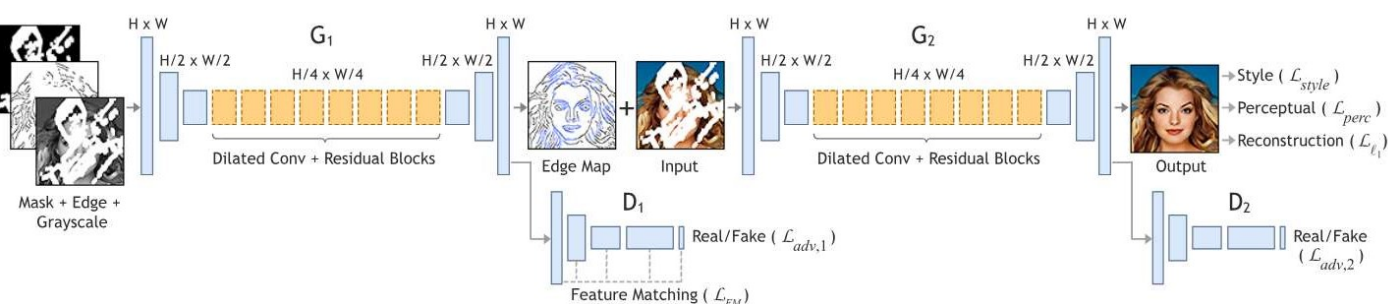


Рис. 1. Архитектура EdgeConnect

Схожим образом работает архитектура, представленная в работе [6], однако в ней вместо границ используются локальные бинарные шаблоны (local binary patterns, LBP). В данной работе используются генераторы с пропускными соединениями (U-net) между соответствующими downsampling и upsampling слоями, также в главном генераторе (возвращающем изображение) добавлен слой пространственного внимания (spatial attention), извлекающий признаки патчей из известной и восстанавливаемой областей изображения. Патчи из восстанавливаемой области обновляются с использованием взвешенного среднего наиболее похожих патчей. Это, а также само по себе использование локальных бинарных шаблонов должно улучшить качество восстановления различных текстур.

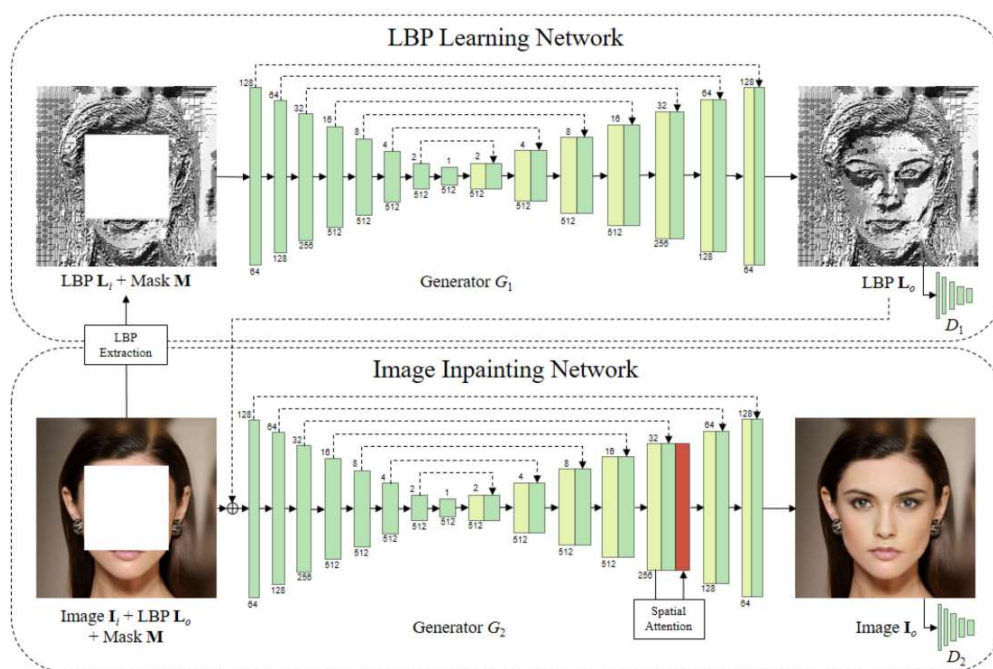


Рис. 2. Архитектура на основе LBP

## Метрики

Для получения достоверной оценки качества предложенных методов используем метрики PSNR и SSIM для попарного сравнения исходных ( $X$  в формулах) и восстановленных ( $Y$  в формулах) изображений размера  $m$  на  $n$ .

**Peak signal-to-noise ratio (PSNR)** – пиковое отношение сигнала к шуму, рассчитывается по формуле



$$PSNR(X, Y) = 10 \log_{10} \left( \frac{MAX^2}{MSE(X, Y)} \right)$$

Где MAX – максимально возможное значение интенсивности пикселя (255 в случае кодирования восьмибитным числом), а MSE – среднеквадратическая ошибка, рассчитываемая по приведённой ниже формуле:

$$MSE(X, Y) = \frac{1}{mn} \sum_{i=1}^m \sum_{j=1}^n (X_{ij} - Y_{ij})^2$$

Более высокие значение PSNR указывают на меньшее искажение восстановленного изображения. Значения 40 и больше соответствуют отличному качеству изображения, 30–40 хорошему, 20–30 удовлетворительному, менее 20 – плохому качеству. PSNR чувствителен к различным пиксельным искажениям, в том числе небольшим, из-за этого не всегда коррелирует с человеческим восприятием визуального качества.

**Structural similarity index measure (SSIM)** – индекс структурного сходства. Был предложен в статье [5] как более соответствующий физиологии человеческого восприятия чем предшественники. Рассчитывается на основе среднего ( $\mu_X$ ), дисперсии ( $\sigma_X$ ) и ковариации ( $\sigma_{XY}$ ) для каждого окна (по умолчанию размера 11x11), затем значение усредняется чтобы получить результат для всего изображения.

$$SSIM(X, Y) = \frac{(2\mu_X\mu_Y + c_1)(2\sigma_{XY} + c_1)}{(\mu_X^2 + \mu_Y^2 + c_1)(\sigma_X^2 + \sigma_Y^2 + c_2)}$$

Таким образом значение 1 соответствует идеальному качеству восстановления, а 0 – отсутствию структурного сходства. В редких случаях может быть обнаружена антикорреляция, тогда значения метрики будут отрицательными. Удовлетворительное качество восстановления начинается со значения 0.84, хорошее со значения 0.88, отличное со значения 0.95.

Для датасета places и нерегулярных масок скрывающих от десяти до двадцати процентов площади были получены следующие результаты:

| Метрика | EdgeConnect | На основе LBP |
|---------|-------------|---------------|
| PSNR    | 27.932      | 28.882        |
| SSID    | 0.919       | 0.930         |

Представленные архитектуры действительно справляются с поставленной задачей: достигается удовлетворительный уровень зашумленности (согласно PSNR) и сравнительно высокое структурное сходство с восстановленного изображения с исходным (согласно SSID). Архитектура на основе u-net и локальных бинарных шаблонов показывает лучшие результаты.

## Список литературы

1. Elharrouss O., Almaadeed N., Al-Maadeed S. et al. *Image inpainting: A review*.
2. Goodfellow I. J., Pouget-Abadie J., Mirza M. et al. *Generative Adversarial Networks*.
3. Nazeri K., Ng E., Joseph T. et al. *EdgeConnect: Generative Image Inpainting with Adversarial Edge Learning*.
4. Patel V. S., Agrawal K., Baraheem S. S. et al. *Image de-photobombing benchmark*.
5. Wang Z., Bovik A. C., Sheikh H. R. et al. *Image Quality Assessment: From Error Visibility to Structural Similarity*.
6. Wu H., Zhou J., Li Y. *Deep Generative Model for Image Inpainting with Local Binary Pattern Learning and Spatial Attention*.

#### Рецензия на статью

#### «АНАЛИЗ МОДИФИКАЦИЙ АРХИТЕКТУРЫ GAN ДЛЯ ЗАДАЧ ИНПЕЙТИНГА»

#### Резюме

Рецензируемая статья действительно имеет актуальную тематику. На сегодняшний день алгоритмы глубокого обучения активно используются в задачах восстановления и замены отсутствующих частей изображения. В статье приводятся две архитектуры генеративно-сопоставительных сетей, а также вводятся метрики для сравнения эффективности этих подходов между собой. Тем не менее, нельзя назвать рецензируемую статью полноценным анализом, так как описание принципов работы упомянутых сетей приведено недостаточно подробно, а количество метрик не позволяет сделать цельные выводы о сравнительной эффективности двух разных подходов.

#### Критические замечания

После прочтения рецензируемой статьи возникли следующие замечания:

1. Введение статьи плохо отражает эффективность использования нейронных сетей в задачах восстановления изображений. Необходимо привести больше фактических данных, в которых методы, использующие глубокое обучение сравниваются с предшествующими методами;
2. В тексте статьи не раскрываются общие сведения о работе генеративно-состязательных сетей (*Generative adversarial network* - GAN). Поэтому, говоря о модификациях архитектуры GAN следует подробнее рассказать об общих принципах работы;
3. Модификации GAN описаны недостаточно подробно. Для лучшего понимания следует привести наглядные примеры работы каждого алгоритма с детальным пояснением ключевых моментов;
4. Использование всего лишь двух метрик не помогает всесторонне проанализировать эффективность нейронных сетей. Также используемые метрики показывают незначительные отличия в результатах чтобы говорить о достаточном сравнении этих архитектур. Необходимо подобрать большее количество метрик, которые бы указывали на различия (или их отсутствие) между двумя подходами;
5. Отсутствие описания данных, которые были использованы для сравнения ставит под сомнение выводы статьи. Следует подробно рассказать про тип данных и размерность датасета, а также использовать примеры из него для демонстрации работы нейронных сетей.

Помимо вышесказанного, имеются замечания касательно оформления статьи:

1. Отсутствие аннотации с кратким содержанием статьи;
2. Отсутствие списка ключевых слов;
3. Отсутствие фамилии и инициалов автора в шапке статьи;
4. Отсутствие перечисления объекта, предмета и метода исследования;
5. Отсутствие секции с заключением и выводами.

## **Мелкие правки**

Мелкие правки относятся не к содержанию статьи, а к её оформлению:

1. Следует добавить название и нумерацию таблицы;
2. Следует добавить ссылки на рисунки в тексте, а также более подробно пояснить обозначения на самих рисунках;
3. Следует добавить нумерацию для приводимых формул;
4. Для упомянутого набора данных *places* следует привести ссылку для более удобной навигации;
5. Для материалов в секции «Список литературы» следует перечислять в порядке их упоминания, а также необходимо добавить ссылки на статьи в открытом доступе.

## **Рекомендации редактору**

Рецензируемую статью взять на доработку и принять только после исправления вышеперечисленных замечаний. Необходимо дополнение теоретической части статьи, переработка методологии анализа, расширение секции с результатами анализа и выводом. В работе должны быть соблюдены правила оформления научных статей.

*Рецензент:*

*Студент 1-го курса магистратуры*

*1 группа, ФИИТ*

*Нуреев Андрей Олегович*

# Анализ алгоритмов симуляции жидкости в контексте видеоигр

*Т. Р. Мамедов<sup>1</sup>*

*<sup>1</sup>Южный федеральный университет, Ростов-на-Дону*

**Аннотация:** Аннотация на русском языке представляет собой краткое точное изложение содержания статьи, включающее основные фактические сведения и выводы описываемой работы, и рассчитана на самый широкий круг читателей.

**Ключевые слова:** 10-15 ключевых слов, именительный падеж, единственное число, строчными буквами, через запятую.

## Введение

На данный момент доступны более 500 различных публикаций, которые затрагивают темы симуляции, но большинство из них фокусируются на точности симуляции. Исследуемые методы и их модификации по прежнему не подходят для программ реального времени в трёх измерениях (видеоигры, интерактивные визуализации и др.).

Существуют различные сильно упрощенные алгоритмы основанные на:

- Клеточных автоматах (Cellular Automata [1])
  - (примеры: Minecraft, Noita(двумерные))
- Высотных картах (Heightmap / Heightfield [2])
- Трохоидальные волны, Быстрое преобразование Фурье (Trochoidal Waves / Gerstner Wave [2], Fast Fourier Transform (FFT) [3])
  - (примеры: Sea of Thieves)

В этой статье будут рассматриваться недостатки всех вышеперечисленных методов, а также более точные алгоритмы (и их быстродействие), которые пока что не используются в видеоиграх, такие как:

- Smoothed-Particle Hydrodynamics (далее SPH) [4]
- Particle-In-Cell (далее PIC) [5]
- Fluid-Implicit Particles (далее FLIP) [5]
- Material Point Method (далее MPM) [6]
- Moving Least Squares Material Point Method (далее MLS-MPM) [7]

## Упрощённые алгоритмы

Клеточные автоматы дают дешёвую и стабильную симуляцию. Алгоритм работает на единственной глобальной сетке вокселей (или пикселей) и каждый кадр симуляции сравнивает соседствующие клетки, чтобы определить следующее состояние. У клеток жидкости нет ни скорости, ни массы, поэтому данный алгоритм подходит только для полу-статичных объёмов воды без взаимодействия с физикой.

Высотные карты позволяют очень точно симулировать поверхность воды. Алгоритм работает на сетке, которая обычно напрямую связана с визуальной геометрией, где каждая точка имеет вертикальную позицию и скорость и связана «пружинами» с несколькими соседями. С помощью высотных карт легко можно симулировать поверхность воды в аквариуме, но посчитать волны на площади всего океана не выйдет — размер симулируемой поверхности ограничен. В видеоиграх чаще используются только для декораций.

Трохоидальные волны отлично подходят для симуляции океанов. Алгоритм работает складывая множество привязанных ко времени синусоидальных волн для каждой точки, поэтому состояние всей системы легко узнать зная начальные свойства и время. Симуляционный домен практически не ограничен и детерминированно считается в любой точке, также, легко можно посчитать силы влияющие на тот или иной объект/судно.

Сравнивать между собой подобные алгоритмы бессмысленно, так как они работают на абсолютно разных уровнях объёмов симуляции. Все эти алгоритмы довольно просты в реализации, но имеют общий главный недостаток — жидкость привязана к какому-то симуляционному домену. То есть, её нельзя зачерпнуть, вылить на водяное колесо и т. д., подобные взаимодействия могут очень пригодиться в видеоиграх в жанрах песочницы, автоматизации, и др..

## Точные алгоритмы

Все описанные здесь алгоритмы используют сетку, как промежуточный шаг, а также частицы имеющие позицию, скорость (и где-то — массу). Симуляция не имеет полной привязки к сетке благодаря частицам — жидкость можно черпать, создавать на ней волны, наполнять произвольные резервуары и т.д.. Данные алгоритмы часто предоставляются в различных пакетах программ, такие как Houdini, Blender и др..

PIC и FLIP практически идентичны, но FLIP даёт более реалистичные результаты с минимумом дополнительных затрат. Также с MPM и MLS-MPM, MLS-MPM дешевле MPM в несколько раз и легче параллелизуется, отличить их на глаз тяжело. В данной статье не будут указываться детали ухудшенных алгоритмов.

Ниже в таблицы приведены для сравнения различные алгоритмы, их параметры (дающие похожую точность) и результирующий средний объём жидкости для времени исполнения в 5 мс (см. таблица №1).

Таблица №1 1

| Алгоритм           | Параметры              | Объём (м <sup>3</sup> ) |
|--------------------|------------------------|-------------------------|
| SPH                | Particle radius: 0.1m; | 69                      |
| FLIP               | Grid cell size: 0.2m;  | 105                     |
| MLS-MPM            | Grid cell size: 0.2m;  | 388                     |
| SPH (parallel)     | Particle radius: 0.1m; | 121                     |
| FLIP (parallel)    | Grid cell size: 0.2m;  | 294                     |
| MLS-MPM (parallel) | Grid cell size: 0.2m;  | 895                     |

Как можно заметить, MLS-MPM работает значительно быстрее других алгоритмов, но объёма всё равно недостаточно для видеоигровых задач. В данном исследовании был установлен бюджет в 5 мс, что соответствует 200 кадрам в секунду в изоляции, но стоит не забывать, что процессору требуется вычислять всё остальное: часть отрисовки, физику твёрдых тел, анимации и др..

### Литература

1. Simple Fluid Simulation With Cellular Automata // w-shadow.com URL: <https://w-shadow.com/blog/2009/09/01/simple-fluid-simulation/> (дата обращения: 01.05.2025).
2. Matthias Müller-Fischer, Fast Water Simulation for Games Using Height Fields // GDC2008 URL: <https://ubm-twideo01.s3.amazonaws.com/o1/vault/gdc08/slides/S6509i1.pdf> (дата обращения: 01.05.2025).
3. GPU Gems // nvidia.com URL: <https://developer.nvidia.com/gpugems/gpugems/part-i-natural-effects/chapter-1-effective-water-simulation-physical-models> (дата обращения: 01.05.2025).
4. Smoothed Particle Hydrodynamics // Wikipedia URL: [https://en.wikipedia.org/wiki/Smoothed-particle\\_hydrodynamics](https://en.wikipedia.org/wiki/Smoothed-particle_hydrodynamics) (дата обращения: 01.05.2025).
5. PIC/FLIP Fluid Simulation Using A Block-Optimized Grid // DIVA Portal URL: <https://www.diva-portal.org/smash/get/diva2:441801/FULLTEXT01.pdf> (дата обращения: 01.05.2025).
6. Material Point Method // Wikipedia URL: [https://en.wikipedia.org/wiki/Material\\_point\\_method](https://en.wikipedia.org/wiki/Material_point_method) (дата обращения: 01.05.2025).
7. Yuanming Hu, Yu Fang, Ziheng Ge, Ziyin Qu, Yixin Zhu, Andre Pradhana, Chenfanfu A Moving Least Squares Material Point Method with Displacement Discontinuity and Two-Way Rigid Body Coupling // ACM Transactions on Graphics. - 2018. – №37.



## References

1. Simple Fluid Simulation With Cellular Automata // w-shadow.com URL: <https://w-shadow.com/blog/2009/09/01/simple-fluid-simulation/> (date of access: 01.05.2025).
2. Matthias Müller-Fischer, Fast Water Simulation for Games Using Height Fields // GDC2008 URL: <https://ubm-twideo01.s3.amazonaws.com/o1/vault/gdc08/slides/S6509i1.pdf> (date of access: 01.05.2025).
3. GPU Gems // nvidia.com URL: <https://developer.nvidia.com/gpugems/gpugems/part-i-natural-effects/chapter-1-effective-water-simulation-physical-models> (date of access: 01.05.2025).
4. Smoothed Particle Hydrodynamics // Wikipedia URL: [https://en.wikipedia.org/wiki/Smoothed-particle\\_hydrodynamics](https://en.wikipedia.org/wiki/Smoothed-particle_hydrodynamics) (date of access: 01.05.2025).
5. PIC/FLIP Fluid Simulation Using A Block-Optimized Grid // DIVA Portal URL: <https://www.diva-portal.org/smash/get/diva2:441801/FULLTEXT01.pdf> (date of access: 01.05.2025).
6. Material Point Method // Wikipedia URL: [https://en.wikipedia.org/wiki/Material\\_point\\_method](https://en.wikipedia.org/wiki/Material_point_method) (date of access: 01.05.2025).
7. Yuanming Hu, Yu Fang, Ziheng Ge, Ziyin Qu, Yixin Zhu, Andre Pradhana, Chenfanfu A Moving Least Squares Material Point Method with Displacement Discontinuity and Two-Way Rigid Body Coupling // ACM Transactions on Graphics. - 2018. – №37.

## **Рецензия на статью Мамедова Т.Р. «Анализ алгоритмов симуляции жидкости в контексте видеоигр»**

### **Резюме**

Статья посвящена анализу алгоритмов симуляции жидкости в видеоиграх. Автор разделяет методы на упрощённые (клеточные автоматы, высотные карты, трохоидальные волны) и точные (SPH, PIC, FLIP, MPM, MLS-MPM). Упрощённые алгоритмы описаны как ограниченные в динамическом взаимодействии с окружающей средой, а точные — как слишком ресурсоёмкие для текущих игровых приложений. В работе приведена сравнительная таблица, демонстрирующая объём симулируемой жидкости за 5 мс, что подчёркивает производительность алгоритмов. Основной вывод: даже самый быстрый метод (MLS-MPM) не удовлетворяет требованиям современных видеоигр.

### **Критические замечания**

В таблице №1 приведены данные о производительности алгоритмов, но не указано, на каком оборудовании проводились замеры и характеристики этого оборудования. Это очень важно для воспроизведения результатов. Обзор алгоритмов носит поверхностный характер. Например, описание SPH, FLIP и MLS-MPM не раскрывает их базовые принципы. Рекомендуются добавить краткие пояснения о механизмах работы методов, чтобы статья была полезна читателям без глубоких знаний в области компьютерной графики. Таблица №1 ограничивается объёмом жидкости за 5 мс. Утверждение о неприменимости точных алгоритмов в видеоиграх не подкреплено конкретными примерами или ссылками на исследования. Автору стоит привести данные о попытках интеграции таких методов в игровые движки и их результатах. Для полноты картины желательно включить дополнительные параметры, такие как стабильность симуляции или требования к памяти, что помогло бы оценить применимость алгоритмов в разных сценариях.

### **Мелкие правки**

Аннотация и ключевые слова к статье не приведены. Литература содержит ссылки на Wikipedia и GDC, но мало академических источников (например, последних статей по MLS-MPM). Термин «симуляционный домен» требует пояснения при первом упоминании. В таблице №1 разный формат определения единиц измерения: в параметрах английская m в каждой строке, а в объёме русская м<sup>3</sup> только в заголовке таблицы.

### **Рекомендация**

Статья представляет ценность для специалистов в области разработки видеоигр и компьютерной графики, но нуждается в доработке. Рекомендуются

дополнить её анализом визуальной точности, детальным описанием алгоритмов и примерами их применения. После внесения изменений статья может быть принята к публикации.

**Рецензент:** Рамазанов А.М.

# Обнаружение рукописных подписей на изображениях документов

*Д. Р. Пономарев*

*<sup>1</sup>Южный федеральный университет, Ростов-на-Дону*

**Аннотация:** В данной статье рассматривается применение нейронных сетей глубокого обучения для решения задачи детекции и идентификации рукописной подписи выполненных на бумажных документах. Приводятся примеры подходящих для обучения моделей наборов данных и способы построения на их основе собственного датасета с применением аугментаций. Рассматривается применение нейросетей–детекторов для обнаружения подписей на изображениях и приводятся полученные после их обучения метрики качества.

**Ключевые слова:** распознавание рукописной подписи, машинное обучение, классификация подписи, нейронная сеть, глубокое обучение, обнаружение объектов, распознавание текста, обработка изображений, аугментация данных, сверточные нейросети, одноэтапные детекторы.

## Введение

Рукописные подписи остаются ключевым элементом аутентификации документов в юридической, финансовой и административной сферах [1, 2]. С развитием цифровизации процессов актуальность автоматического обнаружения и верификации подписей на изображениях документов значительно возросла. Несмотря на прогресс в области компьютерного зрения, задача точной локализации рукописных подписей остается сложной из-за вариативности их расположения, стилей исполнения, а также наличия артефактов, таких как шумы, печатный текст и графические элементы [3]. Существующие публичные наборы данных зачастую ограничены в объеме и разнообразии, что снижает эффективность обучения моделей для реальных сценариев. В данной работе рассматривается подход к формированию специализированного набора данных на основе открытых источников, включая оцифрованные архивы, юридические документы и публичные базы рукописных символов. Также внимание уделяется методам предобработки и аугментации данных, что позволяет повысить устойчивость моделей к артефактам сканирования и деформациям.

Для решения задачи детекции исследуются модели семейства YOLO (You Only Look Once), сочетающие высокую скорость обработки и точность локализации объектов.

## Состав набора данных и методы обработки

Набор данных сформирован за счёт объединения нескольких публичных датасетов:

1. Signatures-XC8UP — часть коллекции Roboflow100 [4] с 368 аннотированными изображениями рукописных подписей.
2. Tobacco800 [5]. Подмножество CDIP Test Collection, включающее сканированные документы из табачной промышленности.
3. Датасет ICDAR 2009 Signature Verification Competition [6]. Включает подписи 100 авторов (подлинные) и 33 авторов (подделки). Коллекция объединяет 1953 онлайн- и 1953 оффлайн-подписей. Подделки создавались путём ручной имитации образцов (рис. 1).



Рис. 1 – Примеры изображений. Верхний ряд содержит подлинные подписи, нижний — подделки

4. GPDS Signature Data Base [7]. Данные 300 человек: по 24 подлинные и 30 поддельных подписей на каждого.

В первые два аннотированных датасета интегрированы образцы из третьего и четвёртого по следующей схеме: оригинальные подписи в документах заменялись на настоящие или поддельные подписи из второй половины наборов. Для этого соответствующие фрагменты исходных фотографий вырезались и замещались масштабированными версиями подписей из третьей и четвёртой коллекций изображений. В результате был создан комбинированный датасет, содержащий документы с размеченными как подлинными, так и фальшивыми подписями. Итоговый набор был разбит на 3 выборки: обучающую (70%), валидационную (15%) и тестовую (15%).

## **Обработка данных и методы аугментации для задач анализа документов**

Все изображения предварительно стандартизированы: выполнена автоориентация для коррекции поворотов и масштабирование для приведения к размеру  $640 \times 640$  пикселей. Для повышения качества обучения[8] и улучшения разнообразия данных применены методы аугментации, разделённые на две группы:

### **1. Пространственные преобразования**

- Фиксированный поворот на  $90^\circ$  имитирует типичные сценарии сканирования документов в альбомной или книжной ориентации.
- Случайный поворот ( $\pm 10^\circ$ ) и сдвиг добавляют вариативность в положение объектов, что учит модель распознавать документы с неточностями в выравнивании (например, при ручной съёмке).

### **2. Фотометрические корректировки**

- Изменение яркости ( $\pm 20\%$ ) помогает адаптироваться к разным условиям освещения (например, затемнённые участки или засветы).
- Стохастическое размытие, которое симулирует небольшую расфокусировку камеры или движение при съёмке.
- Гауссов шум, повышающий устойчивость к артефактам сжатия или низкому качеству сенсора.

## **Выбор модели и ее обучение**

Детекция объектов в компьютерном зрении достигла значительного прогресса, обеспечивая баланс между скоростью и точностью для таких задач, как автономное вождение и системы видеонаблюдения.

Эволюция архитектур прошла несколько этапов, и современные подходы можно разделить на категории:

- Двухэтапные детекторы на основе сверточных нейронных (Convolutional Neural Networks, далее CNN): рекурсивные модели этого семейства сначала генерируют регионы-кандидаты, затем классифицируют их [9].

- Одноэтапные детекторы: архитектуры вроде YOLO, предсказывающие координаты ограничивающей рамки и класс объекта за один проход через сеть.
- Трансформерные детекторы: модели семейства DETR, использующие механизмы внимания для end-to-end детекции.
- Гибридные архитектуры: Комбинации CNN (для эффективности) и трансформеров (для анализа контекста).

Модель YOLO, представленная в 2016 году, радикально изменила подход к детекции объектов, объединив предсказание ограничивающей рамки и классификацию в одном шаге. Это сделало YOLO идеальным для устройств с ограниченными ресурсами (например, камеры видеонаблюдения или телефоны) и задач, требующих мгновенного отклика [10]. По этой причине в данной работе было решено рассмотреть именно эту архитектуру для решения поставленной задачи. В работе анализируются архитектура YOLO11, адаптированная под специфику распознавания рукописных подписей. Обучение производилось на протяжении 20 эпох. Продолжительность составила около 40 минут. На рис.2 представлены результаты обучения в зависимости от номера эпохи, в качестве метрики для оценки качества использована mAP50 (Mean Average Precision 50).

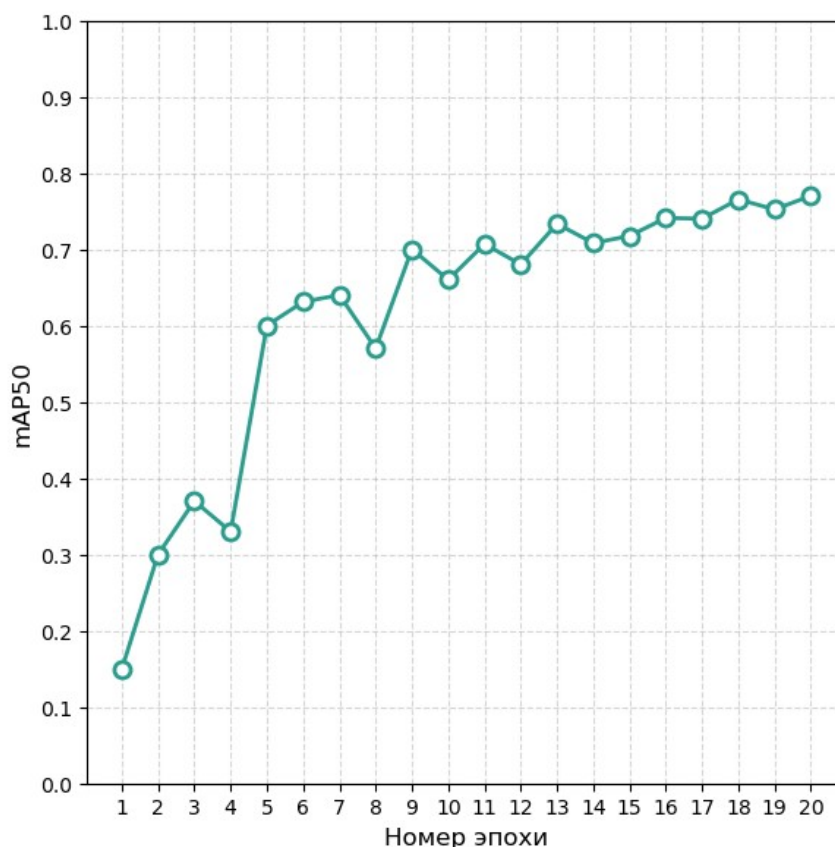


Рис. 2 – Результаты обучения

### Выводы

Предложенный метод построения системы детекции рукописных подписей демонстрирует эффективность комбинации кастомизированного набора данных и моделей YOLO. Использование аугментации на основе синтеза фонов и геометрических преобразований позволило повысить устойчивость алгоритма к вариациям входных изображений. Разработанный подход может быть масштабирован для обработки разноформатных документов в системах автоматизированного делопроизводства. Дальнейшие исследования предполагается направить на оптимизацию алгоритма для интеграции с устройствами с малой производительностью.

### Литература

23. Ю. С. Желудов. Проблемы идентификации в системах распознавания рукописных подписей // Научный журнал. 2018. №9. С. 36-37.



24. А. В. Пятаева, М. А. Мерко, В. А. Жуковская, И. А. Пиньчук, М. С. Елисеева. Распознавание рукописной подписи с применением нейронных сетей // International Journal of Advanced Studies. 2023. №13. С. 130-148.
25. Н. А. Смирнов, Э. С. Анисимова. Распознавание рукописной подписи человека с помощью нейронных сетей // Инженерный вестник Дона. 2024. №11. С. 4-9.
26. Floriana Ciaglia, Francesco Saverio Zuppichini, Paul Guerrie, Mark McQuade, Jacob Solawetz. Roboflow 100: A Rich, Multi-Domain Object Detection Benchmark. 2022.
27. D. Lewis, G. Agam, S. Argamon, O. Frieder, D. Grossman, J. Heard Building a test collection for complex document information processing // 29th Annual Int. ACM SIGIR Conference (SIGIR 2006). 2006. С. 665-666.
28. V.L. Blankers, C.E. van den Heuvel, K. Franke, L. Vuurpijl. The ICDAR 2009 Signature Verification Competition // Proceedings of the 2009 10th International Conference on Document Analysis and Recognition. 2009. С. 1403-1407.
29. Ferrer, M. A., Diaz-Cabrera, M., Morales, A Static signature synthesis: A neuromotor inspired approach for biometrics // IEEE Transactions on pattern analysis and machine intelligence. 2014. №37(3). С. 667-680.
30. Ахунджанов У. Ю., Старовойтов В. В. Предварительная обработка изображений рукописных подписей для последующего распознавания // Системный анализ и прикладная информатика. 2022. №2. С. 4-9.
31. А. С. Жданов, Т. С. Евдокимова Распознавание русскоязычного рукописного текста на изображениях с использованием сверточной рекуррентной нейронной сети // Инженерный вестник Дона. 2025. №4.
32. Д.В. Никитин, И.С. Тараненко, А.В Катаев. Детектирование дорожных знаков на основе нейросетевой модели YOLO // Инженерный вестник Дона. 2023. №7.

## References

11. Ju. S. Zheludov. Problemy identifikacii v sistemah raspoznavanija rukopisnyh podpisov // Nauchnyj zhurnal. 2018. №9. pp. 36-37.
12. A. V. Pjataeva, M. A. Merko, V. A. Zhukovskaja, I. A. Pin'chuk, M. S. Eliseeva. Raspoznavanie rukopisnoj podpisi s primeneniem nejronnyh setej // International Journal of Advanced Studies. 2023. №13. pp. 130-148.
13. A. Smirnov, Je. S. Anisimova. Raspoznavanie rukopisnoj podpisi cheloveka s pomoshh'ju nejronnyh setej // Inzhenernyj vestnik Dona. 2024. №11. pp. 4-9.
14. Floriana Ciaglia, Francesco Saverio Zuppichini, Paul Guerrie, Mark McQuade, Jacob Solawetz. Roboflow 100: A Rich, Multi-Domain Object Detection Benchmark. 2022.
15. D. Lewis, G. Agam, S. Argamon, O. Frieder, D. Grossman, J. Heard Building a test collection for complex document information processing // 29th Annual Int. ACM SIGIR Conference (SIGIR 2006). 2006. pp. 665-666.
16. V.L. Blankers, C.E. van den Heuvel, K. Franke, L. Vuurpijl. The ICDAR 2009 Signature Verification Competition // Proceedings of the 2009 10th International Conference on Document Analysis and Recognition. 2009. pp. 1403-1407.
17. Ferrer, M. A., Diaz-Cabrera, M., Morales, A Static signature synthesis: A neuromotor inspired approach for biometrics // IEEE Transactions on pattern analysis and machine intelligence. 2014. №37(3). pp. 667-680.
18. Ahundzhanov U. Ju., Starovojtov V. V. Predvaritel'naja obrabotka izobrazhenij rukopisnyh podpisov dlja posledujushhego raspoznavanija // Sistemnyj analiz i prikladnaja informatika. 2022. №2. pp. 4-9.
19. A. S. Zhdanov, T. S. Evdokimova Raspoznavanie russkojazychnogo rukopisnogo teksta na izobrazhenijah s ispol'zovaniem svertochnoj rekurrentnoj nejronnoj seti // Inzhenernyj vestnik Dona. 2025. №4.
20. D.V. Nikitin, I.S. Taranenko, A.V Kataev. Detektirovanie dorozhnyh znakov na osnove nejrosetevoj modeli YOLO // Inzhenernyj vestnik Dona. 2023. №7.

## **Рецензия на статью**

### **«Обнаружение рукописных подписей на изображениях документов»**

Автор: Д. Р. Пономарев.

#### **Резюме.**

Статья посвящена задаче автоматического обнаружения рукописных подписей на документах с использованием нейросетевых моделей (YOLO). Автор предлагает метод комбинирования нескольких датасетов и применения аугментаций для улучшения качества детекции. Описываются этапы предобработки данных, обучения модели и полученные метрики (mAP50). Работа имеет прикладное значение для цифровизации документооборота, но требует уточнения экспериментов и сравнения с альтернативными подходами.

#### **Критические замечания.**

##### **1. Недостаточная детализация эксперимента:**

- Нет численных значений точности на тестовой выборке.
- Отсутствует сравнение с другими архитектурами (Faster R-CNN, DETR).

##### **2. Проблемы с воспроизводимостью:**

- Не описано железо, на котором проводились эксперименты.

##### **3. Нечеткость формулировок:**

- Во введении не выделена конкретная цель исследования.
- В выводах отсутствуют количественные результаты.

### **Мелкие правки.**

- Уточнить, какая версия YOLO использовалась (v5, v8?).
- Проверить оформление ссылок (например, в References у некоторых статей указаны страницы, у других — нет).

**Рекомендация редактору:** “Требуется доработки”.

Статья затрагивает актуальную тему, но нуждается в:

- Добавлении сравнения с другими методами.
- Уточнении экспериментальной части.

После исправления указанных недостатков работа может быть рекомендована к публикации.

## Реализация алгоритмов минимакс и альфа-бета отсечение на примере игры «крестики-нолики»

Л.А. Попов<sup>1</sup>

<sup>1</sup>Южный федеральный университет, Ростов-на-Дону

**Аннотация:** Задача нахождения конечного решения в игре двух игроков остаётся нерешённой для таких крупных игр, как шахматы, сёги и го, хотя возможности компьютера в этих играх уже преодолели возможности сильнейших человеческих игроков. В то же время, методы, используемые для их решения, могут применяться и для решения более простых игр. В данной статье произведено исследование и практическая реализация метода минимакса с альфа-бета отсечением на примере игры «крестики-нолики».

**Ключевые слова:** минимакс, альфа-бета отсечение, теория игр, искусственный интеллект, Python, максимизация, минимизация, оптимизация

**Минимакс.** Пытаясь найти лучший ход в игре двух игроков, человек обычно рассматривает несколько ходов, просчитывает возможные ответные ходы оппонента (а затем свои ответные ходы, и так далее), и выбирает ход, который ему кажется наиболее перспективным. Подобное поведение можно симитировать программно, рассматривая каждую линию игры до определённой глубины и таким образом оценивая позицию. Предполагается, что оба игрока пытаются максимизировать собственный выигрыш, при этом с точки зрения игрока А максимизация игроком Б своего выигрыша эквивалентна минимизации выигрыша А. Таким образом, процесс определения лучшего хода путём просчёта возможных ходов оппонентов называется минимаксом [1].

Крестики-нолики являются простой игрой для решения минимаксом. Максимальное количество ходов в партии равняется 9, что обеспечивает исчислимое количество легальных позиций. Пример части получаемого при использовании минимакса дерева решений приведено на рисунке 3 [2].

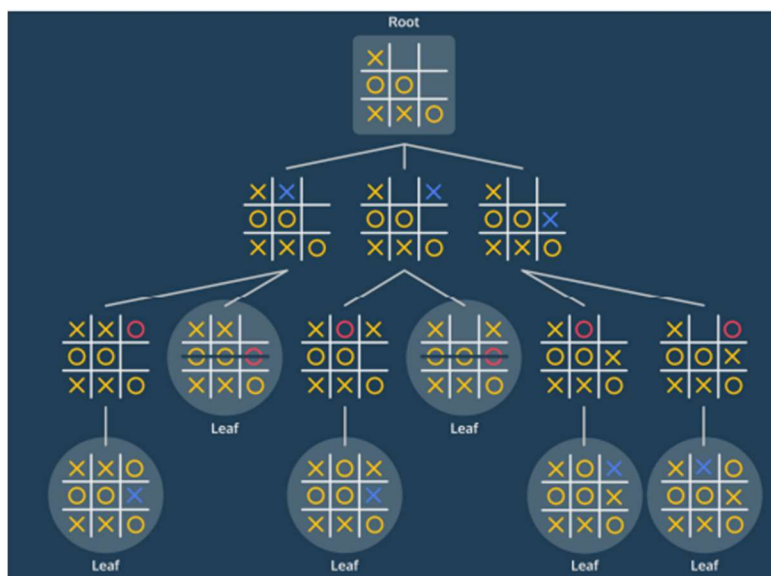


Рисунок 3 – Фрагмент дерева решений минимаксом из заданной позиции

Ввиду краткой продолжительности игры в крестики-нолики логично начислять игроку награду по достижении конечной позиции. Игроку будет начисляться положительная награда в случае победы, отрицательная в случае поражения, и ноль в случае ничьи. Кроме того, логично ввести коэффициент дисконтирования, на который будет умножаться награда, полученная не

непосредственно из текущей позиции, а из последующей. За счёт этого в случае, когда есть несколько путей достижения победной позиции, наибольшая награда будет у кратчайшего, а в случае нескольких путей к поражению, наибольшая награда будет у самого длинного. Таким образом, алгоритм будет выбирать кратчайший путь к победе и длиннейший к поражению, что является желаемым поведением. Код реализации минимакса для крестиков-ноликов на языке Python приведён в листинге 1.

```
def miniMax(game, player, depth, step=0):  
    if game.winner==player:  
        return [None,10]  
    elif game.winner==(player+1)%2:  
        return [None,-10]  
    elif game.ended:  
        return [None,0]  
    elif step==depth:  
        return [None,0]  
    arr = {}  
    discount = 0.9  
    for m in game.getLegalMoves():  
        gameCopy = game.copy()  
        gameCopy.makeMove(*m, False)  
        val = miniMax(gameCopy, player, depth, step+1)[1]  
        arr[m]=val  
    if (game.move%2==player):  
        maxVal = max(arr.values())  
        return [max(arr, key=arr.get), maxVal*discount]  
    else:  
        minVal = min(arr.values())  
        return [min(arr, key=arr.get), minVal*discount]
```

*Листинг 1 – Реализация минимакса для крестиков-ноликов*

В результате мы получаем программу, которая всегда находит самый выгодный ход в заданной позиции. Однако, количество рассматриваемых узлов дерева ходов в зависимости от глубины поиска растёт экспоненциально, что делает невозможным применение алгоритма для более сложных игр без оптимизаций.

**Альфа-бета отсечение.** Естественным направлением возможной оптимизации является уменьшение количества рассматриваемых узлов. Известным методом для достижения данной задачи является альфа-бета отсечение. В данном методе вводятся параметры  $\alpha$  и  $\beta$ , соответствующие максимальной нижней и верхней оценки позиции. Параметры рекурсивно

передаются в функцию, на максимизирующих узлах происходит переоценка значения  $\alpha$ , на минимизирующих –  $\beta$ . Если выполняется условие  $\alpha \geq \beta$ , то очевидно, что мы нашли оптимальное значение минимакса, и поиск можно досрочно прекратить. Код реализации минимакса с альфа-бета отсечением для крестиков ноликов на языке Python приведён в листинге 2.

```
def alphaBeta(game, player, depth, step=0, alpha=-100, beta=100):  
    if game.winner==player:  
        return [None,10]  
    elif game.winner==(player+1)%2:  
        return [None,-10]  
    elif game.ended:  
        return [None,0]  
    elif step==depth:  
        return [None,0]  
    arr = {}  
    discount = 0.9  
    g = -100 if game.move%2 == player else 100  
    for m in game.getLegalMoves():  
        gameCopy = game.copy()  
        gameCopy.makeMove(*m, False)  
        val = alphaBeta(gameCopy, player, depth, step+1, alpha, beta)[1]  
        arr[m]=val  
        if (game.move%2==player):  
            g = max(g, val)  
            alpha = max(alpha, g)  
            if g >= beta:  
                break  
        else:  
            g = min(g, val)  
            beta = min(beta, g)  
            if g <= alpha:  
                break  
    if (game.move%2==player):  
        maxVal = max(arr.values())  
    return [max(arr, key=arr.get), maxVal*discount]
```

```

else:
    minVal = min(arr.values())
    return [min(arr, key=arr.get), minVal*discount]

```

Листинг 2 – Реализация альфа-бета отсечения для крестиков-ноликов

**Сравнение алгоритмов.** Проведём экспериментальное сравнение обычного минимакса и минимакса с альфа-бета отсечением. В качестве критерия для сравнения было выбрано количество рассмотренных алгоритмами узлов дерева игры в зависимости от максимальной глубины поиска. Результаты данного сравнения представлены на рисунке 4.

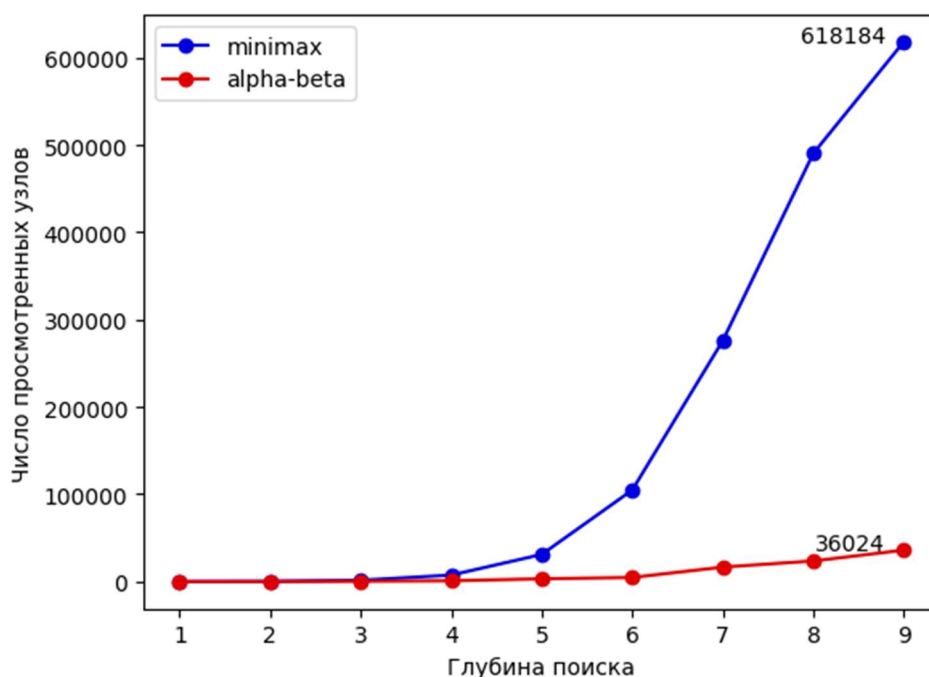


Рисунок 4 – График зависимости числа рассмотренных позиций от глубины поиска

Как видно по данному графику, альфа-бета отсечение позволяет значительно сократить количество рассматриваемых узлов на большой глубине поиска. На максимально возможной глубине для крестиков-ноликов минимакс рассмотрел 618184 узлов, в то время как альфа-бета отсечение позволило сократить это количество до 36024 узлов.

## Список использованных источников

1. A. Plaat. Research Re: Search & Re-search. – 1996. – URL: <https://arxiv.org/abs/2403.13705>
2. Жданов, Н. А. Алгоритм "минимакс" и его реализация на примере игры "крестики-нолики" / Н. А. Жданов, Ю. М. Бурханова, Ю. О. Воронецкий // Политехнический молодежный журнал. – 2019. – № 5(34). – С. 4. – URL: <https://www.elibrary.ru/item.asp?id=38525855>



**Рецензия на статью**  
**«Реализация алгоритмов минимакс и альфа-бета отсечение на примере  
игры „крестики-нолики“»**

---

**Резюме**

Статья посвящена исследованию и практической реализации алгоритмов минимакс и альфа-бета отсечение на примере игры «крестики-нолики». Автор последовательно излагает теоретические основы, описывает принципы работы алгоритмов и демонстрирует их реализацию на языке Python. Отдельное внимание уделено сравнению эффективности алгоритмов по числу рассматриваемых узлов на разных глубинах поиска. Работа выполнена в образовательном формате и может быть полезна студентам и начинающим разработчикам, однако содержит методологические и технические недочёты, требующие доработки.

---

**Сильные стороны**

- Чёткое изложение теоретической базы алгоритмов минимакс и альфа-бета отсечение.
  - Приведение практических примеров кода на Python, что делает статью наглядной и доступной для восприятия.
  - Экспериментальное сравнение производительности алгоритмов с использованием графиков.
  - Логичная структура текста, наличие аннотации, ключевых слов и списка литературы.
- 

**Критические замечания**

**1. Отсутствие формализованной постановки задачи**

В статье нет явного формулирования исследовательского вопроса или цели работы. Автору следует добавить раздел «Цель и задачи исследования», где описать, что именно он хочет продемонстрировать: проверку эффективности минимакса, иллюстрацию сокращения числа узлов или универсальность метода.

**2. Недостаточно проработана методология эксперимента**

В разделе сравнения (рисунок 2) не указано, на каком оборудовании и с какими параметрами (например, глубина поиска, вариации начальных

позиций) проводился эксперимент. Автору следует уточнить условия эксперимента, чтобы обеспечить воспроизводимость результатов.

### **3. Ограниченность применимости результатов**

Автор фокусируется исключительно на игре «крестики-нолики», которая является тривиальной с точки зрения вычислительных затрат. Следует хотя бы кратко обсудить, какие сложности возникнут при переносе методов на более сложные игры (например, шахматы, го), где минимакс без эвристик становится неработоспособным.

### **4. Отсутствие обсуждения сложности алгоритмов**

Необходимо включить раздел с анализом временной и пространственной сложности алгоритмов. Например, минимакс имеет экспоненциальную сложность относительно глубины дерева, и это стоит указать явно.

### **5. Не до конца объяснён код**

Листинги кода на Python приведены без комментариев, что затрудняет понимание. Автору следует добавить пояснения к ключевым участкам кода: назначение параметров, роль коэффициента дисконтирования, интерпретация возвращаемых значений.

---

## **Мелкие правки**

- В аннотации: следует исправить «произведено исследование» на «проведено исследование» для соблюдения норм русского языка.
- В тексте: нужно уточнить, что рисунок 1 представлен схематически (в данный момент описание неполное).
- В листинге 1: следует добавить информацию о том, какие значения принимают `player` и `game.move`, так как это неочевидно для внешнего читателя.
- В списке литературы: нужно указать дату последнего доступа для онлайн-источников.
- График на рисунке 2: следует добавить подписи осей (например, «Глубина поиска», «Число рассмотренных узлов»), чтобы график был полностью самодостаточным.

---

## **Рекомендация редактору**

**Статья требует доработки.** Основные претензии касаются недостаточной формализации цели, описания методологии эксперимента и отсутствия

пояснений к коду. После внесения указанных изменений работа может быть рекомендована к публикации в учебно-научных или популярно-научных изданиях, ориентированных на начинающих специалистов в области искусственного интеллекта и алгоритмов поиска.

# Минимизация количества пересылок данных при распараллеливании программ

Мурашко Е.М., Штейнберг Б.Я.

Южный федеральный университет, Ростов-на-Дону

**Аннотация:** Задача минимизации количества пересылок данных сводится к алгоритму нахождения оптимального размещения массивов в распределенной памяти параллельной вычислительной системы. Данная работа является шагом на пути к разработке нового поколения распараллеливающих компиляторов для вычислительных систем с распределенной памятью. Такие компиляторы могут быть полезны для многоядерных систем на кристалле с адресуемой локальной памятью (не кэш-памятью) для каждого ядра. Это особенно важно для систем на кристалле с большим количеством процессорных ядер, где обмен данными с оперативной памятью повышает производительность. горлышко бутылки. Получено построение графа специальной вспомогательной программы для оценки минимального количества передач данных. В данной работе строится размещение массивов с оптимальным количеством передач данных

**Ключевые слова:** автоматизация распараллеливания, распределенная память, преобразования программ, размещение данных, пересылки данных

## Введение

В данной работе проведён анализ инструментов и методов, которые направлены на решение проблемы минимизации количества пересылок данных при распараллеливании программ. Результаты данной работы могут найти применение при создании распараллеливающих компиляторов на вычислительные системы с распределенной памятью. Такие компиляторы, в первую очередь, должны быть интересны для микросхем типа «суперкомпьютер на кристалле» [8], [9].

## Блочнo-аффинные размещения массивов и межпроцессорные пересылки

Основная особенность параллельного выполнения цикла на ВС с распределенной памятью состоит в том, что для каждой операции ее аргументы должны быть в одном модуле распределенной памяти. Будем полагать, что ВС состоит из  $p$  процессорных элементов (ПЭ). Каждый ПЭ состоит из процессора и собственного модуля памяти получения данных, из которого происходит быстрее, чем из модулей памяти других ПЭ. Все ПЭ занумерованы, начиная с нуля. Размещение массива в памяти – это функция, для каждого элемента массива возвращающая номер ПЭ, в котором этот элемент находится. При описании параллельных алгоритмов рассматриваются размещения матриц (двумерных массивов) «по строкам», «по столбцам», «по полосам строк», «по полосам столбцов», «по скошенным диагоналям». Эти описания, как и многие другие, могут быть описаны, как блочно-аффинные размещения по модулю количества ПЭ.

**Определение 1.** Пусть натуральные числа  $p, d_1, d_2, \dots, d_m$  и целые константы  $s_0, s_1, s_2, \dots, s_m$  зависят только от  $m$ -мерного массива  $X$ . **Блочнo-аффинное** по модулю  $p$  размещение  $m$ -мерного

массива X – это такое размещение, при котором элемент  $X[i_1, i_2, \dots, i_m]$  находится в модуле памяти ПЭ с номером:

$$u = \left( \left\lfloor \frac{i_1}{d_1} \right\rfloor * s_1 + \left\lfloor \frac{i_2}{d_2} \right\rfloor * s_2 + \dots + \left\lfloor \frac{i_m}{d_m} \right\rfloor * s_m + s_0 \right) \bmod p$$

Число  $s_0$  показывает номер модуля памяти, в котором размещается нулевой элемент  $X[0, 0, \dots, 0]$ . При описанном блочно-аффинном способе размещения  $m$ -мерный массив представляется как массив блоков размерности  $d_1 \ d_2 \ \dots \ d_m$ , который размещается так, что у каждого блока все элементы оказываются в модуле памяти одного ПЭ. Числа  $p, d_1, d_2, \dots, d_m, s_0, s_1, s_2, \dots, s_m$  будем называть параметрами размещения.

Программы для ВС с распределенной памятью отличаются наличием межпроцессорных пересылок данных, которые являются самой дорогостоящей (по времени) операцией. Пересылки и должны минимизироваться при распараллеливании программ на такие архитектуры.

**Пересылка** – это команда коммуникационной системы.

Если для вычислений в одном ПЭ необходимы данные, лежащие в другом ПЭ, то необходимо ожидание вычислений перед получением необходимых данных. Проще всего прервать вычисления во всех ПЭ, пока совершается межпроцессорная пересылка. Такие прерывания синхронизируют вычисления, но при большом количестве ПЭ могут создавать большие потери времени.

### Граф “операторы-переменные”

Важнейшим элементов анализа программных циклов в распараллеливаемых программах является граф «операторы-переменные» (ГОП) [2]-[7].

```
for (int j = 0; j < N; j++) {  
1   X[j+1] = Y[j+2];  
2   Z[j+2] = X[j+2];  
3   Y[j+2] = T[j+2];  
4   T[j+2] = X[j] + Y[j+1] + Z[j+2]  
   }
```

Листинг 1.

**Граф “операторы-переменные”** строится на основе тела цикла. Это **двудольный граф**, вершины одной части которого являются операторами цикла, другой - переменными. Ребро соединяет переменную и оператор, если такая переменная встречается в этом операторе. Каждое появление переменной соответствует одному ребру графа (возможны кратные рёбра).

**Вес ребра** соответствует целочисленной константе в индексном выражении массива

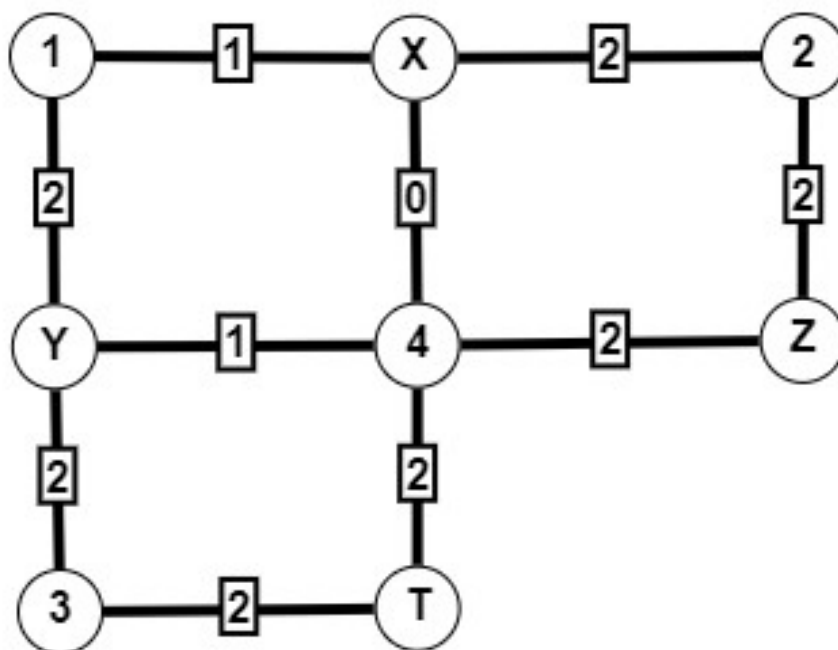


Рис. 1 - ГОП программного цикла листинга 1. Вершины графа (номера операторов и имена переменных) в кружечках, веса ребер в прямоугольниках.

Таблица 1. ГОП, представленный в виде списка ребер

| Номер ребра в ГОП | Вершина – номер оператора | Вершина – имя переменной | Вес ребра – константа в индексном выражении |
|-------------------|---------------------------|--------------------------|---------------------------------------------|
| 1                 | 1                         | X                        | 1                                           |
| 2                 | 1                         | Y                        | 2                                           |
| 3                 | 2                         | Z                        | 2                                           |
| 4                 | 2                         | X                        | 2                                           |
| 5                 | 3                         | Y                        | 2                                           |
| 6                 | 3                         | T                        | 2                                           |
| 7                 | 4                         | T                        | 2                                           |
| 8                 | 4                         | X                        | 0                                           |
| 9                 | 4                         | Y                        | 1                                           |
| 10                | 4                         | Z                        | 2                                           |

Приведем свойства графа «операторы-переменные» [2], [5], [7].

**Теорема 1.** Если граф “операторы-переменные” представляет собой лес (каждый подключенный компонент представляет собой дерево), то массивы могут быть размещены таким образом, чтобы цикл мог выполняться без пересылок данных.

**Теорема 2.** Для любого цикла графа “операторы-переменные” и ребра этого цикла в программу можно добавить пересылку данных так, что вместо этого ребра будет другое, в графе будет на один цикл меньше (цикл, о котором идет речь, будет нарушен).

**Теорема 3.** Если в графе “операторы-переменные” программного цикла есть  $k$  компонент связности,  $n$  вершин и  $m$  ребер, то  $(m - n + k)$  межпроцессорных пересылок данных может привести граф к форме “леса”, и цикл может выполняться параллельно.

## Согласованное размещение массивов

Построим блочно-аффинное размещение массивов [10] в распределенной памяти и вставим межпроцессорные пересылки в программу листинг 1. Для простоты будем полагать, что в вычислительной системе имеется 4 процессорных элемента (ПЭ). Необходимо привести программу к виду, в котором переменные согласованы в каждом операторе с использованием наименьшего количества передач данных.

**Определение 2.** Мы будем говорить, что значения индексных переменных  $X[j + b]$  и  $Y[j + b]$  **согласованы**, если для каждого значения счетчика цикла  $j$  эти значения находятся в одном и том же модуле памяти. В каждом операторе программы для параллельного выполнения цикла должны быть помещены все вхождения переменных

**Лемма 1.** (условие согласованности двух вхождений). Пусть вхождения индексных переменных  $X[j + b]$  и  $Y[j + c]$  расположены аффинно со сдвигами  $s_1$  и  $s_2$ , соответственно. Эти события считаются согласованными, если для каждого значения счетчика циклов  $j$  выполняется следующее условие:

$$(j + b + s_1) \bmod p = (j + c + s_2) \bmod p$$

| ПЭ0    | ПЭ1    | ПЭ2   | ПЭ3   |
|--------|--------|-------|-------|
| X[0]   | X[1]   | X[2]  | X[3]  |
| X[4]   | X[5]   | X[6]  | X[7]  |
| Y[1]   | Y[2]   | Y[3]  | Y[4]  |
| Y[5]   | Y[6]   | Y[7]  | Y[8]  |
| Z[0]   | Z[1]   | Z[2]  | Z[3]  |
| Z[4]   | Z[5]   | Z[6]  | Z[7]  |
| T[1]   | T[2]   | T[3]  | T[4]  |
| T[5]   | T[6]   | T[7]  | T[8]  |
| ZZ[-2] | ZZ[-1] | ZZ[0] | ZZ[1] |
| ZZ[2]  | ZZ[3]  | ZZ[4] | ZZ[5] |
| TT[-2] | TT[-1] | TT[0] | TT[1] |
| TT[2]  | TT[3]  | TT[4] | TT[5] |

```

    for (int j = 0; j < N; j++) {
1      X[j+1] = Y[j+2];
2      Z[j+2] = X[j+2];
3      Y[j+2] = T[j+2];
Transfer
TT[j] ← (-2) ← T[j-2]
```

```

ZZ[j] ← (-2) ← Z[j-2]
4   TT[j] = X[j] + Y[j+1] + ZZ[j]
    }

```

Листинг 2.

Как видно по листингу 2, в приведённой выше программе понадобилось провести 2 пересылки данных

### Заключение

Данная статья представляет собой шаг на пути к созданию оптимизирующих распараллеливающих компиляторов на высокопроизводительные системы на кристалле нового поколения типа «суперкомпьютер на кристалле». В ней был проведён анализ следующих инструментов достижения минимизации пересылок данных:

1. Блочнo-аффинные размещения
2. Граф “операторы-переменные”
3. Согласованное размещение массивов

### Список используемой литературы

- 1) Харари Ф. Теория графов. М.: Мир, 1973. Изд. 3, М.: КомКнига, 2006. — 296 с.
- 2) [Nikita M. Krivosheev, Boris Ya. Steinberg. “Algorithm for Searching Minimum Inter-node Data Transfers”, Procedia Computer Science Volume 193, 2021, Pages 306-313](#)
- 3) [Баглий А.П., Кривошеев Н.М., Штейнберг Б.Я. Автоматизация распараллеливания программ с оптимизацией пересылок данных // Научный сервис в сети Интернет: труды XXIV Всероссийской научной конференции \(19-22 сентября 2022 г., онлайн\). — М.: ИПМ им. М.В.Келдыша, 2022. — С. 81-92.](#)
- 4) [Баглий А. П., Кривошеев Н. М., Штейнберг Б. Я., Штейнберг О.Б. Преобразования программ в Оптимизирующей распараллеливающей системе для распараллеливания на распределенную память/ Инженерный вестник Дона, 2022, № 12, 20 с. \(электронный журнал\)](#)
- 5) [А.П. Баглий, Н.М. Кривошеев, Б.Я. Штейнберг Автоматизация распараллеливания программ для многоядерных процессоров с распределенной локальной памятью. Электронные библиотеки. 2023. Т. 26. № 2. С. 135-153.](#)
- 6) [Bagliy A.P., N.M. Krivosheev, B.Ya. Steinberg «Automatic mapping of sequential programs to parallel computers with distributed memory», Volume 229, 2023, Pages 236-244.](#)
- 7) [Bagliy A.P., Krivosheev N.M., Steinberg B.Ya., Steinberg O.B. Automation of Parallelization of Loops with Minimization of Data Transfers // Параллельные вычислительные технологии - XVII всероссийская конференция с международным участием, ПаВТ'2023, г. Санкт-Петербург, 28-30 марта 2023 г. Короткие статьи и описания плакатов. Челябинск: Издательский центр ЮУрГУ, 2023. С. 7-16](#)
- 8) [Epiphany-V: A 1024-core 64-bit RISC processor. URL: https://parallella.org/2016/10/05/epiphany-v-a-1024-core-64-bit-risc-processor](https://parallella.org/2016/10/05/epiphany-v-a-1024-core-64-bit-risc-processor), дата обращения 26.03.2022.



- 9) SoCEsperanto. URL: <https://www.esperanto.ai/technology>, дата обращения 26.03.2022
- 10) Штейнберг Б.Я. Блочно-аффинные размещения данных в параллельной памяти // Информационные технологии. 2010. №6. С. 36–41.

#### Рецензия на статью

#### **«Минимизация количества пересылок данных при распараллеливании программ»**

(Мурашко Е.М., Штейнберг Б.Я., ЮФУ, Ростов-на-Дону)

#### Резюме

Авторы рассматривают задачу минимизации межпроцессорных пересылок в циклических фрагментах программ для систем с распределённой памятью. Предложена формальная модель блочно-аффинного размещения массивов и граф «операторы-переменные» (ГОП) как средство оценки и уменьшения числа передач. На иллюстративном примере цикла (листинг 1) показано построение ГОП, выводятся три теоремы о связи его свойств с количеством необходимых пересылок и демонстрируется преобразованный код, требующий лишь двух передач данных. Работа позиционируется как этап в разработке распараллеливающих компиляторов для систем-на-кристалле.

#### Критические замечания

1. **Неопределённость целевой аудитории.** Во «Введении» заявлены как академические, так и прикладные цели (компиляторы для SoC «суперкомпьютер на кристалле»), но критерии эффективности метода не соотнесены с реальными ограничениями аппаратных платформ (ёмкость локальной памяти, пропускная способность сети-на-кристалле).
2. **Отсутствие экспериментальной валидации.** Статья ограничивается теоретическим анализом и единичным примером ( $N$  — параметр цикла). Не представлены ни количество реально сокращённых пересылок на стандартных тестах, ни выигрыш по времени выполнения.
3. **Недостаточная проработка алгоритма размещения.** Формальное определение блочно-аффинного размещения (Определение 1) приведено, однако не описан алгоритм автоматического выбора параметров  $d_i$  и  $s_i$ , что критично для компиляторной реализации. Также может быть непонятна формула, скобки вида « $()$ » вызывают разночтения.
4. **Неясность доказательств теорем 2 и 3.** Формулировки интуитивно правдоподобны, но отсутствуют полные доказательства или ссылки на более детальные публикации.

5. **Неполное сравнение с существующими подходами.** В разделе «Заключение» перечислены рассмотренные инструменты, однако не проведён анализ преимуществ перед классическими методами распределения данных (HPF, PLUTO, Polly) и эвристиками минимизации коммуникаций.
6. **Стилистическая неоднородность.** В тексте встречаются разговорные выражения («горлышко бутылки») и повторяющиеся фрагменты (дважды описан процесс построения ГОП).
7. **Неточности обозначений.** В листинге 2 используются переменные **ТТ** и **ZZ**, которые не определены в исходном листинге 1; следует пояснить их происхождение и правила именования.
8. **Ссылки без даты обращения.** В списке литературы веб-источники [8], [9] приведены без стандартизированного указания даты доступа, что противоречит ГОСТ.

#### Мелкие правки

- **Таблица 1 (стр. ≈ 6).** Заголовки столбцов выравнивайте по центру и уточните единицы измерения веса ребра («константа сдвига, ед.»).
- **Рисунок 1.** Сделайте подписи вершин крупнее и выделите различными стилями операторные и переменные узлы для лучшей читаемости.
- **Листинг 1.** Выравнивание кода нарушено: отступы смешивают табуляции и пробелы; используйте единый стиль.
- **Формула (1).** Добавьте описание всех переменных непосредственно под формулой, чтобы не перелистывать к определению 1.
- **Таблица перед Листингом 2.** Добавить описание.
- **Орфография.** Исправьте «является» → «является», «обраще-ния» → «обращения».

#### Сильные стороны

- Чёткое формальное описание блочно-аффинных размещений данных.
- Интуитивно понятная визуализация зависимости «оператор  $\leftrightarrow$  переменная» через двудольный граф, что облегчает объяснение алгоритма.
- Практическая ориентированность на современные многоядерные SoC, где кэш-иерархия ограничена или отсутствует.

#### Рекомендация редактору

**Статус:** *Требуется доработки*

Предлагаю вернуть работу авторам для существенного расширения экспериментального раздела и/или дополнения статьи.



# Создание шейдера для отображения эффекта меха с помощью комбинирования методов *fin* и *shell* в рамках игрового движка Unity.

М.К. Селина<sup>1</sup>

<sup>1</sup>Южный федеральный университет, Ростов-на-Дону

**Аннотация:** В статье рассматриваются методы создания эффекта меха в компьютерных играх с целью достижения баланса между визуальным качеством и производительностью. Анализируются основные подходы: метод оболочек (*shell*), метод плавников (*fin*) и метод преобразования геометрии модели. В работе предложена комбинированная реализация методов *shell* и *fin* на базе игрового движка Unity с использованием HLSL и ShaderLab, что позволяет улучшить реализм меха при приемлемой производительности, а так же описаны ключевые этапы реализации. Исследование способствует развитию технологий рендеринга меховых поверхностей в интерактивных приложениях..

**Ключевые слова:** компьютерная графика, генерация меха, Unity, HLSL, ShaderLab, метод *shell*, метод *fin*.

## Введение

В современном мире компьютерные игры занимают значимое место в индустрии развлечений и технологий. Одним из ключевых элементов, определяющих визуальное качество, реалистичность и стилизацию игр, являются шейдеры. Они представляют собой небольшие программы, выполняемые на графическом процессоре (GPU), и отвечают за обработку и отображение графических объектов.

Шейдеры бывают разной сложности: от простых эффектов освещения и текстурирования до сложных вычислительных алгоритмов, моделирующих взаимодействие света с материалами. К числу последних относится рендеринг меховых материалов. С точки зрения компьютерной данные шейдеры сложные и ресурсоёмкие поскольку требуют тщательных вычислений для передачи мельчайших деталей и динамики волосков.

Однако, несмотря на высокую сложность данный эффект остается востребованным и актуальным, поскольку отображение и детализация меховых поверхностей активно обсуждается и привлекают внимание игроков. Кроме того, с каждым годом появляются новые технологии и методы, позволяющие оптимизировать эти шейдеры и повысить их производительность без потери качества [1].

## Обзор существующих методов

В рамках исследования были рассмотрены основные методы создания эффекта меха, описанные в статьях разработчика под псевдонимом *hesomi*: метод *shell* (оболочки), метод *fin* (плавника) и метод преобразования геометрии модели.

Метод *Shell* предполагает создание нескольких дополнительных слоев полигонов вокруг основной модели, смещённых вдоль нормалей вершин [2]. Это позволяет имитировать объемный мех без сложной геометрии, управляя толщиной меха программно или средствами моделирования. Визуальные эффекты усиливаются использованием карты шума и расчётом освещения с учётом уровня слоя. Однако при близком рассмотрении слои оболочки могут быть заметны, особенно на краях модели. Это снижает реализм и разрушает красоту эффекта.

Метод *Fin* создаёт мех за счёт добавления перпендикулярных граней к полигонам модели - «финов» [3]. Для повышения качества и равномерности финов применяется тесселяция, которая позволяет повышать плотность полигонов модели. Освещение рассчитывается с помощью карт нормалей, адаптированных под направление финов. Этот метод хорошо подходит для длинного меха и хорошо отображается на краях фигуры, но требует большого количества полигонов, что влияет на производительность, а так же не естественно ведет себя на полигонах, нормаль которых направлена в сторону камеры.

Метод изменения геометрии заключается в преобразовании каждого полигона модели в вытянутую пирамиду с помощью добавления новых вершин и тесселяции [4]. Это обеспечивает объём и фактуру меха без потери качества на краях и при различных углах обзора. Для корректного освещения нормали пересчитываются с учётом новой геометрии. Несмотря на визуальные преимущества, метод усложняет расчёт освещения и может негативно сказаться на производительности из-за увеличения числа вершин.

В итоге, наиболее подходящим для реализации является комбинация метода `fin` с методом `shell`, что позволяет сбалансировать качество визуализации и производительности.

### **Реализация**

Для реализации в рамках игрового движка Unity используется язык компьютерной графики HLSL, а также язык ShaderLab. Для создания комбинированного эффекта отображения меха необходимо реализовать каждый из методов по отдельности.

В вершинном шейдере метода `shell` происходит передача входных данных модели без изменений. Основная логика создания оболочек реализуется в геометрическом шейдере. Для каждого полигона исходной модели создается несколько копий вершин, которые смещаются вдоль нормали с учетом базового смещения, физики и влияния ветра. Смещение рассчитывается в зависимости от номера слоя, что позволяет создавать постепенное увеличение смещения по слоям, которое немного скрывает явные недостатки данного метода. Добавлена проверка угла между направлением взгляда и нормалью, чтобы отсеять слои, которые не видны под определенным углом. После этого во фрагментный шейдер передаются UV-координаты, мировые позиции и нормали, а также индекс слоя для управления прозрачностью. Цвет каждого пикселя определяется текстурой меха с учетом альфа-канала и рассчитывается как произведение значения канала текстуры меха и соответствующей пропорции слоя, чем выше слой – тем больше прозрачность. К полученному значению применяется затенение с учетом удаленности от самой внешней поверхности. Для оболочек, кроме изначальных полигонов модели, реализована дополнительная проверка угла между нормалью и направлением взгляда, чтобы отсеять пиксели по краям модели, что представлено на рис.1.

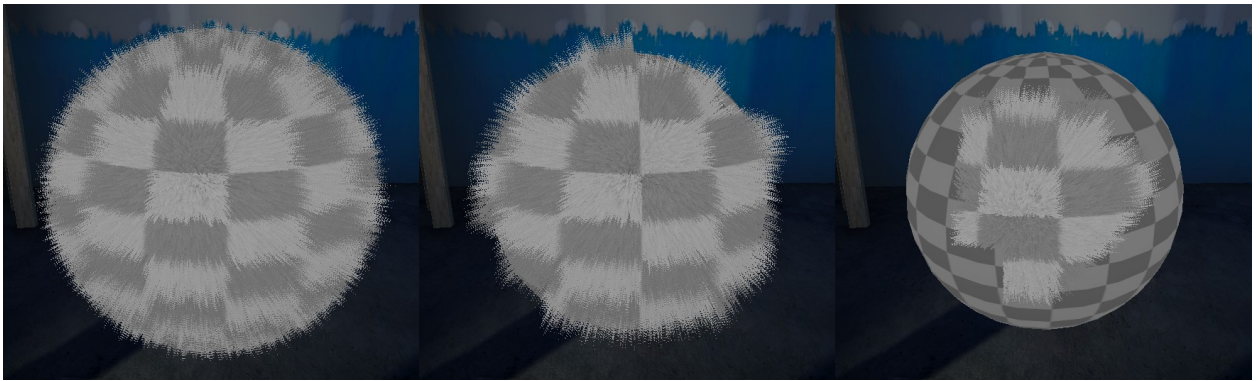


Рис. 2. – Демонстрация метода shell с учетом изменения значения угла отрисовки фрагментов

В вершинном шейдере метода *fin* так же выполняется базовая передача входных данных без изменений. В отличие от предыдущей реализации данные передаются сначала в шейдер тесселяции, где используется метод Фонга, подобный способу затенения. Уровень разбиения на полигоны адаптируется динамически в зависимости от расстояния до камеры. Рассчитанные новые позиции и нормали с учетом измененной кривизны поверхности передаются в геометрический шейдер, где для каждого полигона исходной модели генерируются дополнительные перпендикулярные плоскости – «фины». Для этого вычисляется нормаль полигона, его центр и проверяется угол между направлением взгляда и нормалью, чтобы отсеять плавники слишком удаленные от края модели, что представлено на рис. 2. Направление каждого «фина» определяется случайным образом, для придания естественности, после чего создаются его вершины и присваиваются UV-координаты для корректного наложения текстуры клочка меха. С помощью дополнительных параметров можно влиять на плотность меха, его длину и направление роста. Во фрагментном шейдере к «финам» применяется текстура клочка меха с учетом прозрачности некоторых ее участков. Цвет дополнительных полигонов смешивается с базовой текстурой и цветом объекта, после чего как и в предыдущей реализации применяется затенение в зависимости от их длины «фина», создавая объем и глубину.



Рис. 2. – Демонстрация метода *fin* с учетом изменения значения угла отрисовки фрагментов

Для комбинации двух методов, теперь достаточно поместить их выполнение в разные проходы и подредктировать параметры. Итоговый вид шейдера представлен на рис. 3.



Рис. 3. – Демонстрация комбинированного метода с изменением длины меха

### Дальнейшая работа

Реализованные шейдеры не оптимизированы и содержат большое количество ветвлений и дорогих операций, поэтому нуждаются в доработке. Во многих обзорных статьях и научных работах комбинация методов *fin* и *shell* отмечалась как теоретически наиболее подходящая как по визуальному отображению, так и по относительно низким затратам по производительности.

Однако авторы в упомянутых работах при реализации либо не получали желаемого эффекта или просто не приводили итоговые результат. Именно поэтому изначальная задача стояла в удачном совмещении методов, и получении приемлемого результата хотя бы визуально.



В дальнейшем работа с данным шейдером будет направлена на его оптимизацию и доработку визуального отображения.

### **Литература**

33. Самюэль Стайлс. Бакалаврская работа на тему 3d генерации меха. // Статья в личном блоге автора. URL: <https://sstyles93.github.io/blog/posts/BachelorProject/>
34. Автор под псевдонимом hecomi. Статья на тему метода shell. // Статья в личном блоге автора. URL: <https://tips.hecomi.com/entry/2021/06/27/185835>
35. Автор под псевдонимом hecomi. Статья на тему метода fin. // Статья в личном блоге автора. URL: <https://tips.hecomi.com/entry/2021/07/24/121420>
36. Автор под псевдонимом hecomi. Статья на тему метода изменения геометрии. // Статья в личном блоге автора. URL: <https://tips.hecomi.com/entry/2021/08/12/155948>.

### **References**

21. Samuel Styles. Bachelor's thesis on 3d fur generation. // An article in the author's personal blog. URL: <https://sstyles93.github.io/blog/posts/BachelorProject/>
22. The author is under the pseudonym ecama. An article on the shell method. // An article in the author's personal blog. URL: <https://tips.hecomi.com/entry/2021/06/27/185835>
23. The author is under the pseudonym hecomi. An article on the find method. // An article on the author's personal blog. URL: <https://tips.hecomi.com/entry/2021/07/24/121420>
24. The author is under the pseudonym hecomi. An article on methods for changing geometry. // An article in the author's personal blog. URL: <https://tips.hecomi.com/entry/2021/08/12/155948>.

# **Рецензия на статью «Создание шейдера для отображения эффекта меха с помощью комбинирования методов `fin` и `shell` в рамках игрового движка Unity»**

## **Резюме**

Представленная статья посвящена разработке шейдера для эффекта меха в игровом движке Unity. В работе рассматриваются три метода: `shell`, `fin` и изменение геометрии, и предлагается их комбинирование с целью улучшения баланса между качеством и производительностью. Автор подробно описывает реализацию каждого метода, демонстрирует промежуточные результаты и делает выводы о необходимости дальнейшей оптимизации. Статья направлена на развитие рендеринга меха в интерактивных приложениях.

## **Критические замечания**

1. В разделе «Дальнейшая работа» отмечено, что шейдеры не оптимизированы, однако нет результатов бенчмарков (FPS, нагрузка на GPU) до и после объединения методов. Можно уточнить как изменяется производительность при разных уровнях детализации и настройках.
2. Литературные источники преимущественно неакадемические, ссылки ведут на персональные блоги. Добавление публикаций из рецензируемых журналов, посвящённые компьютерной графике усилило бы научную обоснованность исследования.

## **Мелкие замечания**

1. В списке литературы стоит добавить даты обращения к онлайн-ресурсам.
2. На рисунке 1 можно указать единицы угла (градусы) и масштаб смещения оболочек.

## **Рекомендации**

Принять. Статья представляет собой качественную прикладную работу, в которой автор демонстрирует уверенное владение инструментами разработки графических шейдеров и предлагает комбинированный подход к визуализации меха в игровом движке Unity. Работа обладает чёткой структурой, включает подробное описание этапов реализации,

сопровождается наглядными иллюстрациями и демонстрирует потенциал для дальнейших исследований в области оптимизации визуальных эффектов.

# Криптография на эллиптических кривых: математический фундамент

А.С. Харина

<sup>1</sup>Южный федеральный университет, Ростов-на-Дону

<sup>2</sup>Институт математики, механики и компьютерных наук имени И.И. Воровича

**Аннотация:** Современная криптография во многом опирается на сложные математические задачи, которые трудно решить даже с использованием мощных вычислительных ресурсов. Одним из самых перспективных направлений в этой области стала криптография на эллиптических кривых (ECC – Elliptic Curve Cryptography), предлагающая высокий уровень безопасности при меньших вычислительных затратах по сравнению с традиционными методами, такими как RSA. Эллиптические кривые – это не просто абстрактные математические объекты, а мощный инструмент, позволяющий строить эффективные криптосистемы. Их стойкость основывается на сложности задачи дискретного логарифмирования в группе точек эллиптической кривой (ECDLP), которая на сегодняшний день не поддается эффективному решению с помощью известных алгоритмов, включая квантовые атаки (в отличие от факторизации, на которой основан RSA).

## Введение

Современный этап развития информационных технологий характеризуется стремительным ростом объемов передаваемых и хранимых данных, что приводит к ужесточению требований к их защите. В этих условиях особую актуальность приобретают эффективные криптографические методы, способные обеспечить надежную информационную безопасность при минимальных вычислительных затратах.

Традиционные криптосистемы с открытым ключом, такие как RSA, основанные на сложности факторизации больших чисел, постепенно уступают место более современным решениям.

Криптография на эллиптических кривых (ECC) представляет собой принципиально иной подход к защите информации, основанный на математических свойствах алгебраических кривых. Основное преимущество ECC заключается в том, что она обеспечивает сопоставимый с RSA уровень безопасности при существенно меньшей длине ключа (например, 256-битный ключ ECC эквивалентен по стойкости 3072-битному ключу RSA). Это делает ECC особенно

перспективной для применения в условиях ограниченных вычислительных ресурсов - в мобильных устройствах, системах интернета вещей (IoT) и блокчейн-технологиях.

## Математические основы ЕСС:

### 1.1 Уравнение эллиптической кривой

Эллиптическая кривая над простым полем  $F_p$  задается уравнением Вейерштрасса [1]:

$$y^2 \equiv x^3 + ax + b \pmod{p},$$

где:  $p$  – простое число,

$a, b$  – коэффициенты, удовлетворяющие условию

$$4a^3 + 27b^2 \not\equiv 0 \pmod{p}$$

### 1.2 Математические операции

Операции над конечным полем основаны на аффинной системе координат, в которой каждая точка системы координат представлена вектором  $(x, y)$ . [6]

*Сложение точек*

Пусть даны две различные точки  $P(x_1, y_1)$  и  $Q(x_2, y_2)$  на кривой

$E(a, b)$ . Тогда сумму точек вычислим следующим образом:

$$R(x_3, y_3) = P(x_1, y_1) + Q(x_2, y_2)$$

- Если  $x_1 \neq x_2$

$$x_3 = \lambda^2 - x_1 - x_2$$

$$y_3 = \lambda(x_1 - x_3) - y_1,$$

где  $\lambda$ :

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1}$$

$$x_2 - x_1$$

- Если  $x_1 = x_2$ , но  $P \neq Q$  то результатом является точка на бесконечности

### Удвоение точек

Пусть определены две точки  $P(x_1, y_1)$  и  $Q(x_2, y_2)$  на эллиптической кривой  $E(a, b)$ , который равны друг другу ( $P = Q$ ). Тогда удвоение точек определим, как [6]:

$$R(x_3, y_3) = P(x_1, y_1) + P(x_1, y_1) = 2P$$

$$x_3 = \lambda^2 - 2x_1$$

$$y_3 = \lambda(x_1 - x_2) - y_1$$

Где  $\lambda$ :

$$\lambda = \frac{3x_1^2 + a}{2y_1}$$

Вычитание  
точек:

1

Пусть определены две точки  $P(x_1, y_1)$  и  $Q(x_2, y_2)$  на эллиптической кривой  $E(a, b)$ , тогда отрицательную точку можно определить, как [6]:

$$-Q(x_2, y_2) = Q(x_2, -y_2)$$

Тогда вычитание двух точек, можно  
представить в виде:

$$P(x_1, y_1) - Q(x_2, y_2) = P(x_1, y_1) + Q(x_2, -y_2)$$

Скалярное умножение:

Для точки  $P(x_1, y_1)$  и целого числа  $n$  операция  $nP$  выполняется через повторное сложение и удвоение.

Например[6]:



$$9P = 2(2(P + P)) + P$$

## Проблема дискретного логарифма (DLP)

### 2.1 Основные понятия

Пусть  $G$  — конечная циклическая группа, и пусть  $\alpha$  — её образующий элемент. Проблема дискретного логарифма в группе  $G$  по основанию  $\alpha$  заключается в нахождении для заданного элемента  $\beta \in G$  единственного целого числа  $d \in [0, |G| - 1]$ , удовлетворяющего уравнению  $\alpha^d = \beta$ . Это число  $d$  обозначается как  $\log_\alpha \beta$ . [3]

### 2.2 Примеры групп и сложность DLP

Рассмотрим два классических примера:

*Аддитивная группа  $\mathbb{Z}_N$*

Здесь DLP сводится к решению уравнения  $d * \alpha \equiv \beta \pmod{N}$ . Если  $\gcd(\alpha, N) = 1$ , то решение находится за полиномиальное время ( $O(\log^2 N)$ ) с помощью расширенного алгоритма Евклида [3].

*Мультипликативная группа  $\mathbb{Z}_p^*$*

Предположим, что  $G \cong \mathbb{Z}_p^*$  — мультипликативная группа по модулю  $p$ , которая является циклической, и  $\alpha$  — её образующий элемент. Тогда DLP сводится к решению уравнения  $\alpha^d \equiv \beta \pmod{p}$  [3]. На данный момент не существует известных классических алгоритмов, способных решить эту задачу за полиномиальное время.

### 2.3 DLP на эллиптических кривых

Эллиптические кривые над простым полем  $F_p$  образуют конечные абелевы группы. Для точки  $P$  на такой кривой DLP рассматривается в циклической подгруппе, порождённой  $P$ . [3] Проще говоря DLP на кривой, является поиск  $k$ : где  $P$  и  $Q$  — известные точки на эллиптической кривой:

$$Q = kP$$

## **Заключение**

Криптография на эллиптических кривых представляет собой одно из наиболее перспективных направлений современной криптографии. Как показано в статье, уникальные свойства эллиптических кривых, в частности их групповая структура и сложность проблемы дискретного логарифма, создают прочный фундамент для построения криптографических систем.

Особого внимания заслуживает тот факт, что ECC обеспечивает сопоставимый уровень безопасности с традиционными криптосистемами при значительно меньших размерах ключей. Это преимущество становится особенно ценным в условиях развития мобильных технологий и интернета вещей (IoT), где требования к вычислительным ресурсам и энергоэффективности особенно строги.

## Список литературы

1. Aqeel Khalique, K. S. (2010). Implementation of Elliptic Curve Digital Signature Algorithm.
2. Housni, Y. E. (2018). *Introduction to the Mathematical Foundations of Elliptic Curve Cryptography*
3. John Proos, C. Z. (2024). Shor's discrete logarithm quantum algorithm for elliptic curves.
4. Mohammed Shakir Oudah, A. T. (2022). Lightweight Authentication Model for IoT Environments Based on Enhanced . Retrieved from CodedInsights
5. Rahmawati, M. S. (2024). Systematic Analysis of Mathematical Fundamentals in Elliptic Curve Cryptography: Concepts, Applications, and Challenges.
6. Yasin Genç, E. A. (2023). Design and Implementation of an Efficient Elliptic Curve Digital Signature Algorithm (ECDSA).

## Рецензия на статью

# «Криптография на эллиптических кривых: математический фундамент»

### Резюме

Статья посвящена криптографии на эллиптических кривых (ECC), её математическим основам и преимуществам перед традиционными методами, такими как RSA. Автор подробно описывает уравнение эллиптической кривой, операции с точками (сложение, удвоение, вычитание, скалярное умножение) и проблему дискретного логарифма (DLP) в контексте ECC. Подчёркивается, что ECC обеспечивает высокий уровень безопасности при меньших вычислительных затратах, что делает её перспективной для IoT, мобильных устройств и блокчейна.

### Сильные стороны

1. Четкое описание операций с точками эллиптической кривой, включая сложение, удвоение, скалярное умножение — полезно для понимания принципов работы ECC.
2. Сравнение с RSA по длине ключа и вычислительной сложности, что помогает оценить практическую значимость ECC.
3. Актуальная тема: рассмотрение применения ECC в IoT и блокчейн-технологиях.
4. Упоминание стойкости к квантовым атакам, что важно в свете развития квантовых вычислений.

### Критические замечания

1. Недостаточная глубина в объяснении DLP: в разделе 2.3 проблема DLP на эллиптических кривых описана слишком кратко. Стоит добавить примеры или сравнение сложности DLP в разных группах.
2. Отсутствие практических примеров: Статья была бы более наглядной, если бы включала конкретные примеры вычислений или применения ECC в реальных криптосистемах.
3. Поверхностное заключение: в заключении не подводятся итоги по разделам, а лишь повторяются общие тезисы. Желательно добавить выводы по каждому разделу.

### Мелкие правки

1. В разделе 1.2 следует добавить пояснение для случая  $P=Q$  (например, что происходит при сложении точки с собой).

2. Следует уточнить формулу для  $y_3$  при удвоении точек: в текущем виде она содержит  $x_2$ , который не определён.
3. Неполные или некорректные ссылки в списке литературы.  
Рекомендуется проверить соответствие стандарту ГОСТ или АРА.  
Также в некоторых случаях автор пишет «[6]», в других – «[3]» без пробела после номера. Следует унифицировать.

### **Рекомендация редактору**

Предлагаю принять статью после авторской доработки по моим замечаниям. Но при этом на исправлении по взысканным замечаниям рецензент не настаивает. Автор имеет право опубликовать статью в предоставленном им виде.

# Имитационное моделирование беспилотного автомобиля средствами ROS 2

А.М. Рамазанов<sup>1</sup>

<sup>1</sup>Южный федеральный университет, Ростов-на-Дону

**Аннотация:** В данной работе рассматривается создание имитационной модели беспилотного автомобиля с использованием Robot Operating System 2 (ROS 2). Разработка включает создание модели в формате URDF, обеспечивающей точное описание геометрии, кинематики и динамики автомобиля, а также интеграция датчиков: лидара, сонара и камеры.

**Ключевые слова:** беспилотный автомобиль, ROS 2, URDF, имитационная модель, Gazebo, RViz, лидар, сонар, камера, моделирование, симуляция.

## Введение

Данная работа посвящена созданию имитационной модели беспилотного автомобиля средствами ROS 2. В ходе данной работы была разработана модель формата URDF с точным описанием геометрии, кинематики, динамики робота, были рассмотрены средства моделирования, симуляции, визуализации роботов, были интегрированы в ROS 2 и протестированы датчики сонара, лидара и камеры.

Полученные результаты помогут не только в понимании платформы ROS 2, но и в разработке и тестировании автономных систем в виртуальной среде. Это является важным шагом в развитии робототехники и автономных систем.

## Разработка модели

Первым шагом при разработке проекта было создание URDF модели беспилотного автомобиля с реалистичными взаимодействиями отдельных компонентов. Основные элементы модели:

- 1) Шасси – корпус автомобиля
- 2) Четыре колеса: два передних, два задних.

3) Датчики: лидар, сонар, камера. Лидар – датчик, необходимый для обнаружения объектов. Этот датчик будет полезен в будущем, когда надо будет решать задачи по локализации и навигации нашего робота. Сонар – датчик, необходимый для определения расстояния до объектов.

Для создания имитационной модели беспилотного автомобиля вначале нужно определить звенья (links) для каждого компонента робота и соединения - для связывания этих элементов (joints). Схему звеньев четырёхколёсной модели робота можно представить следующим образом (рис. 1).



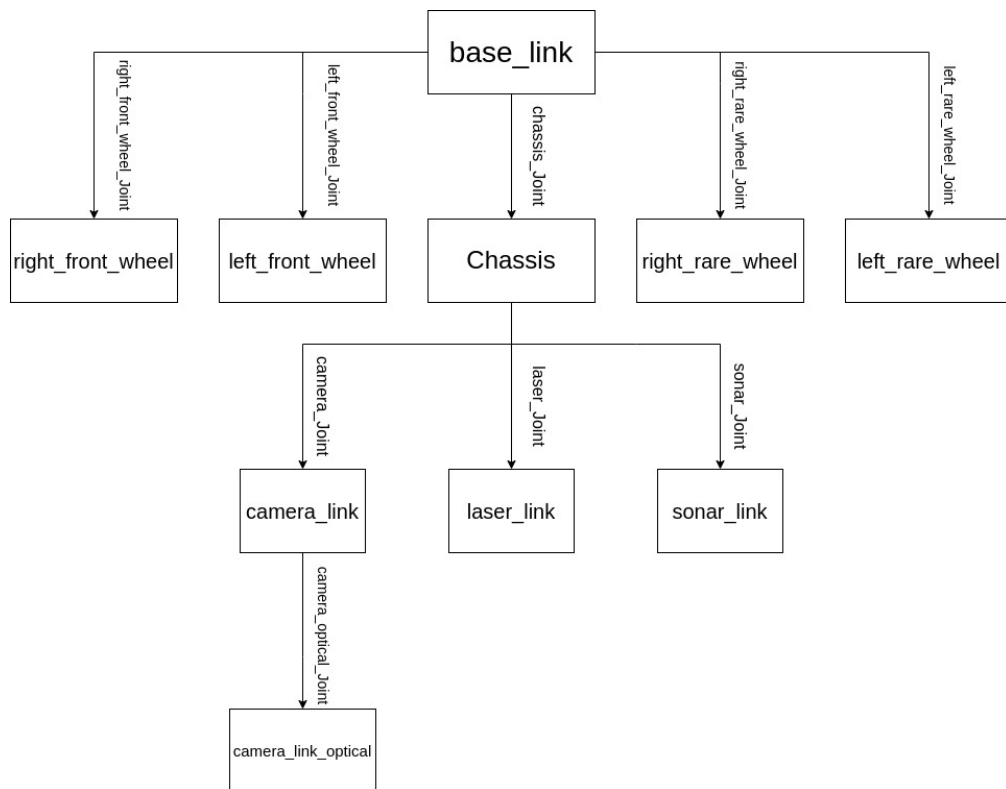


Рис. 1. Схема звеньев имитационной модели автомобиля

Все звенья описаны. Теперь мы можем посмотреть, как модель выглядит, запустим её в Rviz (рис.2). В итоге получена имитационная модель автомобиля, которая по внешнему виду похожа на реальную модель автомобиля.

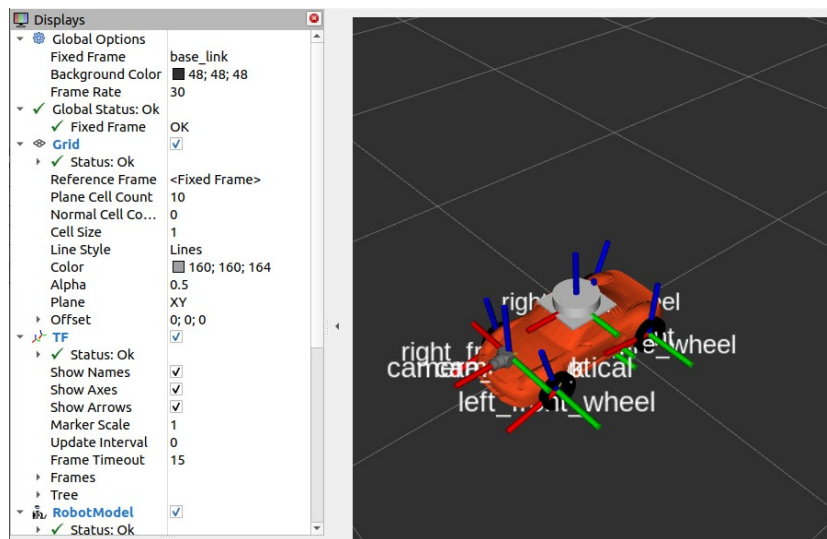


Рис. 2. Визуализация имитационной модели автомобиля в Rviz

## Реализация управления роботом

Следующий этап – реализовать управление моделью. Это нужно не только для настройки компонентов модели, описанных в **urdf** файле, но в будущем поможет для проверки правильности работы датчиков. Для того, чтобы управлять нашим роботом нужно создать отдельный **urdf** файл. Назовём его **gazebo\_control.urdf**. Плагин **libgazebo\_ros\_diff\_drive.so**

используется для управления движения робота с двумя парами колёс, используя дифференциальный привод. Для каждого колеса указывается ссылка на соответствующее звено колеса. Задаются в файле кинематические свойства (такие как диаметр колеса, расстояние между колёс) и лимиты (максимальный крутящийся момент, приложенный к колесам, максимальное ускорение). Необходимо также указать базовое звено робота (в нашем случае это `base_link`).

Плагин `libgazebo_ros_state_publisher.so` публикует данные о состоянии узлов колёс.

Управлять имитационной моделью беспилотного автомобиля с клавиатуры можно с помощью встроенного пакета `teleop_twist_keyboard`. Запустим нашего робота в Gazebo, введём команду в терминал (рис.3), и после этого можно управлять нашим роботом с клавиатуры.

```
artem@artem-Aspire-A317-52:~/mesh_car$ ros2 run teleop_twist_keyboard
teleop_twist_keyboard

This node takes keypresses from the keyboard and publishes them
as Twist/TwistStamped messages. It works best with a US keyboard layout.
-----
Moving around:
  u   i   o
  j   k   l
  m   ,   .

For Holonomic mode (strafing), hold down the shift key:
-----
  U   I   O
  J   K   L
  M   <   >

t : up (+z)
b : down (-z)

anything else : stop
```

Рис.3. Команда для управления роботом с клавиатуры и ответ системы

Управление осуществляется с помощью клавиш клавиатуры: «u»; «i»; «o»; «j»; «k»; «l»; «m»; «,»; «.». Клавишами «q»; «w»; «e» регулируются угловая и линейная скорости. Имитационная модель беспилотного автомобиля поворачивает плавно, без пробуксовки, едет прямо ровно, без смещения.

## Интеграция датчиков

Одним из ключевых этапов разработки имитационной модели беспилотного автомобиля является интеграция датчиков: лидар, сонар, камера. Лидар, сонар и камера ранее в работе были смоделированы и визуализированы в Gazebo. Далее необходимо произвести интеграцию датчиков в имитационную модель автомобиля для обработки данных в режиме реального времени при движении машины.

Теперь мы можем запустить нашу имитационную модель беспилотного автомобиля в Gazebo и убедиться, что датчик сонара, лидара и камеры интегрированы в Gazebo (рис. 4, 5, 6)

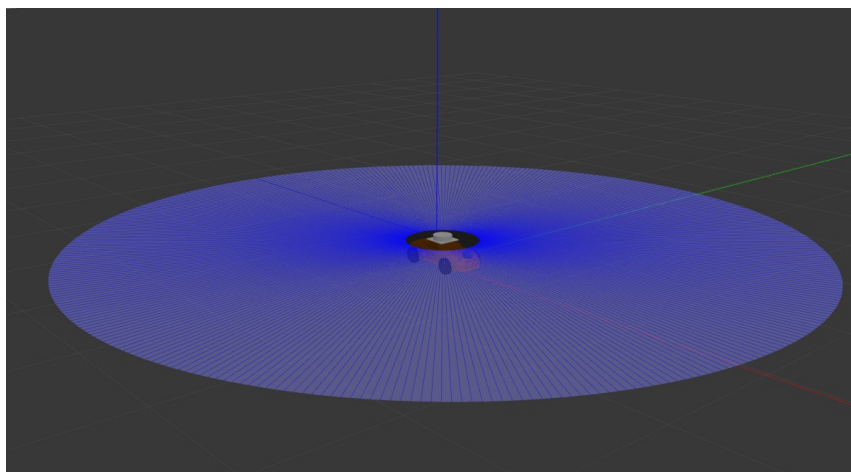


Рис. 4. Интеграция датчика лидара имитационной модели беспилотного автомобиля в Gazebo.

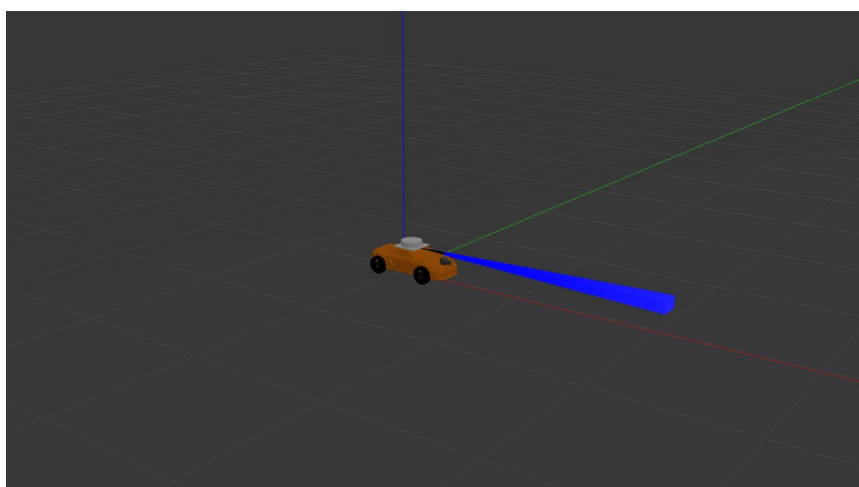


Рис. 5. Интеграция датчика сонара имитационной модели беспилотного автомобиля в Gazebo.

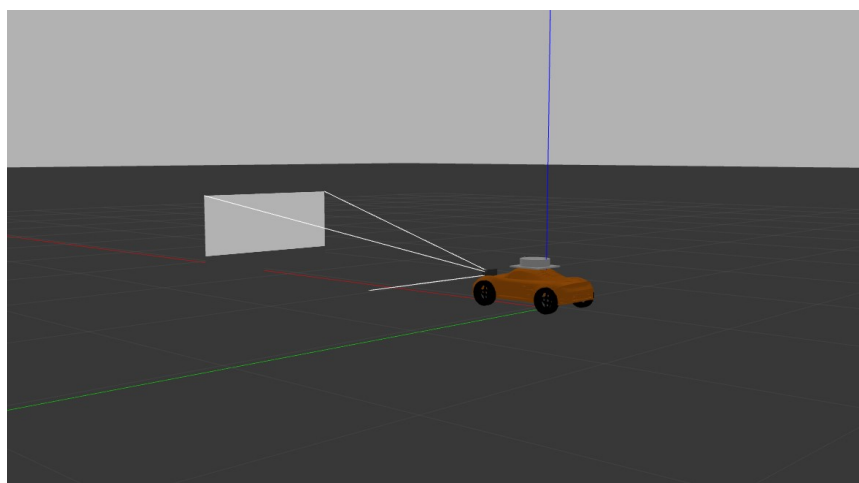


Рис.6. Интеграция камеры имитационной модели беспилотного автомобиля в Gazebo.

## Результаты

В ходе работы была разработана имитационная модель беспилотного автомобиля с использованием средств ROS 2, включающая камеру, датчики лидара и сонара.

В ходе работы было реализовано:

- Разработана имитационная модель беспилотного автомобиля в формате URDF, с корректным позиционированием и настройкой всех компонентов.
- Интегрирована в имитационную модель датчик лидар, сонар, камера
- Реализовано управление беспилотным автомобилем с клавиатуры
- Реализована симуляция разработанной системы средствами ROS 2 в рамках системы Gazebo, выполнен анализ результатов.

Исходный код доступен на платформе GitHub:

[https://github.com/artx52135/mesh\\_car.git](https://github.com/artx52135/mesh_car.git)

## Литература

1. Документация ROS 2 Humble: сайт. – URL: <https://docs.ros.org/en/humble/index.html>
2. ROS Wiki: сайт. – URL: <http://wiki.ros.org/>
3. Tutorial: ROS integration overview // Gazebo: сайт. – URL: [https://classic.gazebosim.org/tutorials?tut=ros\\_overview](https://classic.gazebosim.org/tutorials?tut=ros_overview)
4. Gazebo Answers: сайт. – URL: <https://answers.gazebosim.org/questions/>
5. Patrick Goebel. ROS By Example. A Do-It-Yourself Guide to the Robot Operating System. Volume 1. 2015
6. Damith Herath, Foundations of Robotics: A Multidisciplinary Approach with Python and ROS. 2022
7. Lentin Joseph, Jonathan Cacace. Mastering ROS for Robotics Programming. Third Edition. 2021
8. Nils Leusmann, Arthur Niedzwiecki, Stefan Eirich, Alina Hawkin. Course: Robot Programming with ROS: сайт. – URL: <https://ai.uni-bremen.de/teaching/cs-ros-ws23>
9. Articulated Robotics: сайт. – URL: <https://articulatedrobotics.xyz/>

## **Рецензия на**

### **«Имитационное моделирование беспилотного автомобиля средствами ROS 2»**

*Т. Р. Мамедов<sup>1</sup>*

*<sup>1</sup>Южный федеральный университет, Ростов-на-Дону*

## **Резюме**

Статья представляет собой подробное описание процесса создания имитационной модели беспилотного автомобиля с использованием платформы ROS 2.

В работе последовательно рассматриваются следующие этапы:

- Разработка URDF-модели автомобиля с детальным описанием геометрии, кинематики и динамики
- Создание системы управления моделью через клавиатуру с помощью пакета `teleop_twist_keyboard`
- Интеграция и тестирование различных датчиков: лидара, сонара и камеры
- Визуализация модели в среде Rviz и симуляция в Gazebo

В результате была создана полноценная имитационная модель беспилотного автомобиля, которая может использоваться как для изучения платформы ROS 2, так и для тестирования автономных систем в виртуальной среде. Исходный код проекта доступен на GitHub, что позволяет другим разработчикам воспроизвести результаты и использовать их в своих проектах.

## **Критические замечания**

1. В списке литературы не указаны даты обращения к интернет источникам.
2. Отсутствует блок списка литературы на английском языке (с транслитерализацией).

### **Мелкие правки**

- В основном тексте статьи желательна расшифровка «ROS 2» и «URDF».
- Выделить заголовок «Разработка модели» жирным.
- Исправить: ...В итоге получена ~~имитационной~~ → имитационная.
- Тавтология в предложении «В итоге получена имитационной модель автомобиля, которая по внешнему виду похожа на реальную модель автомобиля.».

### **Рекомендация редактору**

Статья не соответствует требованиям журнала. Требуются серьёзные доработки по оформлению.

# Геймификация в бизнесе.

*Кирчак К.Г.*

*Южный федеральный университет, Ростов-на-Дону*

**Аннотация:** В статье рассматривается геймификация как эффективный инструмент повышения вовлеченности пользователей и конверсии в бизнесе. Приводятся статистические данные о росте вовлеченности на 48% и мотивации на 89% благодаря игровым механикам. Разбираются успешные кейсы (Duolingo, Nike, СберСпасибо), ключевые элементы геймификации (уровни, бейджи, квесты) и технологии для их внедрения (Captain Up, BadgeOS). Подчеркивается, что геймификация — это не просто тренд, а стратегическое преимущество для бизнеса.

**Ключевые слова:** геймификация, вовлеченность пользователей, конверсия, игровые механики, бизнес-адаптивность, Duolingo, Nike,

СберСпасибо, уровни, бейджи, квесты, рейтинги, программы лояльности, Captain Up, BadgeOS, мотивация, e-commerce, онлайн-образование, маркетинг, статистика, лидерборды, прогресс-бар,

челленджи, персонализация, Gamification360, Mambo.IO, вовлекающий контент.



**Введение:**

- эпоху цифровой перегрузки, когда среднестатистический пользователь ежедневно сталкивается с сотнями коммерческих предложений, традиционные методы маркетинга и управления персоналом теряют эффективность. Клиенты стали более искушенными, их внимание сложнее удержать, а мотивация сотрудников требует новых подходов. Именно в этом контексте геймификация превращается из модного тренда в стратегически важный бизнес-инструмент.

Последние исследования подтверждают: компании, внедряющие игровые механики, получают **на 30-50% более высокие показатели вовлеченности** по сравнению с конкурентами, использующими традиционные подходы. Но что делает геймификацию такой эффективной? Ответ кроется в фундаментальных принципах человеческой психологии.

## Психологические основы геймификации

Успех геймификации основан на нескольких ключевых психологических принципах:

### р Дофаминовая мотивация

- q Мозг человека запрограммирован получать удовольствие от достижения целей. Каждый пройденный уровень, полученная награда или улучшение позиции в рейтинге вызывают выброс дофамина - "гормона удовольствия", что формирует позитивную ассоциацию с брендом или продуктом.

- г **Эффект прогрессии:** Человеческий мозг любит видеть свой прогресс. Прогресс-бары, системы уровней и визуализация

достижений удовлетворяют эту базовую потребность, увеличивая вовлеченность.

- s **Социальное доказательство и конкуренция:** Лидерборды, публичные рейтинги и возможность сравнить свои достижения с

результатами других включают мощный социальный механизм мотивации.

- р **Эффект обладания:** Виртуальные валюты, коллекционные элементы и персональные достижения создают у пользователя чувство владения, что значительно повышает лояльность.

## **Расширенный анализ успешных кейсов**

- **Starbucks Rewards: как кофейный гигант увеличил продажи на 20% :** Программа лояльности Starbucks - это блестящий пример геймификации в ритейле. Клиенты получают "звезды" за каждую покупку, которые можно обменивать на бесплатные напитки и десерты. Но настоящий гений системы в ее многоуровневости:

**Два статуса участников** (Green и Gold) с разными привилегиями

**Персональные челленджи** ("купи 3 латте за неделю - получи двойные звезды")

**Сезонные акции** с ограниченными по времени наградами

Результат: участники программы тратят **в 3 раза больше**, чем обычные клиенты.

**2. LinkedIn: геймификация профессионального роста:** Социальная сеть для профессионалов мастерски использует игровые механики:

- **Индикатор полноты профиля** (визуальный прогресс-бар)
- **Система навыков и подтверждений**
- **Рейтинги "Топ-1% специалистов"** в определенных категориях

Эти элементы увеличили **заполняемость профилей на 60%** и время пребывания на платформе на 40%.

### **3. MВидео: геймификация в российском ритейле**

Сеть электроники внедрила мобильное приложение с элементами игры:

- **Викторины о технике** с призами
- **Квесты по магазинам** (найти определенный товар - получить бонусы)
- **Персонализированные рекомендации** в стиле "открой новый уровень знаний"

Это позволило увеличить **количество посещений магазинов на 15%** и средний чек на 8%.

## **Продвинутые механики геймификации**

**1. Динамическая сложность:** Алгоритмы, которые адаптируют сложность заданий под уровень пользователя (как в компьютерных играх), предотвращают скуку от слишком легких задач и разочарование от чрезмерно сложных.

**2. Социальная геймификация:** Интеграция с социальными сетями, где пользователи могут делиться достижениями, создает вирусный эффект и дополнительную мотивацию.

**3. Сторителлинг в геймификации:** Создание нарратива вокруг взаимодействия с брендом (например, "спаси планету, выполняя экозадания") увеличивает эмоциональную вовлеченность.

**4. Гибридные вознаграждения:** Комбинация виртуальных и реальных наград (баллы + скидки + эксклюзивный контент) работает эффективнее, чем однотипные поощрения.

## **Отраслевые особенности внедрения**

**E-commerce**

- Персональные квесты ("собери коллекцию из 5 товаров")
- Виртуальные примерочные с элементами игры
- Система "открой новый уровень покупателя"

## **Образование**

- Интерактивные симуляторы вместо тестов
- Дерево навыков с визуализацией прогресса
- Групповые челленджи для студентов

## **Корпоративное обучение**

- Система бейджей за освоение навыков
- Виртуальные "карьерные лестницы"
- Соревнования между отделами

## **Техническая реализация: от простого к сложному**

### **1. Быстрый старт**

- a. Виджеты (колесо фортуны, мини-квизы)
- b. Готовые плагины для CMS

### **2. Средний уровень**

- a. Интеграция API геймификационных платформ
- b. Кастомизированные программы лояльности

### **3. Продвинутые решения**

- a. AI-адаптивные системы
- b. Блокчейн для учета достижений

## **Ошибки внедрения и как их избежать**



1. **Чрезмерная сложность:** Слишком запутанные правила отталкивают пользователей.
2. **Отсутствие реальной ценности:** Награды должны быть желанными, а не формальными.
3. **Игнорирование аналитики:** Недостаточно просто внедрить механики - нужно постоянно анализировать их эффективность.
4. **Нарушение UX:** Игровые элементы должны органично вписываться в пользовательский опыт.

## **Будущее геймификации**

### **1. Метавселенные:**

Виртуальные миры станут новой ареной для геймифицированного взаимодействия с брендами.

### **2. Нейротехнологии :** Адаптация игровых механик под индивидуальные паттерны поведения мозга.

### **3. Геймификация ИИ:** Персональные ИИ-ассистенты, которые превращают рутинные задачи в увлекательные квесты.

## **Заключение: геймификация как новая парадигма бизнеса**

Геймификация перестала быть просто инструментом маркетинга - она становится новой философией взаимодействия с клиентами и сотрудниками. В ближайшие 5-7 лет мы увидим, как границы между "игрой" и "работой", "развлечением" и "обучением" будут стираться еще сильнее.

Компании, которые уже сегодня инвестируют в развитие геймификационных стратегий, закладывают фундамент для долгосрочного конкурентного преимущества. Важно помнить: успешная геймификация - это не просто набор механик, а глубокое понимание психологии своей целевой аудитории и искреннее желание сделать их взаимодействие с брендом по-настоящему увлекательным.

Литература

1. Werbach, K., & Hunter, D. For the Win: How Game Thinking Can Revolutionize Your Business— базовое руководство по применению игровых механик в бизнесе.
2. Марков А., Иванов Д. "Геймификация в цифровом маркетинге" (рус.) — адаптация под российский рынок.
3. Gartner Reports - отчеты по трендам в корпоративной геймификации.
4. Журнал "Harvard Business Review" — статьи о геймификации в менеджменте.

## Резюме

Статья посвящена анализу геймификации как эффективного инструмента повышения вовлечённости пользователей и конверсии в бизнесе. Автор рассматривает ключевые игровые механики, такие как уровни, бейджи, квесты, и приводит успешные кейсы (Duolingo, Nike, СберСпасибо), подкреплённые статистическими данными. Отмечается стратегическое значение геймификации для современных бизнес-моделей, а также раскрываются психологические механизмы, лежащие в её основе. Представленные данные обоснованы, но методологическая часть требует более глубокого анализа. В статье перечислены инструменты внедрения игровых механик, однако отсутствует подробный разбор их интеграции в бизнес-процессы.

## Критические замечания

1. В разделе «Психологические основы геймификации» описание принципов дофаминовой мотивации и эффекта прогрессии представлено поверхностно. Требуется более глубокое объяснение механизмов воздействия на мотивацию пользователей.
2. Упомянутые платформы для геймификации (Captain Up, BadgeOS) не сопровождаются описанием их ключевых функций и возможностей. Это может затруднить понимание их практического применения.
3. В разделе «Техническая реализация» недостаточно раскрыта тема интеграции игровых механик в существующие бизнес-системы. Стоило бы добавить конкретные примеры внедрения API и плагинов.
4. В «Расширенном анализе успешных кейсов» представлены только положительные результаты, без упоминания о возможных трудностях или ограничениях. Это создаёт одностороннее представление.
5. Перечисленные в разделе «Отраслевые особенности внедрения» механики геймификации описаны слишком общими терминами. Было бы полезно увидеть более детализированный анализ для каждого из представленных направлений.

## Мелкие правки

- На странице с кейсом Starbucks Rewards не хватает ссылок на источники данных, подтверждающих рост продаж на 20%.
- В разделе «Будущее геймификации» пункт о метавселенных требует более конкретных примеров текущих реализаций.

## Рекомендация редактору

Требуется доработка методологической части и более глубокий анализ представленных технологий. Рекомендуется добавить практические примеры интеграции игровых механик в бизнес-процессы для лучшей наглядности.