МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ИНФОРМАТИКА И КИБЕРНЕТИКА

4 (34)

Донецк – 2023

УДК 004.3+004.9+004.2+51.7+519.6+519.7

ИНФОРМАТИКА И КИБЕРНЕТИКА, № 4 (34), 2023,
Донецк, ДонНТУ.

Выпуск подготовлен по материалам VIII Всероссийской научно-технической конференции «Современные информационные технологии в образовании и научных исследованиях» (СИТОНИ-2023), проведенной 29–30 ноября 2023 г., а также текущей научно-технической деятельности аспирантов, соискателей и научных работников. Статьи посвящены вопросам приоритетных направлений научно-технического обеспечения в области информатики, кибернетики, вычислительной техники и инженерного образования.

Материалы предназначены для специалистов народного хозяйства, ученых, преподавателей, аспирантов и студентов высших учебных заведений.

Рекомендовано к печати ученым советом ФГБОУ ВО «Донецкий национальный технический университет» Министерства науки и высшего образования РФ. Протокол № 9 от 22 декабря 2023 г.

Свидетельство о регистрации СМИ: серия ААА № 000145 от 20.06.2017.

Приказ МОН ДНР № 135 от 01.02.2019 о включении в Перечень рецензируемых научных изданий ВАК ДНР.

Контактный адрес редакции
РФ, ДНР, 283001, г. Донецк, ул. Артема, 58, ФГБОУ ВО «ДонНТУ»,
4-й учебный корпус, к. 36., ул. Кобозева, 17.
Тел.: +7 (856) 301-07-35, +7 (949) 334-89-11
Эл. почта: infcyb.donntu@yandex.ru
Интернет: http://infcyb.donntu.ru

СОДЕРЖАНИЕ

Информатика и вычислительная техника

Информатика и вычислительная техника

УДК 004.94

Алгоритмы генеративного моделирования в формообразовании промышленных изделий

М.П. Руденко, Д.Е. Звягинцев
Донецкий национальный технический университет
e-mail: m.p.rudenko@mail.ru, zvyaginsevdmitriy00@gmail.com

Аннотация
В статье приведен анализ алгоритмов генеративного моделирования в формообразовании промышленных изделий на примере работы методов оптимизации топологии и агентного моделирования. Определены преимущества и недостатки данных методов при исследовании следующих параметров: возможность автоматизации проектирования, многовариантность, проверка на прочность и устойчивость полученной модели, время генерации модели, а также необходимость дальнейшей доработки модели. Дальнейшей целью исследования является процесс совершенствования алгоритмов генеративного моделирования на основе полученных результатов.

Введение

На сегодняшний день возможности генеративного моделирования достаточно широки. Процесс автоматического проектирования модели в промышленном дизайне значительно упрощает работу дизайнерам в поиске новых форм, которые бы удовлетворяли эстетическим и эргономическим целям.

Генеративное моделирование ускоряет все этапы разработки продукта – от концептуального дизайна до производства. Используя цифровые инструменты, архитекторы и дизайнеры могут быстро генерировать геометрии с высокой сложностью, от органических деталей свободного потока до повторяющихся шаблонов с миллионами элементов. Поскольку технологичность модели можно указывать уже на ранних этапах процесса проектирования, вероятность того, что позже потребуются трудоемкие пересмотры, намного ниже. За счет этого снижаются трудо- и времязатраты на производство и выход на рынок.

Постановка задачи

Проведя анализ публикаций, можно сделать вывод, что алгоритмы генеративного моделирования активно используются в архитектуре и проектировании [1-4], а также в промышленном дизайне [5-7].

Генеративное моделирование в формообразовании промышленных изделий включает в себя следующие компоненты:

– формулировка проблемы (абстрагирование идеи);

– кодирование алгоритма решения проблемы;

– запуск генерации проектных решений;

– оценка результатов генерации;

– получение конечного результата и завершение генерации.

Исходные данные имеют распределение: p_data (x), а цель генератора - создать данные, соответствующие этому распределению. Генератор получает данные, а затем выводит их в распределении p_data (x).

Далее дискриминатор определяет, какие изображения из исходного распределения p_data (x), а какие – из генератора. Обе функции потерь зависят от производительности друг друга.

Генеративные модели, такие как вариационные автоэнкодеры (VAE) и генеративные состязательные сети (GAN) позволяют получать выходные данные более высокого разрешения и за счет кодирования в пространстве низкой размерности.

Однако есть ряд ограничений для использования в формообразовании генеративных моделей, а именно:

– генеративным моделям чаще всего необходимы конфиденциальные или труднодоступные данные;

– генеративные модели не гарантируют правдоподобного проектирования;

– генеративные модели страдают от коллапса режима - когда они выводят только один тип результата / имеют большой разброс в качестве вывода.

Совершенствование методов алгоритмической генерации описано в [8].

Структура генеративного моделирования показана на рисунке 1.

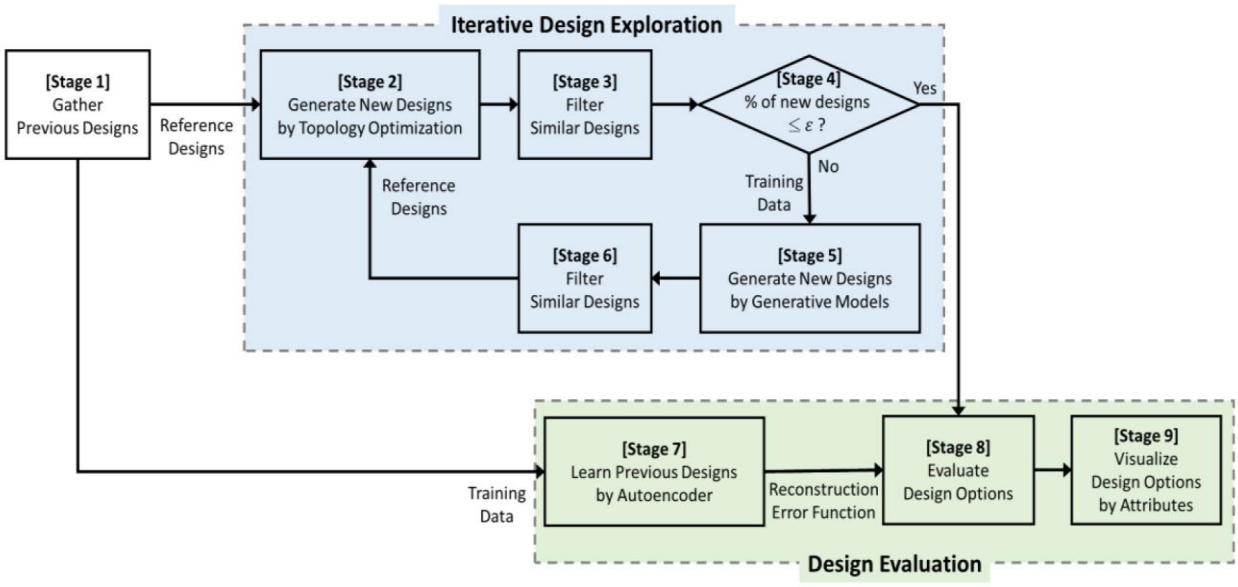


Рисунок 1 – Структура генеративного моделирования

Методы оптимизации топологии и агентного моделирования

В целях анализа алгоритмов генеративного моделирования при формообразовании промышленных изделий приведены два метода, наиболее часто используемых в среде архитектурного и дизайн проектирования [9].

Один из методов генеративного моделирования, который хорошо зарекомендовал себя в структурной оптимизации и используется для проектирования инженерных концепций, называется оптимизацией топологии [10]. При оптимизации топологии конструктор сначала определяет пространство проектирования, условия загрузки, цель оптимизации и другие нетехнические требования, такие как производственные ограничения. В результате модель имеет уменьшенный вес и высокую жесткость.

Кроме того, можно комбинировать оптимизацию топологии с другими методами генеративного проектирования. Следуя гибридному подходу, можно разработать, например, полые компоненты с переменной толщиной, которые имеют повышенную ударопрочность.

Действия, образованные несколькими взаимодействующими интеллектуальными агентами, формируют метод агентного моделирования [11]. Основная цель этого метода в промышленном формообразовании заключается в генерации оптимизированных структур на основе исходных параметров путем внедрения самоорганизующихся процессов.

Существует два вида агентного моделирования – стигмергические и роевые системы. Выбор зависит от исходного объекта, с которого была списана модель поведения алгоритма [12]. Данный алгоритм значительно отличается от метода оптимизации топологии, так как на начальном этапе нет исходной геометрии.

Программная реализация метода оптимизации топологии

Реализация метода оптимизации топологии выполняется с помощью инструмента генеративного проектирования Fusion 360.

Для проведения эксперимента взята модель стула со следующими параметрами:

– высота – 900 мм;
– ширина – 500 мм;
– глубина – 980 мм.

Сам алгоритм генерации в Fusion 360 состоит из следующих этапов, изображенных на рисунке 2.

Рисунок 2 – Схема алгоритма генерации

Процесс генерации модели стула по заданным параметрам показан на рисунке 3.

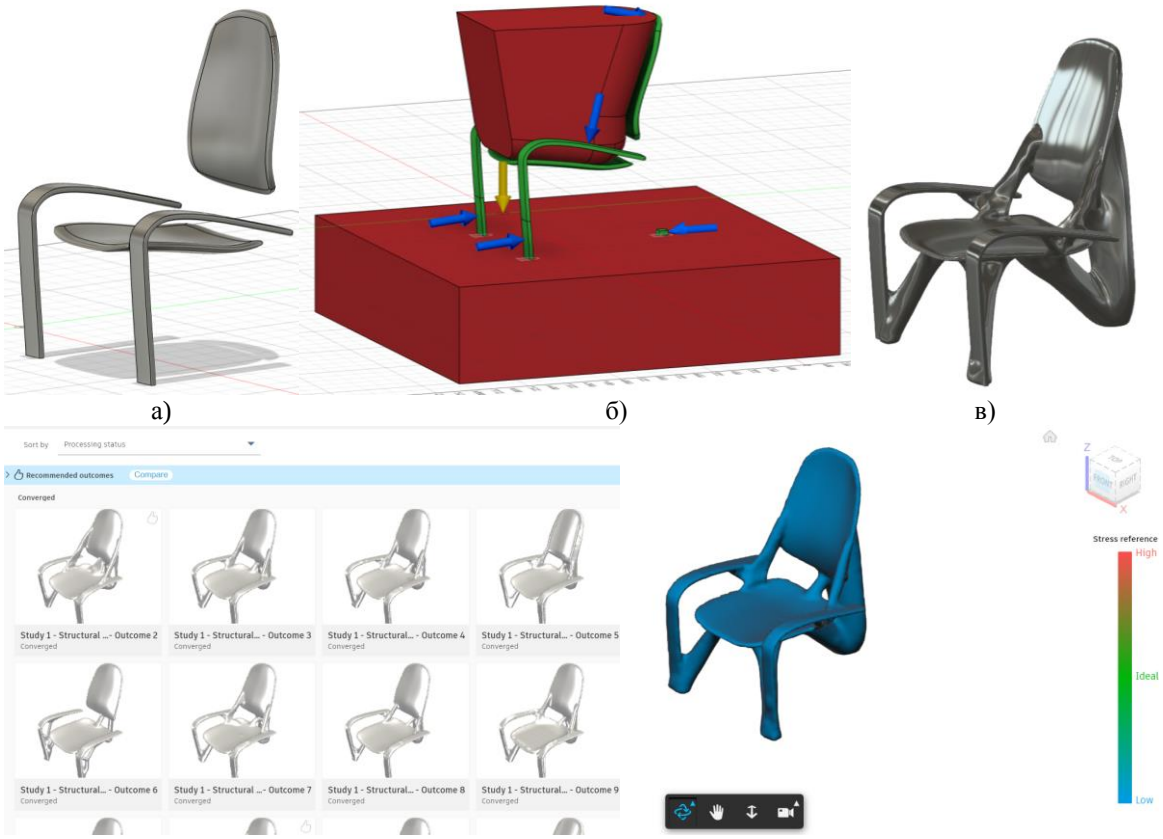


Рисунок 3 – Процесс генерации стула методом оптимизации топологии: а) обобщенная модель стула; б) задание ограничений для генерации; в) сгенерированная модель; г) варианты генераций; д) проверка модели на прочность

После создания обобщенной модели стула с необходимыми параметрами (рис. 3, а) необходимо задать ограничения, в рамках которых будет производиться генерация (рис. 3, б). Зеленым цветом указываются элементы, которые необходимо соединить между собой, ограничения указываются красным цветом. При помощи синих стрелок указываются направления нагрузки на модель и ее силу.

Далее выбирается материал из имеющихся, например, для этой модели подойдут пластик и алюминий. После всех указания всех ограничений запускается процесс генерации дизайна, который может занимать разное время в зависимости от сложности исходной модели.

При запуске генерации модели ожидаются следующие результаты:

– оптимизация материалов;
– легкость и прочность;
– эргономика;
– дизайн с учетом функциональности;
– эстетика и уникальность;
– быстрота и эффективность проектирования;
– инновации в конструкции.

Модель генерировалась 3 часа. В результате программа выдала 34 разных генерации, из которых выбирается одна, наиболее удачная (рис. 3, в). Вариант наиболее удачной модели стула по результатам генерации показан на рисунке 3, г. Модель сгенерировалась достаточно качественно, однако ее необходимо дорабатывать вручную, что тоже займет некоторое время.

После генерации даются результаты проверки модели на прочность в виде шкалы, где синий – модель прочная, красный – модель неустойчивая. Результаты данной модели находятся в пределах ограничений, что указывает на ее прочность (рис. 3, д).

Оптимизация модели в данной программе позволяет достичь оптимального сочетания легкости и прочности, что является ключевым аспектом в современном дизайне мебели. Благодаря легкости в использовании программы и широкому выбору генеративных моделей, дизайнеры получают возможность творчески выражаться и находить инновационные решения.

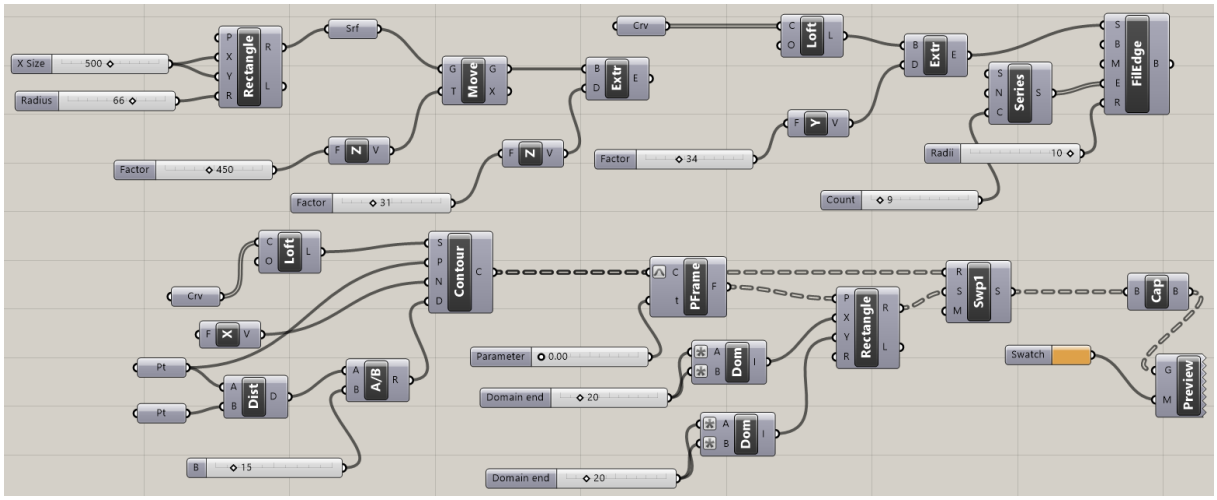
Следует отметить, что процесс не лишен определенных ограничений. Необходимость построения исходной модели ограничивает полную автоматизацию проектирования, а долгая генерация сложных элементов и

требование частичной доработки после генерации могут замедлить сам процесс проектирования.

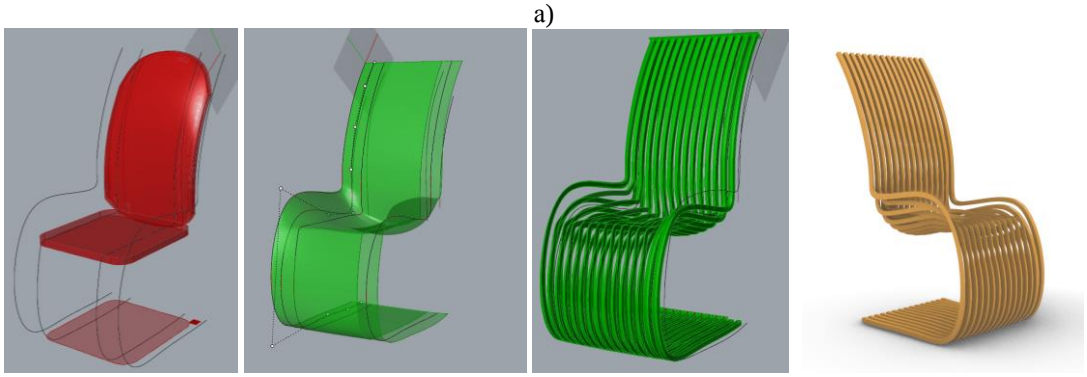
Программная реализация метода агентного моделирования

Реализация метода агентного моделирования выполняется в программном продукте Rhino Grasshopper. Для проведения эксперимента, а также сравнения результатов выбирается модель стула с теми же параметрами и каркасом. Сам алгоритм генерации здесь отличается кардинально, так как построение модели полностью управляется экспертом, а варианты генерации модели задаются изначально с помощью сценариев, созданных с применением визуального программирования.

Модель стула выглядит более обобщенно, так как для последующей генерации необходимы только общие параметры в виде каркаса. Однако для возможности редактирования в режиме реального времени она также программируется с помощью специальных нодов, позволяющих менять различные параметры модели, не нарушая общую геометрию. Процесс генерации модели стула показан на рисунке 4.



а)



б) в) г) д)

Рисунок 4 – Процесс генерации стула методом агентного моделирования: а) алгоритмы генерации модели; б) этап формирования общих элементов; в) этап лофтинга; г) этап конечного проектирования; д) визуализация модели

Процесс моделирования выполняется с помощью визуального программирования с применением специальных нодов (Рис. 4а). Эскизирование происходит перед началом генерации в программе, поэтому эксперт, начиная работу, уже знает, какой будет модель в конечном виде.

На примере одного из вариантов дизайна, показаны этапы его генерации от моделирования общих форм до конечной визуализации (Рис. 4б,в,г,д).

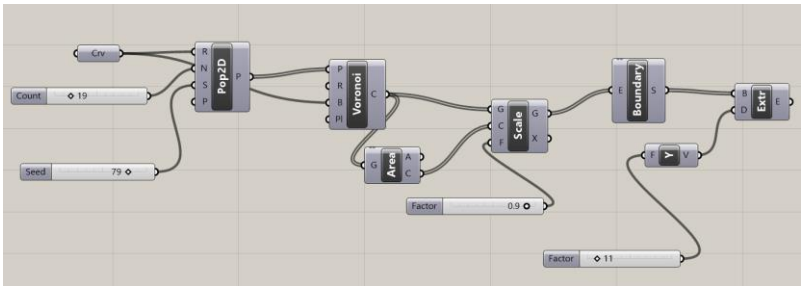
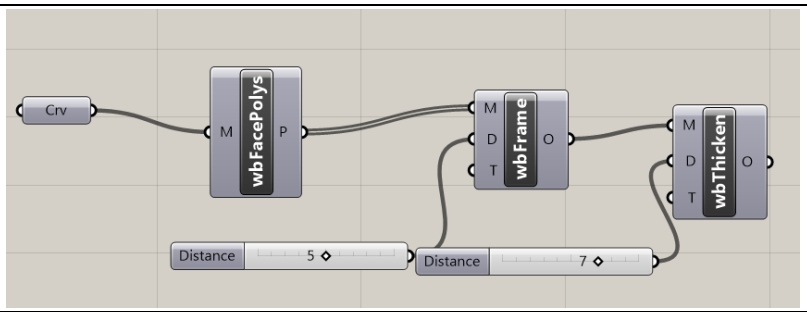
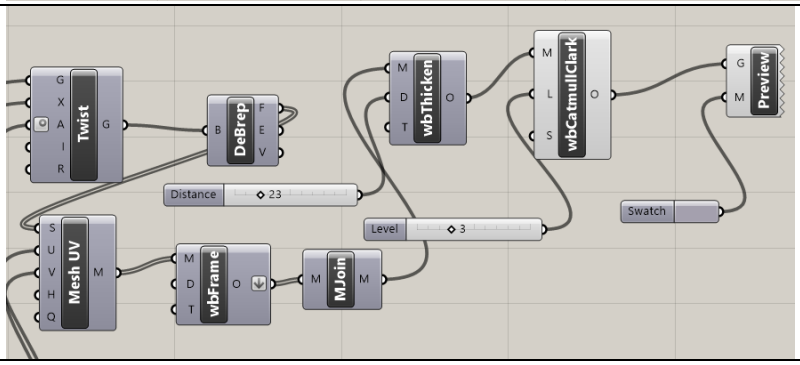
Анализируя алгоритм, показанный на рисунке 4, следует отметить основные параметры, влияющие на генерацию модели:

– Curve – управление и редактирование кривых;
– Contour – управление количеством планок на данной модели, а также исходных точек;
– Rectangle – управление формой контура;
– Preview – управление цветом и материалом модели.

При генерации модели с более сложной геометрией применяются плагины, позволяющие в режиме реального времени кардинально менять структуру, не нарушая при этом общую геометрию. Варианты дизайна создаются в виде алгоритмов, которые можно скрывать, при этом не создавая несколько копий одной и той же модели. Изменяющиеся параметры в алгоритмах визуального проектирования при помощи слайдеров значительно облегчают работу эксперта, уменьшая его время на генерацию очередного варианта дизайна, а встроенные плагины дают возможность различных идей при формообразовании объекта.

Применение популярных плагинов при создании дизайна стула показано в таблице 1, где алгоритм 1 – Voronoj, алгоритм 2 – Weaverbird в нескольких модификациях (спинка и основание), алгоритм 3 – Weaverbird Plane.

Таблица 1. Алгоритмы при создании нескольких вариантов одной модели

№	АЛГОРИТМ	МОДЕЛЬ
1		
2		
3		

Проверка адекватности

В результате проведенного эксперимента можно сделать вывод о том, что метод оптимизации топологии и метод агентного моделирования кардинально отличаются как на начальных этапах генерации, так и в получении конечного результата.

Метод оптимизации топологии, реализованный в программе Fusion 360, на начальном этапе требует построения исходной обобщенной модели стула с указанием необходимых параметров, а также ограничений, в границах которых будет производиться генерация модели. Конечный результат генерации выдает программа при помощи обученных нейронных сетей, который в дальнейшем необходимо редактировать. Варианты дизайна, предлагаемые программой, являются похожими и не особо отличающимися по геометрии, что не дает выбора эксперту для дальнейшей работы. Однако преимуществом программы является проверка сгенерированной модели на прочность.

Метод агентного моделирования, реализованный в программе Rhino Grasshopper, предполагает полное управление генерации экспертом, что исключает автоматизацию процесса. Однако, построение алгоритмов с параметрами, которые можно редактировать, включать/выключать и заменять в режиме реального времени, не нарушая общей геометрии модели, на выходе дает законченную модель, готовую к производству.

Сравнение работы методов, а также характеристики получаемых моделей генерации показаны в таблице 2.

Таблица 2. Сравнение методов оптимизации топологии и агентного моделирования при генерации модели

Основные характеристики	Метод оптимизации топологии (Fusion 360)	Метод агентного моделирования (Rhino Grasshopper)
Требования к исходной модели	Обощенная модель с заданными параметрами и ограничениями	Эскиз будущей модели
Время генерации (час)	3	1 -2 на одну генерацию
Автоматизация процесса генерации	Есть	Нет
Количество сгенерированных моделей (шт)	20-30	Определяется экспертом
Варианты дизайна полученной модели	Однотипное стилевое решение	Разнообразные варианты
Дальнейшая доработка	Требуется	Не требуется
Проверка на прочность полученной модели	Встроена в программу	Отстутствует

Выводы

В ходе исследования методом экспериментального проектирования модели стула были проанализированы в работе метод оптимизации топологии и метод агентного моделирования. Определены преимущества и ограничения методов. Основными преимуществами метода оптимизации топологии являются автоматизация проектирования с помощью обученных нейросетей, многовариантность сгенерированных моделей, а также проверка на прочность и устойчивость полученной модели стула.

Среди недостатков следует отметить долгое время процесса генерации, метод «черного ящика» при получении моделей, ограниченность в стилевом решении дизайна модели, а также необходимость дальнейшей доработки модели. Преимуществом метода агентного моделирования является полное управление процессом генерации экспертом, качественная конечная модель, не требующая доработки, различные варианты дизайна. Из недостатков следует отметить полное отсутствие автоматизации генерации и невозможность проверки на прочность модели.

Дальнейшей целью исследования является процесс совершенствования алгоритмов генеративного моделирования на основе полученных результатов.

Литература

1. Комарова, А. А. Образование архитектурной формы с применением алгоритмических методов /А. А. Комарова, С. В. Пыхтюк, Д. А. Чернышов, М. Е. Дымченко // Инженерный вестник Дона, 2019. - № 8 (59). – С. 9.

2. Салех, М. С. Внедрение цифровых методов на различных этапах цифрового проектирования / М. С. Салех // Архитектура и современные информационные технологии, 2021. - № 1 (54). – С. 268-278.

3. Бжахов, М. И. Алгоритмическое проектирование в архитектуре / М. И. Бжахов, М. М. Ефимова, А. В. Журтов // Инженерный вестник Дона, 2018. - № 2 (49). – С. 166.

4. Потапенко, А. А. Алгоритмическое проектирование как средство формирования аналитических и проектных моделей в архитектуре / А. А. Потапенко // Архитектура и дизайн: история, теория и инновации, 2021. - № 5. – С. 307-311.

5. Бессарабова, Е. В. Трехмерное моделирование промышленных и архитектурных объектов / Е. В. Бессарабова // Современные наукоемкие технологии, 2016. - № 4-2. – С. 225-229.

6. Chen, A. Forte: User-Driven Generative Design / A. Chen, Y. Tao, R. Kang, G. Wang, T. Grossman, S. Coros, S. E. Hudson // CHI 2018, April 21–26, 2018, Montréal, QC, Canada. – P. 496. Retrieved from https://www.researchgate.net/publication/324671677_Forte_User-Driven_Generative_Design.

7. Li, J. Robiot: A Design Tool for Actuating Everyday Objects with Automatically Generated 3D Printable Mechanisms / J. Li, J. Kim, X. Chen // UIST '19, Session 6A: Fabrication, October 20–23, 2019, New Orleans, LA, USA. – P. 673-685. Retrieved from https://arxiv.org/pdf/2007.11199.pdf.

8. Oh, S. Deep Generative Design: Integration of Topology Optimization and Generative Models / S. Oh, Y. Jung, S. Kim, I. Lee, N. Kang // Journal of Mechanical Design, 2019. - Vol. 141 (11). - P. 111405-1-13. Retrieved from https://www.researchgate.net/publication/334472895_Deep_Generative_Design_Integration_of_Topology_Optimization_and_Generative_Models/link/60269ee0a6fdcc37a8217a39/download.

9. Руденко, М. П. Методы генеративного моделирования в промышленном дизайне / М. П. Руденко, Д. Е. Звягинцев // Программная инженерия: методы и технологии разработки информационно-вычислительных систем (ПИИВС-2022): сборник научных трудов IV научно-практической конференции (студенческая секция), Том 2, 29-30 ноября 2022 г. – Донецк, ГОУВПО «Донецкий национальный технический университет», 2022. – С. 151-154.

10. Титова, М. А. Генеративный дизайн на основе оптимизации топологии с использованием глубокого обучения / М. А. Титова, А. Ю. Громов // Известия ТулГУ. Технические науки, 2022. – № 2. - С. 246-248.

11. Hattab M., Hamzeh F. Analyzing Design Workflow: An Agent-based Modeling Approach // Procedia Engineering, 2016. - Vol. 164. – P. 510-517. - Retrieved from https://www.researchgate.net/publication/311360797_Analyzing_Design_Workflow_An_Agent-based_Modeling_Approach

12. Stieler, D. Agent-based modeling and simulation in architecture / D. Stieler, T. Schwinn, S. Leder, M. Maierhofer, F. Kannenberg, A. Menges. // Automation in Construction, 2022. - 141, 104426. Retrieved from https://doi.org/10.1016/j.autcon.2022.104426

Руденко М.П., Звягинцев Д.Е. Алгоритмы генеративного моделирования в формообразовании промышленных изделий. *В статье приведен анализ алгоритмов генеративного моделирования в формообразовании промышленных изделий на примере работы методов оптимизации топологии и агентного моделирования. Определены преимущества и недостатки данных методов при исследовании следующих параметров: возможность автоматизации проектирования, многовариантность, проверка на прочность и устойчивость полученной модели, время генерации модели, а также необходимость дальнейшей доработки модели. Дальнейшей целью исследования является процесс совершенствования алгоритмов генеративного моделирования на основе полученных результатов.*

Ключевые слова: *генеративное моделирование, промышленный дизайн, оптимизация топологии, агентное моделирование, визуальное программирование, алгоритмы.*

Rudenko M.P., Zvyagintsev D.E. The generative modeling algorithms in the industrial products shaping. *The article provides an analysis of generative modeling algorithms in the shaping of industrial products by the example of topology optimization methods and agent-based modeling. The advantages and disadvantages of these methods in the study of the following parameters are determined: the possibility of design automation, multivariance, testing for strength and stability of the resulting model, the time of model generation, as well as the need for further refinement of the model. The further purpose of the study is the process of improving generative modeling algorithms based on the results obtained.*

Keywords: *generative modeling, industrial design, topology optimization, agent-based modeling, visual programming, algorithms.*

Статья поступила в редакцию 20.11.2023
Рекомендована к публикации доцентом Карабчевским В. В.

УДК 004.02

Определение позиции робота по сформированной двумерной карте помещения

Т. В. Завадская, О. И. Креков
Донецкий национальный технический университет, г. Донецк
tatyana.zavadskaja@gmail.com
OneGlaxyInTheSky@ya.ru

Аннотация
Работа посвящена задаче локализации объекта по заранее известной двумерной карте помещения. Авторами рассматривается предлагаемый метод определения местоположения робота, не требующий хранения карты в памяти устройства, что позволяет организацию вычислений на сервере. Метод математически описан в двух его вариантах: с отслеживанием хода измерительного луча и без него. Приводятся результаты моделирования определения предполагаемых позиций объекта по задаваемым измеренным значениям, а также время выполнения методов в зависимости от изменения разрешения карты.

Введение

Человек издавна размышлял над созданием различных механизмов: человекоподобных слуг, устройств, и статуй. Так, в древнегреческой мифологии имеется одно из первых упоминаний робота — медный человек-великан Талос, охранявший девушку на острове [1]. В XVIII веке автоматоны приобрели наибольшую популярность в интересах развлечений. Тогда были созданы такие известные механические игрушки, как фигуры людей, выполняющие человеческие действия (игра на музыкальных инструментах, письмо), утка с симуляцией пищеварения и махания крыльями, театр из двух сотен фигур [2].

В современном виде робототехника формировалась в середине прошлого века, прежде всего благодаря развитию цифровой вычислительной техники. В это раз потребность в роботах возникла из-за необходимости в помощи либо замене человека в опасных, либо недоступных ему условиях, а также по экономическим причинам. Первые механизмы (1940–1950 гг.) повторяли на расстоянии действия рук человека за счет его собственных усилий. Со временем появлялись новые способы управления, в частности манипуляторы стали автоматизированными и программируемыми [3].

Следующими важными направлениями в робототехнике – создание мобильных и автономных роботов, что привело к началу работы и появлению в 1965–1972 гг. первого мобильного робота Shakey [4], способного самостоятельно ориентироваться в неизвестной среде и перемещаться в ней к заданной точке.

В настоящее время ведется работа по добавлению новых датчиков роботам и совершенствованию искусственного интеллекта, позволяющие им выполнять все новые задачи [5]. Уже сейчас существуют роботы, обучаемые действиям на примере человека вместо их программирования [6].

В мобильных автономных роботах важное место занимает задача навигации. В этой задаче можно выделить следующие подзадачи: построение карты и локализация робота на ней. В работе будет рассматриваться последняя.

Объектом исследований выступает процесс локализации объекта в пространстве.

Предмет исследований: определение позиции объекта на карте по результатам его измерений.

Целью работы является исследование предлагаемых методов определения координат объекта на сформированной карте по результатам измерений, выполняемых объектом.

Представление карт

Карты разделяют на глобальные и локальные. Глобальные состоят из нескольких локальных карт и позволяют рассчитать местоположение робота в глобальной системе координат. С помощью локальных карт зачастую определяются относительные координаты устройства.

Также выделяют карты по способу хранения информации о внешней среде:

– топологическая — информация представляется в виде графа;

– дискретная метрическая — внешняя среда представлена в виде ячеек, содержащих информацию о внешней среде;

– непрерывная метрическая карта — карта содержит перечень геометрических объектов (таких как точки и линии), значения координат которых принадлежат к непрерывному числовому диапазону [7].

Еще карты можно классифицировать по числу измерений: двумерные и трехмерные. Использование трехмерных карт сильно повышает требования к вычислительным мощностям и объему памяти устройства, характеристикам сканирующего модуля.

Робот может использовать заранее подготовленные карты либо самостоятельно их строить и одновременно определять свое местоположение.

Системы навигации

Системы навигации классифицируют:

– пассивные — прием информации о собственных координатах и других характеристиках своего движения от внешних источников (метки, маяки, проложенные полосы, GPS и т.д.);

– активные — определение местоположения только своими силами (одометр, радары, сонары, лидары и т.д.) [8].

В навигации частое применение находят лазерные дальномеры за счет высокой точности и скорости измерений. Однако их недостатками являются наличие зависимостей от отражательной способности объекта и постороннего освещения в среде, вызывающих трудности с обнаружением тел прозрачных или с зеркальной поверхностью, а также в условиях высокой освещенности, плохой погоды. Перспективны и также используются технологии компьютерного зрения, которые в сравнении с лидаром имеют меньшую стоимость измерительного модуля и дают больше информации о самом объекте, однако сталкиваются с проблемами определения объектов, точности расчета расстояния, работы при наличии видимых помех (плохая погода, освещенность), требования высокой вычислительной мощности для обработки изображения [8, 9, 10, 11].

Принятые допущения

В работе используется двумерная дискретная метрическая карта и активная навигация. Приняты следующие допущения:

1. Карта точно соответствует описываемому помещению.
2. Карта ориентирована по географическому северному полюсу.
3. В момент измерения платформа неподвижна либо время измерений бесконечно мало.
4. Используемые датчики выполняют измерения без ошибок и с разрешением, равным разрешению карты.

Математическое описание

Рассматриваемый способ нахождения местоположения робота предполагает использование 2D лидара для определения дистанции d до предполагаемого препятствия и магнитометра для определения азимута α (принимается географическим).

Суть метода заключается в следующем:

1. По результатам измерения датчиков относительно препятствий строится обратный путь измерительного луча.
2. Точки получаемого множества, где путь заканчивается, считаются предполагаемыми местами робота.
3. Имея четыре множества, формируемые измерениями по азимуту с углами 0°, 90°, 180°, 270° и соответствующими измеренными дистанциями, находится их пересечение, сильно сокращающее число возможных позиций устройства.

В описываемой модели можно выделить два ключевых объекта: двумерная карта помещения и предполагаемые позиции робота.

Карта (G) представляется в виде матрицы:

$$G = \{g_{x,y} \mid x = \overline{0, n-1},\ y = \overline{0, m-1}\}, \quad (1)$$

где $g_{x,y} = G(x, y)$ — ячейка двумерной карты G,

x — номер столбца карты,

y — номер строки,

n — количество столбцов,

m — количество строк.

Переменная g характеризует состояние соответствующей ячейки: -1 — состояние ячейки неизвестно (не сканирована), 0 — ячейка свободна, 1 — ячейка имеет препятствие. Определяемый ячейкой размер равен значению точности карты.

Нумерация по оси x происходит слева направо, y — сверху вниз. Столбцы и строки матрицы нумеруются начиная с верхнего левого угла. Если не указано противное, элементы g принадлежат множеству G.

H_α — гипотезы о возможных местах робота относительно препятствий по результатам измеренных углов и дистанций, где α — азимут. Построение гипотез для направлений 0°, 90°, 180°, 270° с учетом хода луча будет следующим:

$$
\begin{aligned}
H_0 &= \{g_{x,y+d0} \mid g_{x,y} = 1, (g_{x,y+1} = 0, \dots, g_{x,y+d0} = 0)\},\\
H_{90} &= \{g_{x-d90,y} \mid g_{x,y} = 1, (g_{x-1,y} = 0, \dots, g_{x-d90,y} = 0)\},\\
H_{180} &= \{g_{x,y-d180} \mid g_{x,y} = 1, (g_{x,y-1} = 0, \dots, g_{x,y-d180} = 0)\},\\
H_{270} &= \{g_{x+d270,y} \mid g_{x,y} = 1, (g_{x+1,y} = 0, \dots, g_{x+d270,y} = 0)\},
\end{aligned} \tag{2}
$$

где d_α — измеренное расстояние по азимуту α.

Без трассировки измерительного луча за счет уменьшения числа проверок гипотез формулы упростятся следующим образом:

$$
\begin{aligned}
H_0 &= \{g_{x,y+d0} \mid g_{x,y} = 1, g_{x,y+d0} = 0\},\\
H_{90} &= \{g_{x-d90,y} \mid g_{x,y} = 1, g_{x-d90,y} = 0\},\\
H_{180} &= \{g_{x,y-d180} \mid g_{x,y} = 1, g_{x,y-d180} = 0\},\\
H_{270} &= \{g_{x+d270,y} \mid g_{x,y} = 1, g_{x+d270,y} = 0\}.
\end{aligned} \tag{3}
$$

Однако, отсутствие проверок наличия препятствий по ходу лучей приведет к увеличению числа предполагаемых позиций во множествах гипотез, в свою очередь, оказывая влияние на время вычисления их пересечения. Важно отметить, что при упрощенном вычислении гипотез могут возникать ситуации, когда расчет позиции ведется «сквозь» препятствия, в результате чего возможно возникновение неверных итоговых позиций.

P — множество полученных позиций как результат пересечения сформированных ранее гипотез:

$$P = H_0 \cap H_{90} \cap H_{180} \cap H_{270}. \tag{4}$$

Экспериментальное исследование

Для тестирования рассматриваемого метода применена карта, представленная на рисунке 1. Размер одной ячейки считается равным 10 см. Используемые значения:

d0 = 40 см,
d90 = 50 см,
d180 = 40 см,
d270 = 40 см.

Результат по методу (формула 2): P = {(13, 4), (22, 4)} (на рис. 1 результирующие ячейки выделены оранжевым цветом и имеют значение 5, синим — препятствия (значение 1), красным — неисследованные участки (значение -1)). Результирующие позиции по второму методу (формула 3) показаны на рисунке 2.

При использовании упрощенного метода видно, что в результате присутствует третья позиция с координатами (31; 4) (рис. 2). Ее наличие объясняется отсутствием проверки пути луча, из-за чего не было учтено препятствие (33;4), что показано на рисунках 3 и 4 (значением -3 и зеленым цветов выделены содержимые в множестве H90 ячейки), где второй метод не учитывает препятствие с координатами (33;3), (33;4) и (33;5).

y \ x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36
0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	1	1	1	1	1	1	1	-1	1	1	1	1	1	1	1	1	-1	1	1	1	1	1	1	1	1	-1
1	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
2	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
3	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	1
4	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	0	0	0	5	0	0	0	0	1	0	0	0	5	0	0	0	0	1	0	0	0	0	0	1	0	0	1
5	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	1
6	-1	-1	-1	-1	-1	1	1	-1	-1	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
7	-1	-1	-1	1	1	0	0	1	-1	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
8	-1	1	1	0	0	0	0	0	1	-1	1	1	1	1	0	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	0	1	1	1	1
9	1	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	1	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	1	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	0	1	1	1	1	1	1	1	0	1	1	1	1	1
12	1	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
13	1	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
14	1	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
15	1	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
16	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
17	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
18	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
19	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
20	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
21	-1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	-1	-1	1	1	1	1	1	1	1	-1	1	1	1	1	1	1	1	1	-1

Рисунок 1 – Карта помещения с разрешением 10 см (метод 1)

y \ x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36
0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	1	1	1	1	1	1	1	-1	1	1	1	1	1	1	1	1	-1	1	1	1	1	1	1	1	1	-1
1	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
2	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
3	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	1
4	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	0	0	0	5	0	0	0	0	1	0	0	0	5	0	0	0	0	1	0	0	0	5	0	1	0	0	1
5	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	1
6	-1	-1	-1	-1	-1	1	1	-1	-1	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
7	-1	-1	-1	1	1	0	0	1	-1	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
8	-1	1	1	0	0	0	0	0	1	-1	1	1	1	1	0	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	0	1	1	1	1
9	1	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	1	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	1	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	0	1	1	1	1	1	1	1	0	1	1	1	1	1
12	1	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
13	1	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
14	1	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
15	1	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
16	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
17	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
18	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
19	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
20	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
21	-1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	-1	-1	1	1	1	1	1	1	1	-1	1	1	1	1	1	1	1	1	-1

Рисунок 2 – Карта помещения с разрешением 10 см (метод 2)

y \ x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36
0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	1	1	1	1	1	1	1	-1	1	1	1	1	1	1	1	1	-1	1	1	1	1	1	1	1	1	-1
1	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	0	0	0	-3	0	0	0	0	1	0	0	0	-3	0	0	0	0	1	0	0	0	-3	0	0	0	0	1
2	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	0	0	0	-3	0	0	0	0	1	0	0	0	-3	0	0	0	0	1	0	0	0	-3	0	0	0	0	1
3	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	0	0	0	-3	0	0	0	0	1	0	0	0	-3	0	0	0	0	1	-3	0	0	0	0	1	0	0	1
4	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	0	0	0	-3	0	0	0	0	1	0	0	0	-3	0	0	0	0	1	-3	0	0	0	0	1	0	0	1
5	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	0	0	0	-3	0	0	0	0	1	0	0	0	-3	0	0	0	0	1	-3	0	0	0	0	1	0	0	1
6	-1	-1	-1	-1	-1	1	1	-1	-1	1	0	0	0	-3	0	0	0	0	1	0	0	0	-3	0	0	0	0	1	0	0	0	-3	0	0	0	0	1
7	-1	-1	-1	1	1	0	0	1	-1	1	0	0	0	-3	0	0	0	0	1	0	0	0	-3	0	0	0	0	1	0	0	0	-3	0	0	0	0	1
8	-1	1	1	-3	0	0	0	0	1	-1	1	1	1	1	0	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	0	1	1	1	1
9	1	0	0	-3	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	1	0	0	-3	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	1	0	0	-3	0	0	0	0	1	1	0	0	0	-3	0	0	0	0	1	1	1	1	1	0	1	1	1	1	1	1	1	0	1	1	1	1	1
12	1	0	0	-3	0	0	0	0	1	1	0	0	0	-3	0	0	0	0	1	1	0	0	-3	0	0	0	0	1	0	0	0	-3	0	0	0	0	1
13	1	0	0	-3	0	0	0	0	1	1	0	0	0	-3	0	0	0	0	1	1	0	0	-3	0	0	0	0	1	0	0	0	-3	0	0	0	0	1
14	1	0	0	-3	0	0	0	0	1	1	0	0	0	-3	0	0	0	0	1	1	0	0	-3	0	0	0	0	1	0	0	0	-3	0	0	0	0	1
15	1	0	0	-3	0	0	0	0	1	1	0	0	0	-3	0	0	0	0	1	1	0	0	-3	0	0	0	0	1	0	0	0	-3	0	0	0	0	1
16	1	0	0	0	0	0	0	0	0	0	0	0	0	-3	0	0	0	0	1	1	0	0	-3	0	0	0	0	1	0	0	0	-3	0	0	0	0	1
17	1	0	0	0	0	0	0	0	0	0	0	0	0	-3	0	0	0	0	1	1	0	0	-3	0	0	0	0	1	0	0	0	-3	0	0	0	0	1
18	1	0	0	0	0	0	0	0	0	0	0	0	0	-3	0	0	0	0	1	1	0	0	-3	0	0	0	0	1	0	0	0	-3	0	0	0	0	1
19	1	0	0	0	0	0	0	0	0	0	0	0	0	-3	0	0	0	0	1	1	0	0	-3	0	0	0	0	1	0	0	0	-3	0	0	0	0	1
20	1	0	0	0	0	0	0	0	0	0	0	0	0	-3	0	0	0	0	1	1	0	0	-3	0	0	0	0	1	0	0	0	-3	0	0	0	0	1
21	-1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	-1	-1	1	1	1	1	1	1	1	-1	1	1	1	1	1	1	1	1	-1

Рисунок 3 – Карта помещения с разрешением 10 см (метод 1)

y \ x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36
0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	1	1	1	1	1	1	1	-1	1	1	1	1	1	1	1	1	-1	1	1	1	1	1	1	1	1	-1
1	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	0	0	0	-3	0	0	0	0	1	0	0	0	-3	0	0	0	0	1	0	0	0	-3	0	0	0	0	1
2	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	0	0	0	-3	0	0	0	0	1	0	0	0	-3	0	0	0	0	1	0	0	0	-3	0	0	0	0	1
3	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	0	0	0	-3	0	0	0	0	1	0	0	0	-3	0	0	0	0	1	-3	0	0	-3	0	1	0	0	1
4	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	0	0	0	-3	0	0	0	0	1	0	0	0	-3	0	0	0	0	1	-3	0	0	-3	0	1	0	0	1
5	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	0	0	0	-3	0	0	0	0	1	0	0	0	-3	0	0	0	0	1	-3	0	0	-3	0	1	0	0	1
6	-1	-1	-1	-1	-1	1	1	-1	-1	1	0	0	0	-3	0	0	0	0	1	0	0	0	-3	0	0	0	0	1	0	0	0	-3	0	0	0	0	1
7	-1	-1	-1	1	1	0	0	1	-1	1	0	0	0	-3	0	0	0	0	1	0	0	0	-3	0	0	0	0	1	0	0	0	-3	0	0	0	0	1
8	-1	1	1	-3	0	-3	-3	-3	1	-1	1	1	1	1	-3	1	1	1	1	1	1	1	1	-3	1	1	1	1	1	1	1	1	0	1	1	1	1
9	1	0	0	-3	-3	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	1	0	0	-3	-3	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	1	0	0	-3	-3	0	0	0	1	1	0	0	0	-3	-3	-3	-3	-3	1	1	1	1	1	-3	1	1	1	1	1	1	1	-3	1	1	1	1	1
12	1	0	0	-3	-3	0	0	0	1	1	0	0	0	-3	-3	0	0	0	1	1	0	0	-3	0	0	0	0	1	0	0	0	-3	0	0	0	0	1
13	1	0	0	-3	-3	0	0	0	1	1	0	0	0	-3	-3	0	0	0	1	1	0	0	-3	0	0	0	0	1	0	0	0	-3	0	0	0	0	1
14	1	0	0	-3	-3	0	0	0	1	1	0	0	0	-3	-3	0	0	0	1	1	0	0	-3	0	0	0	0	1	0	0	0	-3	0	0	0	0	1
15	1	0	0	-3	-3	0	0	0	1	1	0	0	0	-3	-3	0	0	0	1	1	0	0	-3	0	0	0	0	1	0	0	0	-3	0	0	0	0	1
16	1	0	0	0	0	0	0	0	0	0	0	0	0	-3	-3	0	0	0	1	1	0	0	-3	0	0	0	0	1	0	0	0	-3	0	0	0	0	1
17	1	0	0	0	0	0	0	0	0	0	0	0	0	-3	-3	0	0	0	1	1	0	0	-3	0	0	0	0	1	0	0	0	-3	0	0	0	0	1
18	1	0	0	0	0	0	0	0	0	0	0	0	0	-3	-3	0	0	0	1	1	0	0	-3	0	0	0	0	1	0	0	0	-3	0	0	0	0	1
19	1	0	0	0	0	0	0	0	0	0	0	0	0	-3	-3	0	0	0	1	1	0	0	-3	0	0	0	0	1	0	0	0	-3	0	0	0	0	1
20	1	0	0	0	0	0	0	0	0	0	0	0	0	-3	-3	0	0	0	1	1	0	0	-3	0	0	0	0	1	0	0	0	-3	0	0	0	0	1
21	-1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	-1	-1	1	1	1	1	1	1	1	-1	1	1	1	1	1	1	1	1	-1

Рисунок 4 – Карта помещения с разрешением 10 см (метод 2)

На рисунке 5 показан результат работы программы с картой с разрешением 10 см, где время 1 — среднее время расчета гипотез, время 2 — среднее время формирования пересечения гипотез, время 3 — среднее итоговое время выполнения метода как сумма времен 1 и 2. Из рисунка 5 следует, что метод 2 привел к уменьшению времени определения гипотез H, однако из-за расчета пересечения P с большим числом гипотез H общее время выполнения второго метода выше, чем первого.

Для определения скорости выполнения методов использовался таймер с точностью 100 нс. Рассматриваемые методы выполнялись по 10000 раз, после чего определялось среднее время выполнения двух этапов методов и общее время выполнения этапов как сумма их средних значений. Время работы методов при их первом вызове не учитывалось.

В таблице 1 и на рисунке 6 показаны результаты тестирований этой же карты с разрешениями, равными 10, 5, 1, и 0,5 см. Следует отметить, что за счет удаления краевых строк и столбцов с неисследованными ячейками (значение -1) и установкой неисследованными ячеек, недоступных для сканирования изнутри помещения (предположим, внутренних ячеек стен), карта, например, с разрешением 5 см не будет иметь размер, в четыре раза больший, чем карта с разрешением 10 см.

Из результатов следует, что первый метод в случаях рассматриваемых карт с разрешениями 10, 5 и 2 см (814, 3024, 18462 ячеек соответственно) выполнил работу быстрее, чем второй. Однако второй метод при разрешениях карт 1 и 0,5 см (72992 и 289842 ячеек) завершил выполнение быстрее первого.

```
-----------------------------------------
|         Результаты работы методов      |
|----------------------------------------|
|         |-----------Значение-----------|
|-Парам.-|------------------------------|
|         |----Метод 1----|----Метод 2----|
|----------------------------------------|
| h0      | 53       поз. | 79       поз. |
| h90     | 57       поз. | 87       поз. |
| h180    | 53       поз. | 75       поз. |
| h270    | 59       поз. | 83       поз. |
| P       | 2        поз. | 3        поз. |
| Время 1| 4140     нс.  | 2978     нс.  |
| Время 2| 7131     нс.  | 11051    нс.  |
| Время 3| 11271    нс.  | 14029    нс.  |
|----------------------------------------|
Итоговые позиции метода 1: 13;4 22;4.
Итоговые позиции метода 2: 13;4 22;4 31;4.
```

Рисунок 5 – Результат работы программы с картой с разрешением 10 см

Таблица 1 – Время выполнения методов при разном разрешении карты, в наносекундах

Разрешение карты, см	Количество ячеек, элем.	Метод 1			Метод 2		
		Время 1	Время 2	Время 3	Время 1	Время 2	Время 3
10	814	4140	7131	11271	2978	11051	14029
5	3024	13141	13442	26583	9582	29816	39398
2	18462	65411	36089	101500	37940	83614	121554
1	72992	236010	70460	306470	124234	159787	284021
0,5	289842	903649	139280	1042929	430207	338482	768689

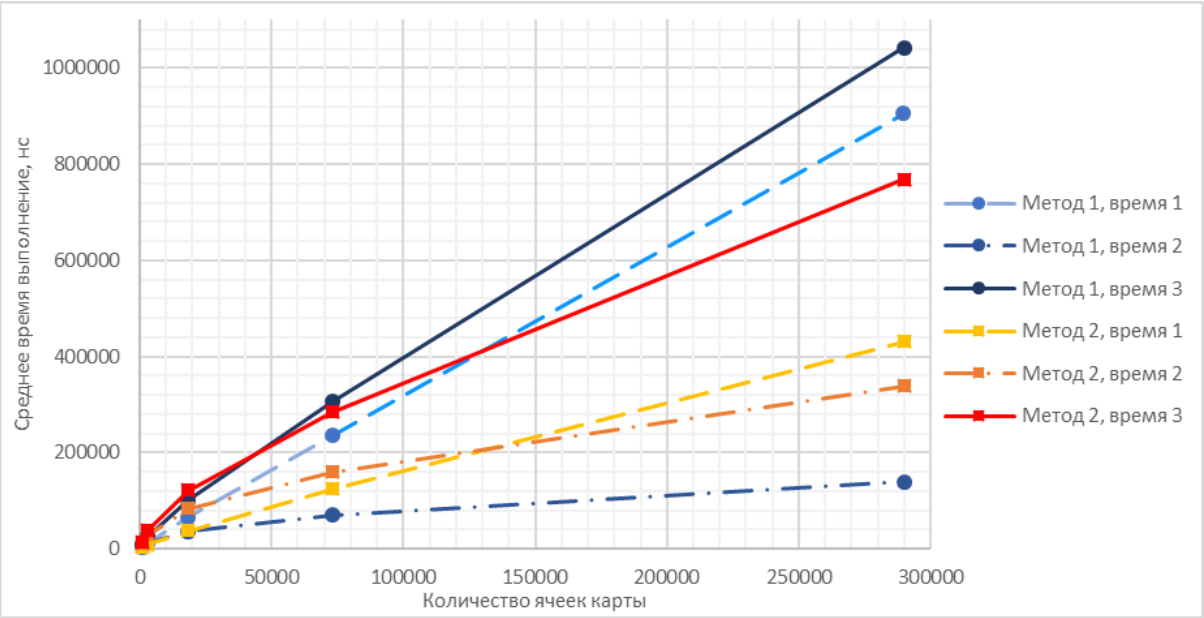


Рисунок 6 – График зависимости скорости выполнения методов от количества ячеек рассматриваемой карты

Выводы

Полученные методы позволяют определить возможные координаты устройства по известной карте. В данном варианте результат определения местоположения при наличии одинаковых помещений или участков с точки зрения их состояния может быть неоднозначен.

Результаты тестирования показали, что второй метод оказался медленнее первого при малом разрешении карты и быстрее при большом. Модификация упрощенного метода дополнительной проверкой для каждой гипотезы, возможно, в данном случае будет способна приблизить его время выполнения к методу с полным отслеживанием хода измерительного луча при малом разрешении карты и сделать быстрее упрощенного метода при большом разрешении.

Рассматриваемый способ локализации не подразумевает необходимости вычислений и хранения карты в памяти робота, следовательно при наличии возможности передачи результатов измерений сенсоров на сервер возможно выполнение расчётов с его помощью, благодаря чему не требуется обладание большими объемом памяти и вычислительными мощностями устройством.

В дальнейших исследованиях для уточнения позиции устройства может применяется его перемещение с ведением истории маршрута и последующим повторным определением возможных позиций. Также в будущем необходимо учитывать параметры датчиков, в частности неточность их результатов измерений.

Литература

1. Мифы народов мира: Энциклопедия / Гл. ред. С. А. Токарев. – М., 2008. – Электронное издание. – Текст : электронный // Internet Archive : [сайт]. – URL: https://archive.org/details/Myths_of_the_Peoples_of_the_World_Encyclopedia_Electronic_publication_Tokarev_and_others_2008/page/n689/mode/2up

2. Сычев, И. Автоматоны: 200-летние роботы / И. Сычев. – Текст: электронный // Хабр : [сайт]. – 2016. – URL: https://habr.com/ru/articles/399133/

3. Юревич, Е. Основы робототехники : учебное пособие / Е. И. Юревич – Текст: электронный // ЭБ СПбПУ. – 2-е изд. – URL: http://elib.spbstu.ru/dl/325.pdf

4. Kuipers, B. Shakey: From Conception to History / B. Kuipers, E. A. Feigenbaum, P. E. Hart, N. J. Nilsson. – Текст : электронный // AI Magazine. – 2017. – № 38. – С. 88–103. – URL: https://onlinelibrary.wiley.com/doi/10.1609/aimag.v38i1.2716

5. Титенок, А. В. Основы робототехники : учебное пособие / А. В. Титенок. — Москва, Вологда : Инфра-Инженерия, 2022. – 236 с. — ISBN 978-5-9729-0872-1. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/124173.html. — Режим доступа: для авторизир. пользователей

6. TRI Teaching Robots to Help People in their Homes // Toyota Research Institute : [site] – URL: https://www.tri.global/news/tri-teaching-robots-help-people-their-homes

7. ГОСТ Р 60.6.8.1-2023. Роботы и робототехнические устройства. Представление

картографических данных для навигации роботов : национальный стандарт Российской Федерации : издание официальное : утвержден и введен в действие Приказом Федерального агентства по техническому регулированию и метрологии от 31 августа 2023 г. № 781-ст : введен впервые : дата введения 2024-01-01 / подготовлен Федеральным государственным автономным научным учреждением «Центральный научно-исследовательский и опытно-конструкторский институт робототехники и технической кибернетики» (ЦНИИ РТК) – Текст: электронный // База ГОСТов : [сайт]. – URL: https://allgosts.ru/25/040/gost_r_60.6.8.1-2023

8. Бобровский, С. Навигация мобильных роботов / С. Бобровский – Текст: электронный // Виртуальный компьютерный музей : [сайт]. – 2016. – URL: https://computer-museum.ru/frgnhist/robonav.htm

9. Радиоэлектронные системы: Основы построения и теория : справочное издание / Я. Д. Ширман, С. Т. Багдасарян, А. С. Маляренко [и др.] – 2-е изд., перераб. и доп. – Москва: «Радиотехника», 2007. – 512 с. – Текст : непосредственный.

10. Игнатов, А. «Глаза» беспилотных автомобилей: LiDAR и компьютерное зрение / А. Игнатов ; FirstVDS – Текст: электронный // Хабр : [сайт]. – 2023. – URL: https://habr.com/ru/companies/first/articles/728224/

11. Мир глазами автомобиля. Каким его видят беспилотники? / Toshiba – Текст: электронный // Хабр : [сайт]. – 2018. – URL: https://habr.com/ru/companies/toshibarus/articles/431388/

Завадская Т. В., Креков О. И. Определение позиции робота по сформированной двумерной карте помещения. *Работа посвящена задаче локализации объекта по заранее известной двумерной карте помещения. Авторами рассматривается предлагаемый метод определения местоположения робота, не требующий хранения карты в памяти устройства, что позволяет организацию вычислений на сервере. Метод математически описан в двух его вариантах: с отслеживанием хода измерительного луча и без него. Приводятся результаты моделирования определения предполагаемых позиций объекта по задаваемым измеренным значениям, а также время выполнения методов в зависимости от изменения разрешения карты.*

Ключевые слова: *дискретная метрическая карта, локализация, определение местоположения, робот.*

Zavadskaya T. V., Krekov O. I. Determination of a robot's position on a preformed two-dimensional indoor map. *This paper is devoted to the problem of robot localization on a known two-dimensional map of an indoor environment. The authors consider the proposed method for determining the robot's location that does not require storing the map in the device memory, which allows organizing the computations on a server. The method is mathematically described in two variants: with and without measuring beam tracing. The results of modeling the determination of the estimated positions of the object according to the given measured values are given, as well as the execution time of the methods depending on the change of the map resolution.*

Keywords: *grid map, localization, position determination, robot.*

Статья поступила в редакцию 02.12.2023
Рекомендована к публикации профессором Мальчевой Р. В.

УДК 004.9

Поиск изображений в графических базах данных по их содержимому

Д. Ф. Ходарев, А. В. Боднар
Донецкий национальный технический университет, г. Донецк
кафедра программной инженерии им. Л.П. Фельдмана
E-mail: vip.khodarev@gmail.com

Аннотация
В статье рассмотрены важность и проблематика поиска изображения по их содержимому. Проведен анализ подходов, включая алгоритмы сопоставления ключевых точек, графовые методы и использование сверточных нейронных сетей. Выполнено сравнение основных известных принципов, а также достоинства и недостатки различных подходов поиска изображений. Представлены результаты кластеризации дубликатов, которые применяются для группировки похожих или идентичных элементов в наборе. Выполнен анализ применения кластеризации дубликатов в различных областях.

Введение

Исследования в области поиска изображений по содержимому стали чрезвычайно актуальны в последние десятилетия в связи с увеличением емкости накопителей информации, распространением цифровой фотографии, а также стремительного роста популярности сети интернет. В текущей ситуации, когда ежедневно в сети публикуются десятки, если не сотни миллионов изображений, становится всё сложнее найти то, что нужно. Раньше использовались различные текстовые метаданные, например, теги описания или ключевые слова. Этот способ потерял эффективность с ростом объема в базах данных из-за невозможности точного отслеживания присвоения им этих метаданных. С появлением более сложных алгоритмов обработки изображений и развитием методов глубокого обучения, системы поиска стали более точными и способны обрабатывать сложные запросы.

Существует множество разных реализаций, например, цветовые гистограммы [1], поиск по ключевым точкам (SIFT, SURF, ORB), использование векторных представлений (Embeddings) и др [2]. Каждый из них нашел своё применение в поиске, обработке и анализе изображений. В завершение следует отметить, что поиск изображений по их содержимому представляет собой активно развивающуюся область исследований в области компьютерного зрения. Современные подходы, такие как использование сверточных нейронных сетей и алгоритмов сопоставления ключевых точек, открывают новые перспективы для повышения эффективности и точности поиска визуальных данных.

Поиск похожих изображений (CBIR)

Поиск похожих изображений, также известный как Content-Based Image Retrieval (CBIR), представляет собой метод, основанный на анализе содержания изображений для поиска и извлечения визуально схожих объектов или сцен. Этот подход учитывает не только метаданные или ключевые слова, но и собственные визуальные характеристики изображений. Процесс поиска можно наблюдать на рис.1.

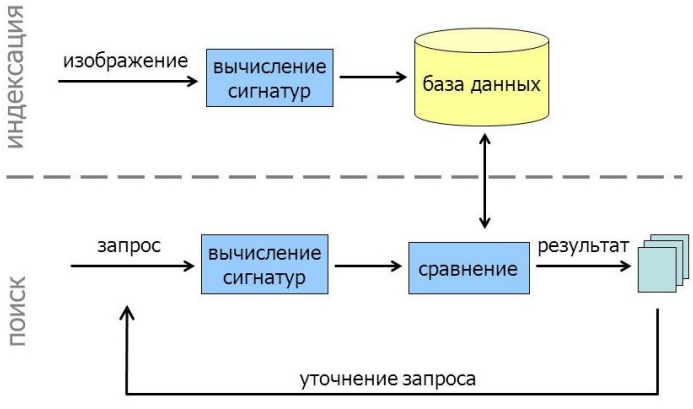


Рисунок 1 – Схема работы CBIR

На рисунке чётко видна последовательность занесения в БД новых данных, а также собственно поиск и сравнение с ней [3].

К основным компонентам этого метода, часто используемым вместе, относятся [4]:

- Извлечение признаков (Feature Extraction): Процесс извлечения важных визуальных характеристик изображений, которые могут включать цвета, текстуры, формы, и другие параметры. Эти признаки помогают создать уникальное представление для каждого изображения.
- Индексация (Indexing): Создание эффективной структуры данных для хранения и организации извлеченных признаков. Индексация ускоряет процесс поиска и обеспечивает более быстрый доступ к хранимым в базе данных изображениям.
- Метрики сходства (Similarity Metrics): Определение методов измерения сходства между изображениями на основе их извлеченных признаков. Эти метрики позволяют оценить, насколько два изображения похожи друг на друга.
- Методы поиска (Search Methods): Разработка алгоритмов для эффективного поиска изображений в базе данных на основе запроса пользователя. Эти методы могут включать в себя индексированный поиск, использование метрик сходства и оптимизированные алгоритмы поиска.

К преимуществам метода можно отнести:

- основан на визуальных характеристиках, что обеспечивает более точные результаты, особенно при многозначном индексировании;
- имеет возможность распознавания объектов, что выделяет его среди других для поиска визуально связанных данных;
- широкий спектр применения, включая медицинскую диагностику, организацию поиска изображений в графических базах данных, в безопасности и видеонаблюдении, а также дизайне и моде.

Несмотря на преимущества, есть и недостатки. Рассмотрим основные.

1) Субъективность восприятия. CBIR основан на визуальных характеристиках, которые могут варьироваться в зависимости от субъективного восприятия разных людей. Таким образом, различия в оценке цветов, текстур или форм могут сказаться на его эффективности [5].

2) Проблемы масштабирования. Некоторые методы извлечения признаков могут быть чувствительны к масштабированию изображений. Это означает, что изображения одного объекта с разными масштабами могут восприниматься как различные, что усложняет поиск.

3) Недостаточный учет разнообразия контента. Некоторые визуальные характеристики, например, форма объектов или текстуры, не всегда полностью передают смысл содержания изображения. В результате, CBIR может не учитывать некоторые аспекты контента, которые важны для пользователя.

4) Сложности в работе с большими базами данных. С увеличением размера базы данных возрастает сложность поиска и сопоставления изображений, требуя более эффективных методов индексации и поиска для поддержания производительности.

5) Неучет семантического контекста. CBIR часто игнорирует семантический контекст, связанный с изображением. Например, для изображения льва CBIR может выделить цвета и форму, но не обязательно понять, что это изображение льва.

6) Чувствительность к изменениям в освещении и угле обзора. Извлечение признаков, таких как цвет, может быть чувствительным к изменениям в освещении, что может затруднить сопоставление изображений при различных условиях освещения.

Поиск по ключевым точкам

Методы поиска по ключевым точкам, такие как SIFT, SURF и ORB, являются популярными в области компьютерного зрения [6]. В этих алгоритмах, точки выбираются так, чтобы они были устойчивы к изменениям масштаба, вращениям и изменениям в яркости.

Каждая ключевая точка описывается локальной окрестностью, называемой дескриптором. Дескрипторы содержат информацию о цвете, текстуре, и других визуальных характеристиках вокруг ключевой точки. Описания ключевых точек индексируются и хранятся в базе данных. Для эффективного поиска создаётся структура данных, позволяющая быстро находить сопоставления.

Когда поступает запрос, система извлекает ключевые точки и их дескрипторы, сравнивает их с данными, хранящимися в базе, и определяет наилучшие сопоставления. Процедура может включать в себя использование метрик сходства, таких как евклидово расстояние или косинусное сходство.

На следующем шаге происходит ранжирование этих сопоставлений, и пользователь получает на выходе набор изображений, наиболее близких к запросу.

Вкратце рассмотрим упомянутые методы и определим их преимущества и недостатки.

SIFT (Scale-Invariant Feature Transform)

Разработан Дэвидом Лоу в 1999 году, SIFT является одним из первых и наиболее

широко используемых методов. Он выделяет ключевые точки, устойчивые к изменениям масштаба и вращения.

Преимущества:

- инвариантность к масштабированию и вращению;
- робастность к изменениям в освещении и частичным сокрытиям.

Недостатки:

- требователен к вычислительным ресурсам;
- большой объем хранимых данных, что может сказаться на производительности.

***SURF (Speeded-Up Robust Features*)**

Разработан в 2006 году Гербертом Бэйем и другими. SURF был создан для улучшения производительности по сравнению с SIFT, сохраняя его преимущества. Он использует быстрый алгоритм определения местоположения ключевых точек.

Преимущества:

- более быстрое вычисление по сравнению с SIFT;
- эффективен для реального времени.

Недостаток – менее устойчив к некоторым изменениям в изображении по сравнению с SIFT.

ORB (Oriented FAST and Rotated BRIEF)

Разработан в 2011 году Эдвардом Рублингом и др. ORB является более новым методом, созданным для обеспечения быстрого и эффективного поиска по ключевым точкам, поддерживая высокую производительность.

Преимущества:

- очень быстрый и эффективный;
- меньшее потребление памяти.

Недостаток - **м**ожет быть менее точным в некоторых случаях по сравнению с SIFT и SURF.

Эти методы широко используются в задачах компьютерного зрения, включая поиск похожих изображений, реконструкцию 3D-сцен, распознавание объектов и другие приложения. Применяются в областях, таких как медицинская диагностика, робототехника, андроиды, виртуальная реальность и многое другое. Выбор метода зависит от требований конкретной задачи, производительности и ресурсов, доступных для вычислений.

Свёрточные нейронные сети

Сверточные нейронные сети (СНС) представляют собой мощный класс алгоритмов машинного обучения, широко применяемых в обработке изображений и анализе визуальных данных [7]. Эти сети спроектированы для автоматического извлечения иерархических признаков из входных изображений.

Основная идея СНС заключается в использовании сверточных слоев для локального анализа изображений. Свертки – это фильтры, которые применяются к небольшим областям входного изображения, что позволяет сети выделять локальные шаблоны и признаки. Постепенно увеличивается абстрактность признаков на более высоких уровнях иерархии.

Пулинговые слои применяются для уменьшения размерности данных, сохраняя важные признаки. Это способствует локализации и обобщению выделенных черт. После нескольких слоев свертки и пулинга, данные передаются полносвязным слоям для классификации или других задач обработки информации.

Сверточные нейронные сети обладают уникальной способностью к автоматическому обучению пространственных и иерархических представлений изображений. Это делает их особенно эффективными для решения задач распознавания образов, классификации изображений, обнаружения объектов и даже создания искусственных генеративных моделей. Их применение не ограничивается только областью компьютерного зрения, и сверточные нейронные сети успешно применяются в различных областях, включая медицинскую диагностику, автономные транспортные средства и анализ естественного языка.

Использование сверточных нейронных сетей (СНС) предоставляет ряд значительных преимуществ. Среди плюсов стоит выделить способность СНС автоматически извлекать иерархические признаки из входных данных, что делает их эффективными в задачах обработки изображений. СНС также демонстрируют хорошую способность к обучению и обобщению, что делает их подходящими для различных задач машинного обучения [7].

Однако, использование СНС сопряжено с некоторыми недостатками. Требовательность к вычислительным ресурсам и объему данных может создавать проблемы в реализации на устройствах с ограниченными возможностями. Также, интерпретация и визуализация внутренней работы СНС могут быть сложными, что усложняет понимание принимаемых ими решений. Кроме того, не всегда легко подобрать оптимальную архитектуру и настройки для конкретной задачи, что может потребовать значительных усилий в процессе настройки модели.

Проблема дубликатов

Среди задач поиска изображений на основе их содержимого есть проблема в наличии дубликатов. Допустим, мы хотим найти какую-то вещь, но из-за чрезмерной популярности данного изображения, мы получим лишь сотни или тысячи дубликатов,

один из которых отображен на рис. 2. Для решения данной проблемв предложена кластеризация дубликатов [8].

Рисунок 2 – Тумбнейлерные дубликаты

Процесс кластеризации дубликатов часто применяется для группировки похожих или идентичных элементов в наборе (рис.3). Это полезно в случаях, когда при работе с базой данных изображений или текстовыми документами требуется выявить, объединить или удалить эти самые дубликаты.

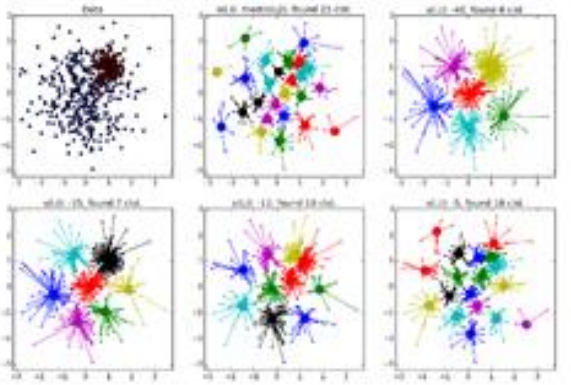


Рисунок 3 – Наглядный пример кластеризации

Рассмотрим несколько методов и подходов реализации этого процесса.

Хэш-функции.

Использование хеш-функций для быстрого обнаружения дубликатов. Если два элемента имеют одинаковый хеш, они могут считаться кандидатами в дубликаты, и затем дополнительные шаги могут быть предприняты для подтверждения этого.

Сравнение по признакам

Реализация сравнения по признакам для выявления сходства между элементами может включать в себя сравнение векторных представлений (например, векторов признаков изображений или текста) и применение метрик сходства.

Методы машинного обучения

Применяются для обучения модели, которая может классифицировать, являются ли пары элементов дубликатами. Может включать в себя использование пар признаков и классификацию на основе близости.

Блочная обработка

Разделение данных на блоки и сравнение элементов только внутри блоков, что может ускорить процесс и сделать его более масштабируемым.

Сверточные нейронные сети

Использование сверточных нейронных сетей для извлечения признаков и определения сходства между изображениями. Этот метод часто используется для кластеризации дубликатов изображений.

Методы на основе графов

Представление данных в виде графа и использование алгоритмов кластеризации для выявления подграфов, представляющих собой дубликаты.

В зависимости от ситуации, типа данных, точности, объема, скорости и вычислительных способностей, могут применяться различные методы. Поэтому выбор метода кластеризации дубликатов зависит от требований конкретной задачи.

Применение кластеризации дубликатов в различных областях

Кластеризация в различных задачах имеет ряд проблем, с которым сталкиваются разработчики и что влияет на эффективность и скорость работы. Среди основных проблем можно выделить следующие:

- выбор подходящего метода;
- определение числа кластеров (неправильно подобранное число может негативно повлиять на результаты);
- сложность алгоритмов (некоторые алгоритмы требуют высокие вычислительные ресурсы);
- чувствительность к выбросам;
- интерпретация результатов;
- необходимость в предварительной обработке данных и др.

В независимости от сложности, данные методы высоко востребованы в различных областях. Среди множества направлений применений можно выделить несколько наиболее важных в данные время.

Медицинская диагностика

В [9] дано обсуждение использования кластеризации дубликатов в медицинской области для обработки и анализа медицинских изображений, включающее в себя выявление дубликатов снимков, что особенно важно для точной диагностики и мониторинга состояния пациентов.

Промышленность и обработка изображений

Исследование использования методов кластеризации для обработки изображений в промышленности. Это может включать в себя поиск и управление дубликатами в графических

базах данных, применение в качестве инструмента для качественного контроля, анализа производственных процессов и т.д.

Архивация и управление данными

Рассмотрение применения кластеризации дубликатов в области архивации и управления данными. Может включать в себя оптимизацию хранения изображений, выявление и удаление дубликатов при архивировании больших объемов данных.

Интернет-платформы и социальные сети

Исследование роли кластеризации дубликатов в сфере социальных сетей и интернет-платформ. Здесь кластеризация может применяться для улучшения качества поиска изображений, а также для обнаружения и управления дубликатами, что особенно актуально в условиях постоянного потока новых данных.

Культурное наследие и цифровые архивы

Рассмотрение применения в сфере сохранения культурного наследия и создания цифровых архивов. Кластеризация дубликатов может помочь в эффективной организации и поддержке цифровых коллекций, облегчая поиск и управление контентом [10].

Образование и научные исследования

Обзор использования кластеризации дубликатов в образовательных и научных целях. Это может включать в себя работу с графическими базами данных для создания обучающих материалов, анализа результатов исследований и обеспечения эффективной обработки изображений в научных проектах.

Безопасность и анализ изображений

Рассмотрение применения кластеризации дубликатов в сфере безопасности и анализа изображений. Это может включать в себя выявление схожих изображений для обеспечения безопасности, а также их кластеризацию для выявления закономерностей и трендов.

Заключение

Выполненное исследование позволило выявить множество аспектов и вызовов в области обработки и анализа графической информации. Методы поиска похожих изображений (CBIR) и по ключевым точкам предоставляют эффективные инструменты для поиска и анализа визуальных данных. Однако, существуют проблемы, такие как сложность выбора оптимального метода, чувствительность к различным типам данных, вопросы интерпретации результатов, а также проблемы, связанные с дубликатами.

Анализ проблем дубликатов выделяет важность кластеризации для обработки и группировки похожих или идентичных элементов в графических базах данных. Этот процесс включает в себя использование различных методов, таких как хэш-функции, сравнение по признакам, методы машинного обучения и сверточные нейронные сети. Кластеризация дубликатов играет ключевую роль в решении проблемы избыточности и обеспечивает эффективное управление данными в графических базах данных.

Одновременно с этим, необходимость решения проблем выбора метода кластеризации, обработки разнородных данных и поддержания сбалансированности весов признаков предоставляет перспективы для дальнейших исследований в области поиска изображений на основе их содержимого. Тенденции в развитии включают в себя интеграцию с технологиями глубокого обучения, семантическое понимание изображений и применение в различных сферах, от медицинской диагностики до обработки изображений в промышленности.

Постоянное внимание к проблемам и направлениям исследований в этой области позволит эффективно решать вызовы и создавать инновационные методы для поиска изображений на основе их содержимого.

Литература

1. Полный перечень методов CBIR [Интернет-ресурс]. - URL: https://en.wikipedia.org/wiki/List_of_CBIR_engines

2. Поиск изображений на основе содержания habr [Интернет-ресурс]. - URL: https://habr.com/ru/articles/103385/

3. Васильева, Н. Поиск изображений. Синтез различных методов поиска при формировании результатов / Н. Васильева, А. Дольник, И. Марков // Интернет-математика 2007: сб. работ участников конкурса науч. проектов по информ. Поиску. — Екатеринбург : Изд-во Урал. ун-та, 2007. — С. 46–55.

4. Karami, E. Image Matching Using SIFT, SURF, BRIEF and ORB: Performance Comparison for Distorted Images [Electronic resource] / Ebrahim Karami, Siva Prasad, Mohamed Shehata. - URL: https://arxiv.org/abs/1710.02726

5. Десятников, И. Е. Поиск изображений по визуальному содержанию в графических базах данных и сети Интернет [Электронный ресурс] / И. Е. Десятников // Информационные технологии и вычислительные системы, 2013. - № 2. – С. 88-95. – URL: https://www.isa.ru/jitcs/images/documents/2013-02/88_95.pdf

6. Шозда, Н. С. Поиск изображений в больших БД с использование коэффициента корреляции цветовых гистограмм [Электронный ресурс] / Н. С. Шозда, Е. А. Башков // Графикон, 2002. – URL: http://www.graphicon.ru/2002/s89.

7. Романов, А. А. Свёрточные нейронные сети [Электронный ресурс] / А. А. Романов // Научные исследования, 2018. - № 1(21). – URL: https://scientificresearch.ru/index.php/blizhajshij-nomer/01-00-00-fiziko-matematicheskie-nauki/266-svertochnye

8. Ouhda, M. Using Image Segmentation in Content Based Image Retrieval Method [Электронный ресурс] / Mohamed Ouhda, Khalid El Asnaoui, Mohammed Ouanan, Brahim Aksasse // Advanced Information Technology, Services and Systems (AIT2S 2017). Conference Paper. - P. 179-195. – URL: https://www.researchgate.net/publication/321019137_Using_Image_Segmentation_in_Content_Based_Image_Retrieval_Method

9. Левчук, В. А. Компьютеризированная диагностика меланомы на базе поиска похожих дерматоскопических изображений в базе данных / В. А. Левчук, В. А. Ковалев, В. В. Баркалин, В. Э. Лозовский // Весці Нацыянальнай акадэміі навук Беларусі, 2016. – № 2. – С. 86-91.

10. Тюрин, А. Г. Кластерный анализ, мето-ды и алгоритмы кластеризации / А. Г. Тюрин, И. О. Зуев // Вестник МГТУ МИРЭА. - М.: Изд-во МГТУ, 2014. - № 3.

Ходарев Д. Ф., Боднар А. В. Поиск изображений в графических базах данных на основе их содержимого. *В статье рассматривается важность и проблематика поиска изображения по их содержимому. Проведен анализ подходов, включая алгоритмы сопоставления ключевых точек, графовые методы и использование сверточных нейронных сетей. Проведено сравнение основных известных принципов, а также плюсы и минусы различных подходов поиска изображений. Представлены результаты кластеризации дубликатов, которые применяются для группировки похожих или идентичных элементов в наборе. Выполнен анализ применения кластеризации дубликатов в различных областях.*

Ключевые слова: *поиск изображений, CBIR, ключевые точки, проблема дубликатов, кластеризация, методы CBIR*

Khodarev D. F., Bodnar A. V. Search for images in graphic databases using their contents. *The article discusses the importance and problems of image search by their content. The analysis of approaches, including algorithms for matching key points, graph methods and the use of convolutional neural networks, is carried out. The main known principles are compared, as well as the pros and cons of various image search approaches. The results of clustering duplicates are presented, which are used to group similar or identical elements in a set. The analysis of the application of duplicate clustering in various fields is performed.*

Keywords: *image search, CBIR, key points, duplicate problem, clustering, CBIR methods*

Статья поступила в редакцию 05.12.2023
Рекомендована к публикации профессором Мальчевой Р. В.

УДК 519.64:622.248.54

Программный комплекс моделирования динамических процессов работы бурильной колонны

Т. В. Кучер
Донецкий институт ГПС МЧС России, г. Донецк
kucher_t@mail.ru

Аннотация
В работе исследуется динамика продольных колебательных процессов при работе бурильной колонны. Для решения дифференциального уравнения в частных производных использован метод конечных разностей, составлена расчетная разностная схема. Разработан программный комплекс, позволяющий выполнять расчеты при заданных знаниях исходных параметров, а также при изменяющемся одном или двух исходных параметрах. По результатам расчетов построены графические зависимости.

Введение

Один из самых распространенных, сложных и трудоемких типов аварии в разведочном бурении – это прихваты, представляющие собой потерю подвижности колонны труб из-за разных причин. На их долю приходится до 60-80% аварийного времени [1].

Существует несколько эффективных методов ликвидации прихватов [1, 2, 3]. В практике разведочного бурения для ликвидации прихватов колонковых наборов обычно применяются ударные механизмы. Такие механизмы действуют по принципу реализации энергии упругой деформации твердого телах [4].

Т.к. размеры ударных механизмов на порядок меньше длины колонны бурильных труб, то в этом случае возможно рассматривать их, соответственно, как пружину и упругий стержень с равномерно распределенной массой. Кроме того, в момент соударения бойка с наковальней устройства допускается рассматривать прихваченный снаряд в состоянии покоя, поскольку, за время от момента размыкания замка ударного механизма до удара, колебания в нем практически затухают.

Колонна является упругим элементом большой протяженности и при определенных условиях работы возникают различного рода колебания. Динамика продольных колебательных процессов при работе бурильной колонны описывается гиперболическим дифференциальным уравнением в частных производных. Особенностью решения данной задачи является то, что рабочий цикл ударного механизма разбивается на несколько этапов, различающихся граничными условиями [5]:

• первая фаза – сжатие или растягивание упругого элемента, предполагается наличие «затвора», что обеспечивает фиксацию бойка при накоплении потенциальной энергии;

• вторая фаза – этот этап рабочего цикла предполагает освобождение бойка путем размыкания «затвора»;

• третья фаза – после размыкания «затвора» осуществляется перемещение бойка, который, достигая наковальни, связанного с прихваченной частью снаряда, передает ей накопленный запас энергии.

Особенностью этой задачи является то, что состояние системы в конце одного рабочего цикла определяет начальные условия для следующего этапа. В этом случае задачу целесообразно решать численными методами с разработкой соответствующего программного обеспечения.

Постановка задачи

При анализе рабочего цикла ударного механизма использовалась расчетная схема, приведенная на рисунке 1. Рассматривается модель, состоящая из бурильной колонны 1, представляющей из себя упругий стержень длиной L, колонна подвешена с помощью талевого каната 6. Талевая система состоит из талевого блока 2, буровой лебедки 7, струн талевой системы 6, представляющих собой упругие нити. Масса талевой системы учитывается в виде сосредоточенной массы M_1, связанной с верхней частью упругого стержня.

На нижнем торце бурильной колонны закреплены включаемые в состав снаряда 5 утяжеленные бурильные трубы 3 (УБТ) массой M_2 (возможно их отсутствие).

К нижней части колонны сверх веса колонны приложена растягивающая нагрузка P.

Поскольку эффективность применения ударного механизма определяется не только величиной ударной силы, но и характером возбуждаемых в колонне волновых процессов, то рабочий цикл механизма исследуется на всем его протяжении до нарушения контакта между бойком и корпусом в момент прихода

отраженной волны растяжения к нижнему торцу колонны бурильных труб.

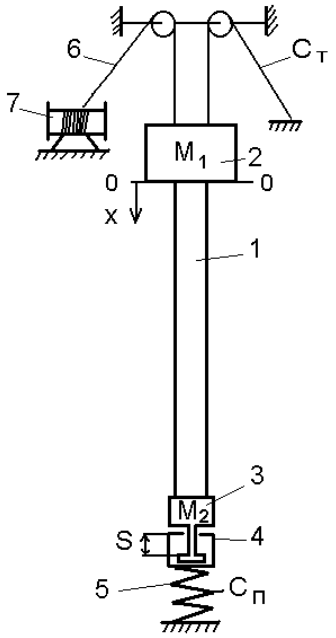


Рисунок 1 - Расчетная схема для анализа рабочего цикла ударного механизма

Цикл работы ударного механизма описывается волновым уравнением продольных колебаний упругого стержня:

$$u_{tt} - c^2 u_{xx} = 0,$$

где x – текущая координата по оси колонны, м, x=0…L;

L – длина бурильной колонны, L=500…1000 м;

t – текущее значение времени, с, $t = 0...T,$ где $T = \frac{2L}{c}$ – общее время рабочего цикла;

u – перемещение сечения колонны с координатой x, м;

c – скорость распространения волны упругой деформации в материале труб, м/с.

Рабочий цикл ударного механизма разделяется на три этапа, отличающимися друг от друга граничными и начальными условиями.

1.Фаза разгона бойка, при которой накопленная в бурильной колонне энергия упругой деформации переходит в кинетическую энергию движения. Этап начинается в момент размыкания замка ударного механизма и заканчивается в момент контакта бойка с корпусом устройства. При этом верхний конец колонны бурильных труб рассматривается как упруго закрепленный (действует сила упругости со стороны талевой системы), а нижний торец – свободный.

Обозначим $u^I(x,t)$ - перемещение сечения на первом этапе, T_1 – время окончания первой фазы работы. Тогда условие окончания расчета по первому этапу (t=T_1) выглядит

$$u(L,T_1) < U^I_{пред}, \quad U^I_{пред} = U_{нач} - U_{хода},$$

где $U_{нач}$ – начальное растяжение нижнего сечения колонны под воздействием силы P,

$U_{хода}$ - перемещение нижнего сечения колонны, зависящее от конструкционных особенностей механизма.

Начальные условия для первого этапа (t_1=0):

- перемещение сечения колонны с координатой x равно упругой деформации участка бурильных труб от 0 до x:

$$u^I(x,0) = \frac{Px}{EF}$$

- сечение колонны с координатой x неподвижно:

$$u^I_t(x,0) = 0.$$

Граничные условия для первого этапа ($t_1 \in [0;T_1]$) – упругая сила в верхнем сечении (x=0) и в нижнем торце (x=l) бурильной колонны:

$$EFu^I_x(0,t_1) = zu^I(0,t_1) + M_1 u^I_{tt}(0,t_1) + P$$
$$EFu^I_x(l,t_1) = -M_2 u^I_{tt}(l,t_1),$$

где EF – жесткость колонны; z – коэффициент жесткости талевой системы.

2.Фаза удара, при которой боек взаимодействует с корпусом ударного механизма (нижний конец бурильной колонны становится упруго закреплённым), при этом на верхний конец бурильных труб ещё продолжает действовать упругая сила со стороны талевой системы. Обозначим $u^{II}(x,t)$ – перемещение сечения на втором этапе Этап заканчивается в момент времени T_2, когда $u^{II}(0,T_2) = -u_c$, где u_c – статическое положение равновесия талевой системы под действием веса бурильной колонны. При этом в конце фазы верхний торец колонны разгружается от действия силы упругости талевой системы и далее рассматривается как свободный.

Начальными условиями второй фазы будут значения, полученные при расчете первого цикла работы – перемещение и скорость сечения колонны с координатой x $u^I(x,T_1)$ и $u^I_t(x,T_1)$:

$$u^{II}(x,0) = u^I(x,T_1), \quad u^{II}_t(x,0) = u^I_t(x,T_1).$$

Граничные условия для второго этапа (для $t_2 \in [0;T_2]$) – упругая сила в верхнем сечении (x=0) и нижнем торце (x=l) бурильной колонны:

$$EFu^{II}_x(0,t_2) = zu^{II}(0,t_2) + M_1 u^{II}_{tt}(0,t_2) + P$$
$$EFu^{II}_x(l,t_2) = -M_2 u^{II}_{tt}(l,t_2) - G\left[u^{II}(l,t_2) - u^I(l,T_1)\right].$$

3.Фаза удара, при которой контакт бойка с корпусом продолжается, т.е. нижний конец бурильной колонны остаётся упруго закреплённым, а верхний конец бурильных труб свободен. Этап заканчивается в момент прихода отраженной волны растяжения к контактному сечению бойка с наковальней ударного механизма при $T_3 = 2 \cdot l / c - T_1 - T_2$.

Следует отметить, что при значениях растягивающей силы, не превышающих вес колонны, вторая фаза длится до окончания рабочего цикла.

При рассмотрении третьего этапа учитывается, что его начальными условиями ($t_2=0$) являются значения $u^{II}(x,T_2)$ и $u_t^{II}(x,T_2)$ – перемещение и скорость сечения колонны с координатой x

$$u^{III}(x,0) = u^{II}(x,T_2), \quad u_t^{III}(x,0) = u_t^{II}(x,T_2).$$

Граничные условия для третьего этапа имеют вид (для $t_3 \in [0;T_3]$):

- упругая сила в верхнем сечении ($x=0$) бурильной колонны равна 0:
$$u_x^{III}(0,t_3) = 0;$$
- упругая сила в нижнем торце бурильной колонны ($x=l$):
$$EFu_x^{II}(l,t_2) = -M_2 u_{tt}^{II}(l,t_2) - G\left[u^{II}(l,t_2) - u^{I}(l,T_1)\right].$$

Зная перемещение нижнего сечения бурильной колонны в течение второй и третьей фаз работы ударного механизма, можно определить величину усилия P_y, действующего на прихваченный буровой снаряд:

$$P_y = G[u^{II-III}(l,t_{2-3}) - u_0]$$

Для решения задачи моделирования динамических процессов в бурильной колонне использован универсальный метод конечных разностей (метод сеток) [5]. Частные производные в уравнении, начальных и граничных условиях заменены разделёнными разностями:

$$U_{tt} = \frac{U_i^{j-1} - 2 \cdot U_i^j + U_i^{j+1}}{\Delta t^2}, \quad U_{xx} = \frac{U_{i-1}^j - 2 \cdot U_i^j + U_{i+1}^j}{h^2},$$

$$U_t = \frac{U_i^j - U_i^{j-1}}{\Delta t}, \quad U_x = \frac{U_{i-1}^j - U_i^j}{h},$$

где h и Δt - шаг сетки по x и t соответственно,

U_i^j - текущий узел сетки, соответствующий значению функции $U(x,t)$.

Шаг по времени выбирается из условия устойчивости явной разностной схемы $\Delta t \le h / c$ [6].

После замены расчетная формула во внутренних узлах сетки на всех трех этапах имеет вид

$$U_i^{j+1} = 2 \cdot U_i^j - U_i^{j-1} + \frac{c^2 \cdot \Delta t^2}{h^2} \cdot \left(U_{i-1}^j + U_{i+1}^j - 2 \cdot U_i^j\right).$$

Аналогичным образом были получены формулы для расчета в пограничных слоях для трех этапов [6].

Результаты работы

В качестве средства разработки программного комплекса решения поставленной задачи была выбрана свободная кроссплатформенная среда визуальной разработки профессионального уровня Lazarus [7, 8, 9].

Стартовое окно разработанной программы показано на рисунке 2, предлагает несколько режимов дальнейшей работы:

- ознакомление с методикой расчета;
- расчет по длине колоны и по времени – в этом случае при заданных характеристиках талевой системы, бурильной колонны и ударного механизма результатами расчета являются величины усилия P_y, действующего на прихваченный буровой снаряд, скорость снаряда, продольные перемещения сечений по длине колонны в зависимости от времени; пример расчета показан на рисунке 3;
- расчет комплекса характеристик физического процесса с одним или двумя изменяющимися параметрами – зависимость перемещений верхнего и нижнего сечений на концах колонны, зависимости скорости снаряда и усилия, действующего на снаряд, от времени.

На рисунках 4 и 5 приведены результаты расчета параметров при одном изменяющемся параметре – коэффициенте для задания значения хода бойка, который влияет на начальное растяжение нижнего сечения колонны под воздействием силы P, и соответственно, на время окончания первой фазы работы, или массы снаряда бурильной колонны. При необходимости можно просмотреть числовые данные, по которым были построены графики. На рисунке 6 приведен результат расчета при двух изменяющихся параметрах, предлагается выбрать два параметра из трех, третьим параметров добавлена длина колонны.

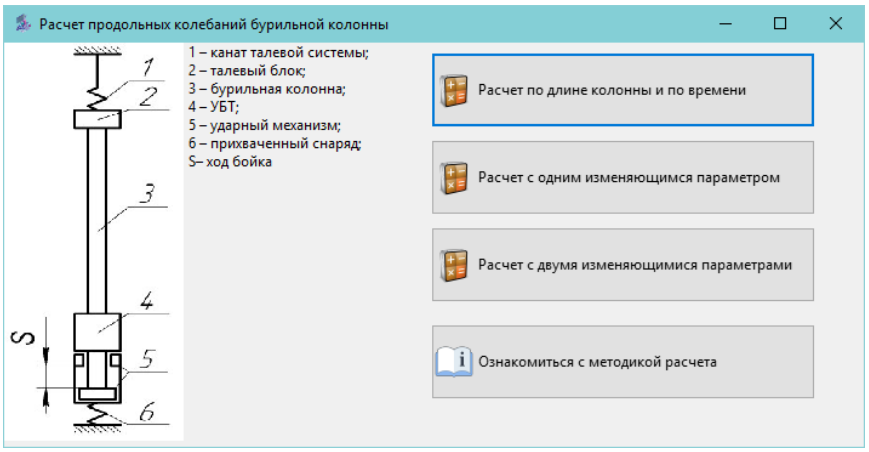


Рисунок 2 – Стартовое окно программы

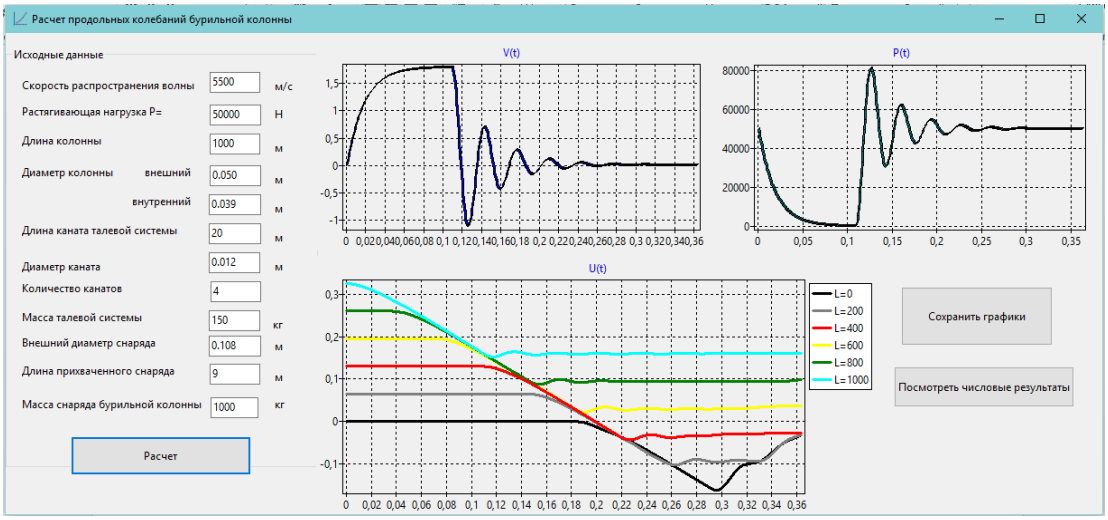


Рисунок 3 –Пример расчета при заданных параметрах

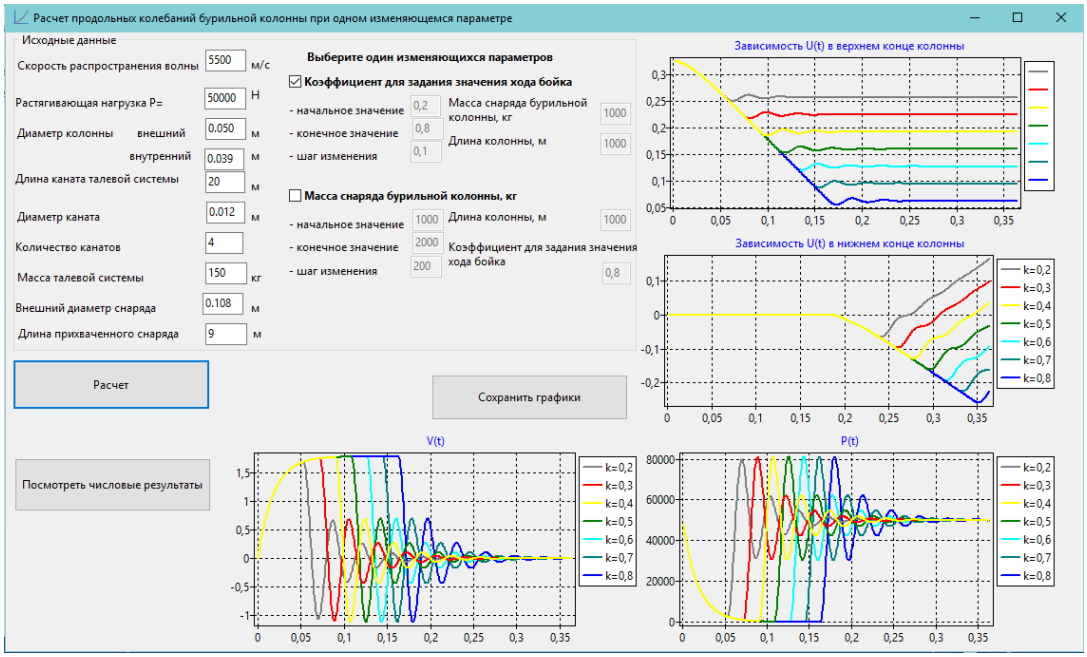


Рисунок 4 – Пример расчета при изменяющемся значении коэффициента хода бойка

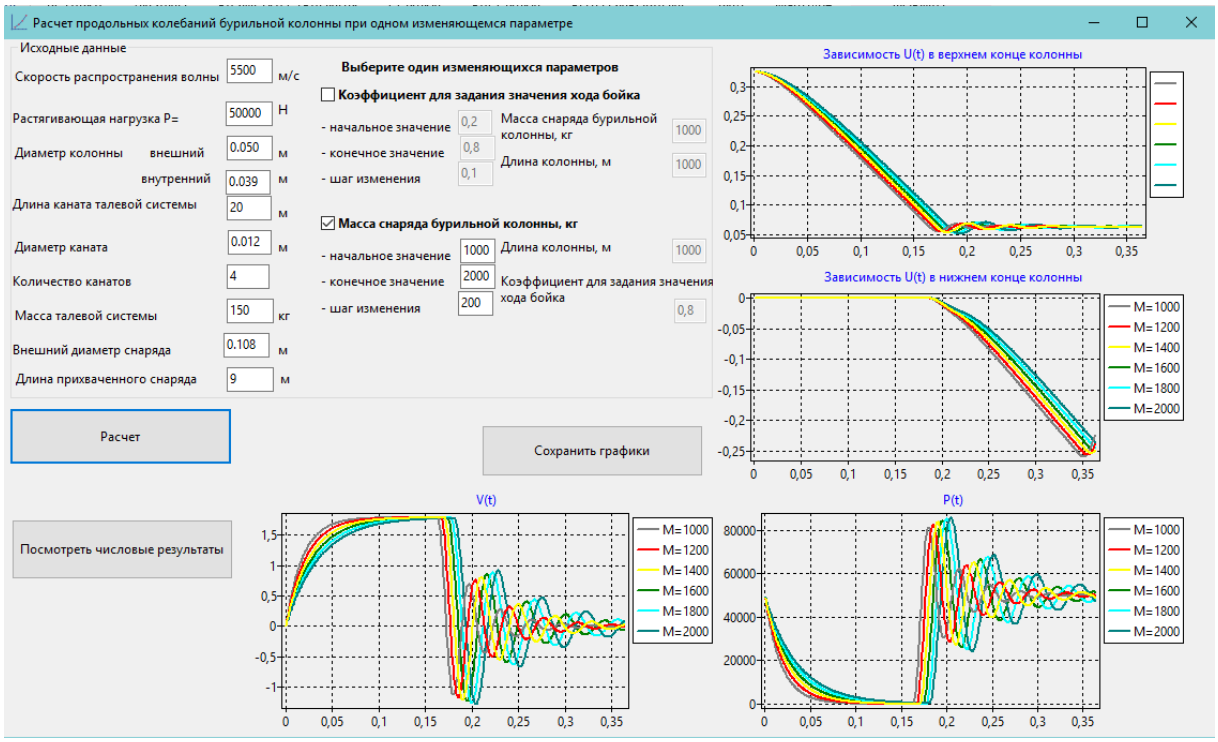
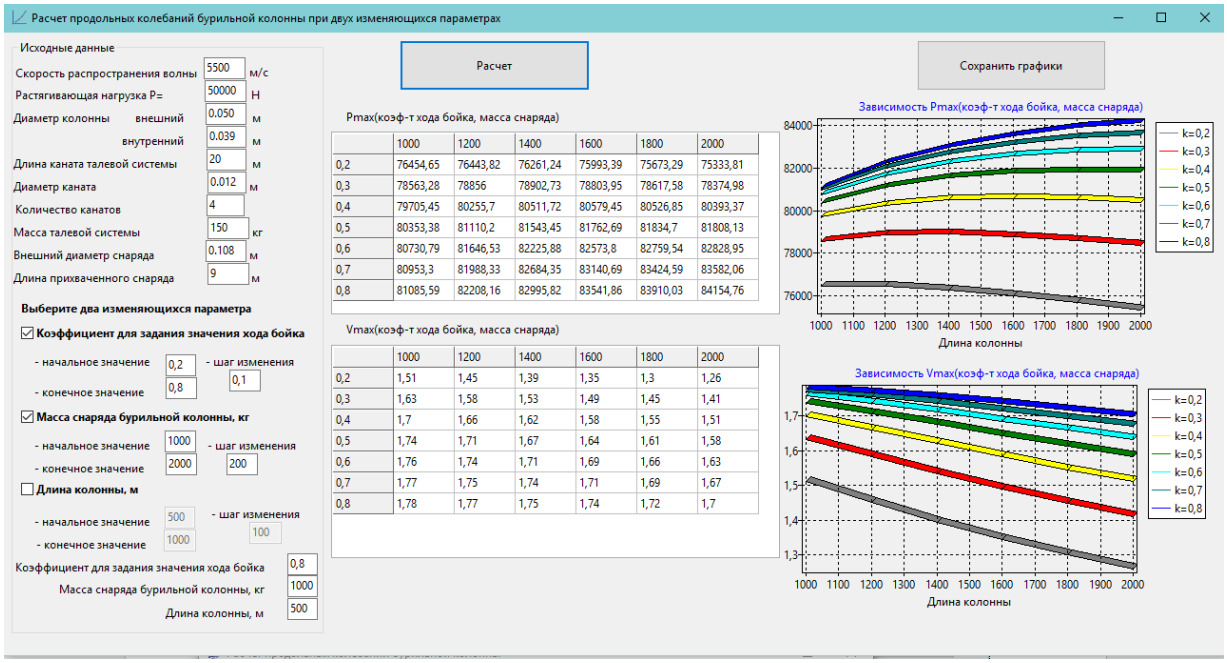


Рисунок 5 - Пример расчета при изменяющемся значении массы снаряда бурильной колонны



	1000	1200	1400	1600	1800	2000
0,2	76454,65	76443,82	76261,24	75993,39	75673,29	75333,81
0,3	78563,28	78856	78902,73	78803,95	78617,58	78374,98
0,4	79705,45	80255,7	80511,72	80579,45	80526,85	80393,37
0,5	80353,38	81110,2	81543,45	81762,69	81834,7	81808,13
0,6	80730,79	81646,53	82225,88	82573,8	82759,54	82828,95
0,7	80953,3	81988,33	82684,35	83140,69	83424,59	83582,06
0,8	81085,59	82208,16	82995,82	83541,86	83910,03	84154,76

	1000	1200	1400	1600	1800	2000
0,2	1,51	1,45	1,39	1,35	1,3	1,26
0,3	1,63	1,58	1,53	1,49	1,45	1,41
0,4	1,7	1,66	1,62	1,58	1,55	1,51
0,5	1,74	1,71	1,67	1,64	1,61	1,58
0,6	1,76	1,74	1,71	1,69	1,66	1,63
0,7	1,77	1,75	1,74	1,71	1,69	1,67
0,8	1,78	1,77	1,75	1,74	1,72	1,7

Рисунок 6 - Пример расчета при двух изменяющихся параметрах

Выводы

В ходе работы была исследована динамика продольных колебательных процессов при работе бурильной колонны.

Для решения дифференциального уравнения в частных производных была составлена разностная схема. Разработанный программный комплекс может работать не только с операционными системами семейства Windows, но и с Unix-подобными системами.

По сравнению с результатами, полученными ранее [6], в работе добавлены результаты расчета с изменяющимися параметрами, что позволяет увидеть и проанализировать динамику колебательного

процесса не только по времени, но и при разных значения массы снаряда бурильной колонны, длины колонны и коэффициента для задания значения хода бойка.

Литература

1. Коломоец, А. В. Предупреждение и ликвидация прихватов в разведочном бурении / А. В. Коломоец. – М.: Недра, 1985. – 220 с.

2. Юртаев, В. Г. Динамика буровых установок / В. Г. Юртаев. - М . : Недра, 1987. - 160 с.

3. Пустовойтенко, И. П. Предупреждение и методы ликвидации аварий и осложениний в бурении: Учебное пособие для профтехобразования / И. П. Пустовойтенко. – М.: Недра, 1987. – 237 с.

4. Головань, Е. В. Применение ударных механизмов для предупреждения и ликвидации прихватов бурильной колонны [Электронный ресурс] / Е. В. Головань, К. Д. Угланов, Р. В. Коршакова, А. А. Пестова // Научно-образовательный журнал для студентов и преподавателей «StudNet», 2020. - №11/2020. – Режим доступа: https://stud.net.ru/primenenie-udarnyx-mexanizmov-dlya-preduprezhdeniya-i-likvidacii-prixvatov-burilnoj-kolonny (дата обращения: 12.09.2023).

5. Тихонов, А. Н. Уравнения математической физики (6-е изд.) // А. Н. Тихонов, А. А. Самарский. - М . : Наука, 1999. - 799 с.

6. Каракозов, А. А. Разработка программного обеспечения математической модели работы бурильной колонны при ликвидации прихватов бурового снаряда// А. А. Каракозов, Е. Р. Алексеев, Т. В. Кучер, С. Н. Парфенюк // Наукові праці ДонНТУ. Серія «Гірничо-геологічна». - № 2(19)’2013. - С. 43–51.

7. Lazarus documentation [Электронный ресурс] – Режим доступа: http://wiki.freepascal.org/Lazarus_Documentation (дата обращения: 12.09.2023).

8. Алексеев, Е. Р. Free Pascal и Lazarus: Учебник по программированию / Е. Р. Алексеев, О. В. Чеснокова, Т. В. Кучер — М. : ALT Linux ; Издательский дом ДМК-пресс, 2010. — 440 с. (Библиотека ALT Linux).

9. Lazarus 2.2.0 - Бесплатная среда по разработке программного обеспечения [Электронный ресурс] – Режим доступа: https://lazarus-rus.ru/Documentation (дата обращения: 12.09.2023).

Кучер Т. В. Программный комплекс моделирования динамических процессов работы бурильной колонны. *В работе исследуется динамика продольных колебательных процессов при работе бурильной колонны. Для решения дифференциального уравнения в частных производных использован метод конечных разностей, составлена расчетная разностная схема. Разработан программный комплекс, позволяющий выполнять расчеты при заданных знаниях исходных параметров, а также при изменяющемся одном или двух исходных параметрах. По результатам расчетов построены графические зависимости.*

Ключевые слова: *бурильная колонна, ударный механизм, волновое уравнение, программный комплекс, Lazarus*

Kucher T. V. A software package for simulation of dynamical processes of the drill string working*. The article examines the dynamics of longitudinal oscillatory processes during the operation of a drill string. To solve the partial differential equation, the finite difference method was used, and a calculated difference scheme was compiled. A software package has been developed that allows calculations to be performed at specified values of the initial parameters, as well as with one or two initial parameters changing. Based on the results of calculations, graphical dependencies are constructed.*

Keywords*: drill string, impact mechanism, wave equation, software package, Lazarus*

Статья поступила в редакцию 08.12.2023
Рекомендована к публикации профессором Павлышом В. Н.

УДК 004.931

Системный анализ переменных психоэмоционального состояния человека в системе принятия решений автоматизированных систем управления

О.В. Теплова[*1], О.А. Криводубский[*2]
[*1]Федеральное государственное бюджетное научное учреждение «Институт проблем искусственного интеллекта»
[*2]ФГБОУ ВО «Донецкий национальный технический университет»
olga-anonim@mail.ru, oleg.krivodubski.dn@gmail.com

Аннотация
В данной работе рассматривается: проявления психических состояний операторов автоматизированных систем по функционально-распределенным признакам лица, приведена классификация показателей зон лица, методы фиксации и численного представления ключевых точек, определены особенности персонала в нормальных и аварийных условиях, принципы, предусматривающие формирование содержания баз данных и разработки моделей и алгоритмов. Работа содержит численное значение ключевых точек в предложенной системе координат.

Введение

В работе выделены основные функционально-распределенные признаки психических состояний персонала, работающего на пультах управления производствами: эмоциональные состояния, сосредоточенность, интерес к деятельности. При анализе мимического проявления выделены следующие эмоции: удивление, страх, гнев, спокойствие. Предложена классификация показателей зон лица и метод фиксации и численного представления ключевых точек с помощью наложения на изображения координатных осей. Рассмотрены особенности психоэмоциональной реакции персонала на внештатные ситуации. На основе рассмотренных особенностей сформированы принципы формирования содержания базы данных.

Цель работы – провести системный анализ основных функционально-распределенных признаков лица человека для определения психических состояний персонала, работающего на пультах управления производствами.

Классификация показателей зон лица

Для анализа мимики человека в физиогномике выделяют три зоны лица сверху вниз [1], как показано на рисунке 1: интеллектуальная (И), эмоциональная (Э), витальная (В). Интеллектуальная зона (И) включает в себя следующие элементы: лоб ($И_1$), глаза ($И_{21}$, $И_{22}$) и брови ($И_{31}$, $И_{32}$). Эмоциональная включает в себя нос ($Э_1$), щеки ($Э_{21}$, $Э_{22}$), скулы ($Э_{31}$, $Э_{32}$). Витальная содержит рот ($В_1$) и подбородок ($В_2$).

Каждая зона лица содержит группы мимических мышц. В зоне И находятся мышцы лба, надбровные дуги, мышцы век. Во второй зоне лица находятся пирамидальные мышцы носа, крылья носа, скулы. Витальную зону составляют губы, подбородок. Согласно данному представлению лица, можно описать качественные и количественные показатели мимики, которые соответствуют переживаемым оператором психическим состояниям. В состоянии спокойствия мышцы во всех трех зонах находятся в состоянии общего тонуса организма (норма-тонус), которое фиксируется как исходно представление человеческого лица.

Формализация и численное представление ключевых точек лица

Для формализации и численного представления ключевых точек предусматривается наложение трехмерной сети координатных осей x, y, z на изображения персонала. Путем нормирования показателей пространственных величин, будет получено значение положения мышц лица человека в состоянии обычного функционирования системы и в случаях внештатных ситуаций.

Получение изображения мимики персонала во время штатных и внештатных ситуаций происходит с помощью видеокамеры, контролирующей лица персонала на рабочих местах.

Обозначив положения основных мышц, задействованных в выражении психических состояний, через ключевые точки и перенеся полученную разметку на координатную ось, получим численное обозначение основных

ключевых точек (P_i), по расположению которых относительно координатной оси можно будет диагностировать переживаемую эмоцию. В качестве примера используем двумерное изображение. Выразив мимику через числовые значения координат ключевых точек P_i (рис. 2), получаем результат, представленный в табл. 1-3.

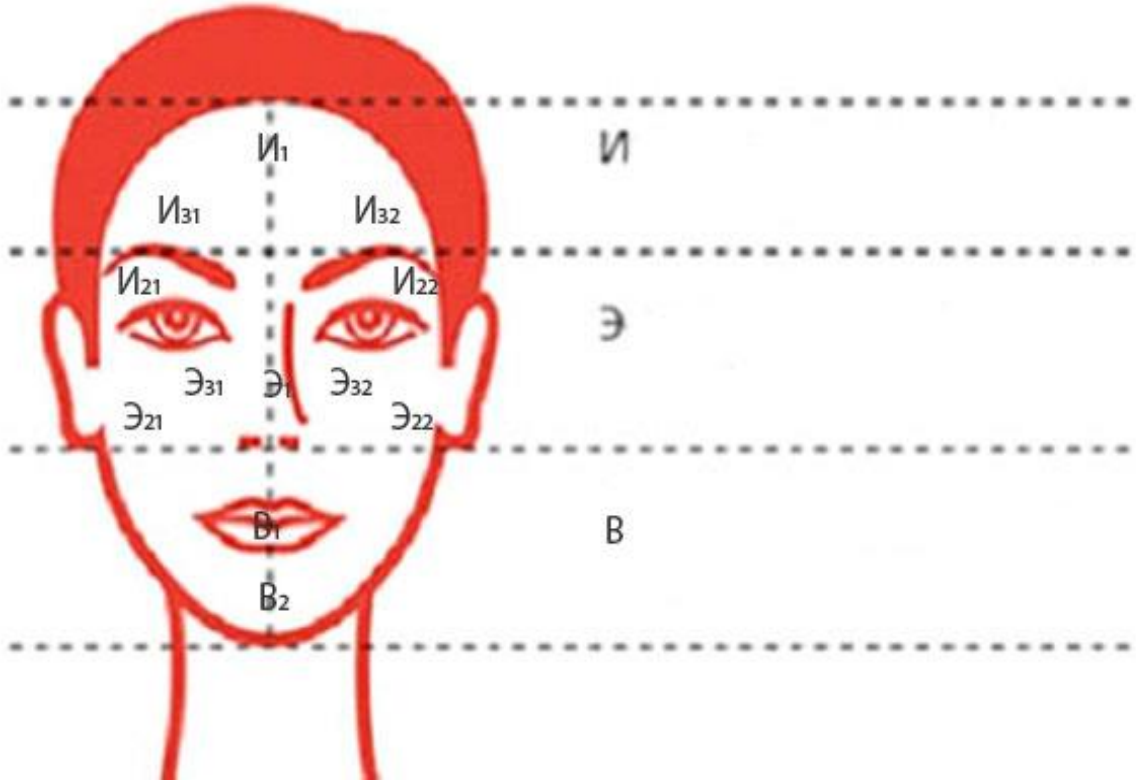


Рисунок 1 – Физиогномическое разделение лица на три зоны

Таблица 1 – Фиксация элементов интеллектуальной зоны лица в системе координат

Психическое состояние	Лоб	Глаза		Брови
		Левый	Правый	
Спокойствие	P_1=(-6,8) P_2=(0,8) P_3=(6,8)	P_4=(-8.5,2) P_5=(-5.5,3) P_6=(-2,1.5) P_7=(-5.5,3)	P_8=(2.25,1.5) P_9=(5.25,3) P_{10}=(8.5,2) P_{11}=(5.5,1)	P_{12}=(-6,6.5) P_{13}=(2,4) P_{14}=(2,4) P_{15}=(5.5,6.5)

Таблица 2 – Фиксация элементов эмоциональной зоны лица в системе координат

Психическое состояние	Нос				Скулы	
	Крылья носа		Пирамидальные мышцы		Левая	Правая
	Левое	Правое	Левая	Правая		
Спокойствие	P_{16}=(-4,-3)	P_{19}=(4,-3)	P_{17}=(-2,-2) P_{18}=(-1,-1)	P_{20}=(2,-2) P_{21}=(1,-1)	P_{22}=(-7,-2)	P_{28}=(7,-2)

Таблица 3 – Фиксация элементов витальной зоны лица в системе координат

Психическое состояние	Рот	Подбородок
Спокойствие	P_{23}=(-5,-8) P_{24}=(0,-8) P_{25}=(4.5,-8) P_{26}=(0,-11)	P_{27}=(0,-14)

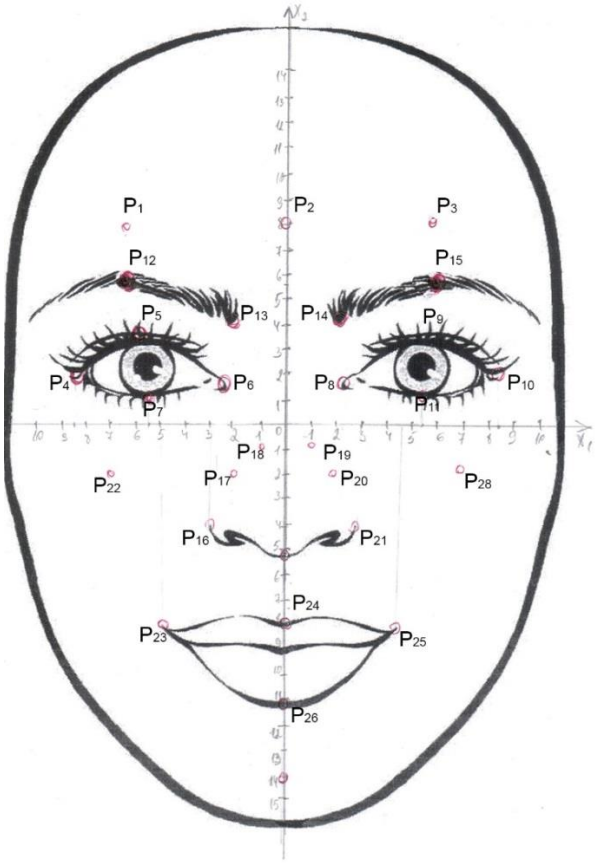


Рисунок 2 – Нанесение координатных осей и ключевых точек на изображение лица человека (тонус-норма)

Методика фиксации точек лица предусматривает группировку точек зонам лица в виде 2D-модели (формулы 1-3).

.(1)

. (2)

. (3)

Тогда формальное представление выражения лица имеет вид (формула 4):

. (4)

Фиксация особенностей персонала в нормальных и аварийных условиях

Согласно формальному представлению, психические состояния персонала на рабочих местах будут влиять на изменение расположения лицевых мышц (табл. 4). Типичными психическими состояниями при работе на производстве являются: радость, сосредоточенность, скука, интерес к работе, спокойствие, печаль, вызванная непроизводственными факторами. Типичные реакции на ситуации, опасные для жизни, следующие: страх, гнев, переходящий в ярость, плач и «изоляция аффекта». Так как внештатная ситуация на производстве, вызванная случайными девиациями в работе систем не всегда представляет собой непосредственную опасность для жизни и здоровья персонала в данной работе рассмотрены мимические изменения состояния тонус-нормы для эмоциональных состояний гнева, страха и удивления (табл. 5-7).

Таблица 4 – Физиогномическое выражения психических состояний на лице человека

Психическое состояние	Интеллектуальная зона лица	Эмоциональная зона лица	Витальная зона лица
Радость	Человек жмурится	Скулы поднимаются	Уголки губ поднимаются, рот слегка приоткрыт
Страх	Брови подняты, между ними образуется морщинка на переносице, верхние веки приподняты, глаза широко распахнуты	Мышцы напряжены	Рот слегка приоткрыт
Удивление	Брови приподняты, изгибаются. Глаза широко раскрыты.		Рот приоткрыт
Печаль	глаза печального человека слегка прикрыты, взгляд отсутствующий, отрешенный, направленный вниз или вглубь себя	Лицо бледнеет, теряет тонус.	Уголки губ опущены, нижняя губа может быть слегка выпячена. Нижняя часть его лица теряет тонус и отвисает
Гнев	Брови нахмурены, над переносицей образуется глубокая складка.	Напряжение в скулах, играющие желваки. Сокращение пирамидальной мышцы носа, на переносье появляются горизонтальные складки	Стиснутые зубы, плотно сжатые губы
Спокойствие	Фиксация взгляда без напряжения	Мышцы расслаблены	Мышцы расслаблены
Скука	Глаза полуприкрыты или взгляд расфокусирован	Мышцы расслаблены	Мышцы расслаблены
Сосредоточенность	Горизонтальные складки на лбу, брови слегка нахмурены, глаза чуть-чуть сужены	Мышцы носа напряжены	Плотно сжатый рот, уголки губ опущены
Интерес	Глаза могут быть расширены, если есть удивление, или сужены, если происходит мыслительная деятельность.		Чуть открытый рот

Таблица 5 – Фиксация элементов интеллектуальной зоны лица в системе координат во время внештатных ситуаций

Эмоция	Лоб	Глаза		Брови
		Правый	Левый	
Страх	P_1=(-8.5,5) P_2=(0,10) P_3=(8.5,5)	P_4=(-5.75,1,5) P_5=(-4,3.75) P_6=(-2,1,5) P_7=(-4,1)	P_8=(2,1.5) P_9=(4,1) P_{10}=(6,1.5) P_{11}=(4,1)	P_{12}=(-5,6) P_{13}=(-2.5,5.5) P_{14}=(2.5,5) P_{15}=(5,6)
Гнев	P_1=(-3.75,10) P_2=(0,10) P_3=(2.75,10)	P_4=(-7,3) P_5=(-4.5,4.5) P_6=(-2.5,2.25) P_7=(-4.75,2.25)	P_8=(2.5,2) P_9=(4.5,4.5) P_{10}=(6.75,3) P_{11}=(4.75,2.25)	P_{12}=(-6,6) P_{13}=(-1.5,3.75) P_{14}=(1.5,3.75) P_{15}=(5.5,6)
Удивление	P_1=(-4,9) P_2=(0,9) P_3=(4,9)	P_4=(-7.25,1.5) P_5=(-5,3.5) P_6=(-2.5,1) P_7=(-5,0.5)	P_8=(1.5,1.5) P_9=(3.5,4) P_{10}=(6.25,1.5) P_{11}=(0.5,0.5)	P_{12}=(-5,6) P_{13}=(-4,5) P_{14}=(3,4) P_{15}=(5.5,6)

Таблица 6 – Фиксация элементов эмоциональной зоны лица в системе координат во время внештатных ситуаций

Психическое состояние	Нос				Скулы	
	Крылья носа		Пирамидальные мышцы		Левая	Правая
	Левое	Правое	Левая	Правая		
Страх	P_{16}=(-2.5,-5)	P_{19}=(0.5,0)	P_{17}=(-1.5,-2) P_{18}=(-0.5,0)	P_{20}=(1.5,-2) P_{21}=(2,3)	P_{22}=(-5,-1.75)	P_{21}=(5,-1.75)
Гнев	P_{16}=(-2,-5)	P_{19}=(1.5,1)	P_{17}=(-1.75,-2) P_{18}=(-1.5,1)	P_{20}=(1.5,-2) P_{21}=(2,-5)	P_{22}=(-5,-1)	P_{21}=(5,-1)
Удивление	P_{16}=(-3.75,-5)	P_{19}=(1,-1)	P_{17}=(-1.5,-2) P_{18}=(-1,-1)	P_{20}=(1.5,-2) P_{21}=(2,-5)	P_{22}=(-6,-1.5)	P_{21}=(5,-1.5)

Таблица 7– Фиксация элементов витальной зоны лица в системе координат во время внештатных ситуаций

Психоэмоциональное состояние	Рот	Подбородок
Страх	P_{23}=(-4,-10.5) P_{24}=(0,-10) P_{25}=(3.5,-10.5) P_{26}=(0,-13)	P_{27}=(0,-19)
Гнев	P_{23}=(-4,-9) P_{24}=(0,-8) P_{25}=(4,-9) P_{26}=(0,-10)	P_{27}=(0,-15)
Удивление	P_{23}=(-5,-11) P_{24}=(0,-10) P_{25}=(3,-11) P_{26}=(0,-14.5)	P_{27}=(0,-18)

Принципы формирования содержания баз данных

Полученные данные будут использованы для формирования содержания баз данных и разработки моделей и алгоритмов, согласно принципам: нормирование входных данных (нормализация значений Pi), адаптивность к характеристикам персонала (пол, возраст, ношение аксессуаров). Эта задача осуществима при многократной фиксации ключевых точек отдельного человека, входящего в обслуживающий персонал, в обычных условиях работы и при тестовой имитации реакций на внештатные ситуации (страх, гнев, удивление).

Во время опытной эксплуатации системы в случае внештатных ситуаций сравнение реальных численных показателей ключевых точек будет проводиться с численными показателями ключевых точек в состоянии нормального функционирования системы с сохранением текущих показателей. Используя трехмерную модель человеческого лица во время реальных внештатных ситуаций на производстве, будет производиться повторное обучение системы. Только после редактирования данных, согласно показателям, снятым во время реальной внештатной ситуации, будет производиться диагностика психического состояния оператора автоматизированных систем управлений во время внештатных ситуаций.

Исходя из диагностированной психической реакции автоматически будет приниматься решение о выполнении команд операторов или выполнения запротоколированной инструкции действий, согласно возникшей ситуации. В случае адекватного состояния персонала, система принятия решений полностью делегирует человеку принятие решений, без введения на исполнение предписанных инструкций.

Выводы

В работе рассмотрен метод описания мимических проявлений психических состояний персонала, произведена классификация зон лица, предложена формализация представления ключевых точек путем наложения трехмерной сети координатных осей x, y, z на изображения

персонала. В качестве примера формально описаны проявления следующих психоэмоциональных состояний: спокойствие, страх, гнев, удивление. Установлены основные принципы формирования содержания баз данных и дальнейшей разработки моделей и алгоритмов.

Литература

1. Равенский, Н. Как читать человека. Черты лица, жесты, позы, мимика / Н. Равенский. – Москва: РИПОЛ классик, 2007. – ISBN 978-5-7905-5021-8

2. Rodehorst, V. Comparison and evaluation of feature point detectors/ V. Rodehorst, A. Koschan // Proceedings of 5th International Symposium Turkish-German Joint Geodetic Days, 2006.

3. Tuytelaars, T. Local Invariant Feature Detectors / T. Tuytelaars, K. Mikolajczyk // Survey. Foundations and Trends in Computer Graphics and Vision, 2008. – No. 3 (3). – P. 177-280.

4. Жабинский, А.В. Метод распознавания эмоций на основе модели распределения ключевых расстояний / А. В. Жабинский // Доклады БГУИР. - Минск: БГУИР, 2014. - №1(79). - С.59-64. - URL: https://cyberleninka.ru/ article/n/metod-raspoznavaniya-emotsiy-na-osnove-modeli-raspredeleniya-klyuchevyh-rasstoyaniy

5. Бондаренко, В. А. Метод поиска и сопоставления ключевых особенностей изображений для распознавания образов и сопровождения объектов / В. А. Бондаренко, Г. Э. Каплинский, В. А. Павлова, В. А. Тупиков // Известия ЮФУ. Технические науки, Раздел V. Техническое зрение, 2019. - С.281-293. – DOI 10.23683/2311-3103-2019-1-281-293

6. Левкин, В. Е. Психические состояния: учебное пособие для вузов / В. Е. Левкин. - Москва : Юрайт, 2016. – 302 с.

7. Миненко, А. С. Система распознавания эмоционального состояния человека / А. С. Миненко, Т. В. Ванжа / Искусственный интеллект, 2020. – №3. – С. 60-69.

8. Р 50-34.119-90. Рекомендации. Информационная технология. Комплекс стандартов на автоматизированные системы. Архитектура локальных вычислительных сетей в системах промышленной автоматизации. Общие положения.

9. Кутбиддинова, Р. А. Психология стресса : учебно-методическое пособие / Р. А. Кутбиддинова. - Южно-Сахалинск: СахГУ, 2019. – 372 с.

10. Экман, П. Узнай лжеца по выражению лица [Текст] : [16+] / П. Экман, У. Фризен; [пер. с англ. В. Кузин под научной редакцией М. Осориной]. - Санкт-Петербург : Питер, 2018. – 315 с.

Теплова О.В., Криводубский О.А. Системный анализ переменных психоэмоционального состояния человека в системе принятия решений автоматизированных систем управления. *В данной работе рассматривается: проявления психических состояний операторов автоматизированных систем по функционально-распределенным признакам лица, приведена классификация показателей зон лица, методы фиксации и численного представления ключевых точек, определены особенности персонала в нормальных и аварийных условиях, принципы, предусматривающие формирование содержания баз данных и разработки моделей и алгоритмов. Работа содержит численное значение ключевых точек в предложенной системе координат.*

Ключевые слова: *психоэмоциональное состояние, по функционально-распределенные признаки, классификация, особенности персонала, формальное представление.*

Teplova O.V., Krivodubsky O.A. System analysis of human psychoemotional state variables in automated control systems decision-making system. *This article treats: manifestations of automated systems operator's mental states according to functionally distributed facial features, provides a facial zones indicators classification, fixation methods and numerical representation of key points, defines the personnel characteristics in normal and emergency conditions, principles providing for the database content formation and the models and algorithms development. The article contains the numerical value of the key points in the proposed coordinate system.*

Keywords: *psychoemotional state, functionally distributed features, classification, personnel characteristics, formal presentation.*

Статья поступила в редакцию 09.12.2023
Рекомендована к публикации профессором Зори С.А.

УДК 004.021

СРАВНЕНИЕ ПРОИЗВОДИТЕЛЬНОСТИ РАЗЛИЧНЫХ ПОДХОДОВ В ЗАДАЧЕ КЛАССИФИКАЦИИ

А. О. Истягин, О. В. Рычка
Донецкий национальный технический университет
e-mail: ist_75@mail.ru, red_rose_2005@mail.ru

Аннотация
В статье рассмотрены различные подходы в решении задачи классификации, реализована нейронная сеть для классификации типов номеров отелей, проведен анализ производительности работы регулярных выражений и нейронных сетей, построены графики зависимости скорости классификации больших объемов данных от выбранного алгоритма, сделаны выводы о целесообразности и уместности применения различных подходов. Сделан вывод, что подход с применением нейронной сети обрабатывает данные значительно быстрее, однако требует дополнительных ресурсов для внедрения.

Введение

После завершения пандемии и периода самоизоляции вновь актуальным стал отельный бизнес. В мире существуют сотни тысяч различных отелей, гостиниц и курортных домов, в каждом – десятки номеров, комнат и тарифов для проживания. Для привлечения большего количества гостей отели заключают контракты с фирмами-агрегаторами, которые собирают в общую базу данных множество отелей и предлагают их клиентам, в том числе другим агрегаторам (далее: поставщикам), каждый из которых, разумеется, увеличивает исходные цены для извлечения собственной прибыли. Подобная цепочка наценок может проходить через множество фирм, прежде чем дойдет до конечного потребителя. Каждая такая фирма должна правильно систематизировать собственные категории номеров и спальных мест всех своих отелей и отелей других поставщиков, с которыми заключен контракт. Единого правила или алгоритма классификации не существует, и все поставщики делают это по-своему, что значительно усложняет классификацию типов номеров.

Постановка задачи

Целью данного исследования является определение наиболее оптимального механизма определения типов номеров отельного бизнеса.

Рассмотрим функционал подробнее: на вход механизма должна подаваться строка, содержащее название номера, на выходе алгоритм должен выдавать одну из записей в заранее сформированных справочниках (тип номера, тип кровати и подтип номера соответственно, в порядке убывания приоритета) [1, 2].

Для лучшего понимания приведем некоторый пример. Пусть в справочнике «тип номера» существуют следующие типы: «стандарт», «люкс», «эконом» и другие, в справочнике «тип кровати»: «одноместная», «две одноместные», «двухместная», «двухместная king size» и другие, в «подтипе номера»: «апартаменты», «вид на море», «вид на горы», «двухэтажный» и другие. Таким образом, номер «односпальный номер эконом с видом на море» должен быть классифицирован как «эконом», «одноместная» и «вид на море» соответственно. Также должна быть реализована мультиязычность и возможность сокращения слов в названии. В случае невозможности определить один из типов возвращать зарезервированный тип «не определено».

Анализ существующих подходов

В общем случае достаточно простую задачу классификации можно решить, используя конечный набор регулярных выражений: шаблонов ключевых слов, по которым можно определить необходимые типы номеров и кроватей. Множество современных информационных систем реализованы на базе трехзвенной архитектуры: клиентское приложение, слой бизнес-логики и слой доступа к данным с самой базой данных или любыми другими источниками.

В случае задачи классификации алгоритм нельзя размещать на стороне клиента, т.к. клиент по определению не должен ничего рассчитывать. Исходя из этого, алгоритм можно разместить в базе данных или на слое бизнес-логики приложения.

В случае с реляционными базами данных, которыми пользуются многие информационные системы, целесообразно разместить алгоритм

внутри хранимой функции или процедуры (в зависимости от СУБД и способов реализации). Это обеспечит наиболее быстрый доступ к данным.

В случае размещения алгоритма на слое бизнес-логики реализация может быть выполнена в любой форме (встраиваться в существующую систему, быть встраиваемым модулем или сторонним сервисом). Также следует изучить возможность применения нейронных сетей для классификации типов номеров и кроватей в отдельном модуле.

Предложенный механизм регулярных выражений уже реализован у одного из поставщиков, название которого не будет упомянуто в этой работе, как и детали реализации. Рассмотрим принцип работы в общих чертах. В ходе исследования будем придерживаться программно-инструментальных средств поставщика для наиболее корректного сравнения производительности алгоритмов. В данном случае будут рассмотрены база данных SQL [3] Server и язык программирования C# [4]. Т.к. решения с нейронными сетями у поставщика нет, оно будет реализовано в демонстрационном режиме самостоятельно с использованием языка Python и библиотеки PyTorch [5].

Регулярные выражения, выполняемые внутри базы данных SQL Server

Рассмотрим подробнее решение поставщика по классификации комнат через регулярные выражения. Сперва надо создать и наполнить необходимые таблицы реляционной базы данных.

Таблица RoomKeyword представляет собой справочник, содержащий регулярные выражения и их идентификаторы.

Таблица RoomKeywordRoomTypesRefs является таблицей пересечений и ссылается на таблицу RoomKeyword и RoomType, а также содержит числовое значение приоритета каждого регулярного выражения (например, в названии номера «Номер люкс бизнес» приоритетнее часть «бизнес», чем «люкс», т.к. люкс – более общее описание номера, а бизнес – более узкое).

Последняя из базовых таблиц – RoomType, содержащая тип номера и его идентификатор.

На основании этих таблиц можно реализовать простейший механизм классификации (и при необходимости дополнительно его оптимизировать созданием индексов, разделением на подзапросы и т.д.). Т.к. механизм предполагает выбор самого приоритетного совпадения, для множественного поиска реализуем циклический проход по названиям комнат, передаваемых как аргумент хранимой процедуры, с использованием курсора. Результаты разберем в следующих разделах.

Регулярные выражения, выполняемые внутри API на C#

Реализация механизма классификации в коде API[6] на C# позволит внедрять классификатор в любую часть существующей бизнес-логики как встроенный модуль или отдельный сервис. Для классификации комнат сперва необходимо загрузить в память названия комнат и все шаблоны.

Для связи C#-программ с базами данных применим библиотеку Dapper, она позволяет писать SQL-команды в сыром виде в коде, выполнять их и приводить ответ к экземплярам классов, что облегчит работу с данными в объектно-ориентированном языке.

Ожидаемо, что загрузка большого количества данных требует времени. Можно предположить, что это время компенсируется фактом того, что с загруженными в оперативную память работа будет проходить быстрее, чем с построчным считыванием из базы в случае работы внутри БД.

Также важно обратить внимание, что комнаты не классифицируются сразу большими блоками: в каждом отеле их обычно меньше сотни, добавление отелей в систему также не происходит большими группами. Для более корректного сравнения проанализируем время классификации большого количества комнат с учетом времени загрузки и без учета времени загрузки по отдельности.

Нейронная сеть, классифицирующая типы комнат

Так как принцип работы нейронной сети не настолько очевиден, как циклический проход по всем названиям комнат и сравнение каждого с шаблоном, целесообразно подробнее описать разработанный подход. Во многом он строится на применении базовых утилит языка Python и библиотек PyTorch [7], Transformers, SKLearn, в том числе предобученная модель rubert-tiny2.

Для обучения модели и синтеза результата сперва необходимо определить классы входных данных. Класс RoomDataset содержит информацию обо всех входных данных: типы комнат, названия, их количества и др.

Класс BalancedDataset имеет схожие атрибуты, но при инициализации разбивает все названия на группы по типу номера и балансирует их так, чтобы редкие типы номеров имели такое же влияние при обучении, как и более частые типы.

Класс RoomTypeElement содержит информацию об одной такой группе и хранит только идентификатор типа комнаты и список названий номеров, которые ему соответствуют.

Далее необходимо загрузить исходные данные в объекты вышеописанных классов. Загружать на обучение сразу все – не лучшая идея (особенно, когда речь идет о миллионах названий номеров, когда необходимо обучить сеть с нуля), поэтому уместно будет разделить все данные на блоки меньшего размера и загружать их постепенно. Этим займется класс DataLoader.

Также необходимо разделить все исходные данные на обучающую и проверяющую (тестовую) выборку. Усредненно-оптимальный процент разбиения – 85 к 15 соответственно. Самая ответственная часть реализации нейронной сети – обучение. Передавая данные небольшими блоками, механизм PyTorch[8] будет подстраивать веса нейросети.

После обучения блока необходимо проверить, насколько лучше нейросеть стала справляться с классификацией номеров. По результатам проверки строятся метрики: потери, точность, F1-меры (микро и макро).

Для предотвращения переобучения (когда модель идеально справляется с классификацией данных из обучающей выборки, но хуже справляется с тестовыми данными) модели реализована ранняя остановка обучения.

После завершения обучения (это достаточно длительный процесс) модель готова к классификации любых входных параметров с доверительной точностью (порядка 99.8%). Полученный модуль можно встроить в любую другую систему как самостоятельный микросервис и использовать изолированно для быстрого получения результатов.

О скорости обучения и работы подробнее в следующем разделе. Важно отметить, что ключевая особенность работы нейросетей – расчеты на видеопамяти, а не на центральном процессоре как в предыдущих методах. Расчеты на видеокартах значительно увеличивают скорость выполнения операций за счет автоматизированного распараллеливания процессов. Дальнейший анализ будет учитывать эту особенность подхода нейронных сетей.

Анализ производительности рассмотренных механизмов

Производительность будем сравнивать по скорости классификации. Все тесты будут проводиться на одном и том же ПК, в проверке будет задействовано переменное число названий номеров, по результатам времени обработки которых построим наглядный график. На текущий момент поставщик имеет количество комнат, кратное миллиону, однако классифицируются они зачастую маленькими группами.

Конфигурация тестового стенда:
CPU: Ryzen 5 2600,
RAM: 32 GB DDR4,
GPU: GeForce 1660.

В таблице 1 приведена сводная информация по результатам экспериментов, а на рисунке 1 представлены графики зависимости времени расчета от количества классифицируемых номеров и подхода. Примечание: анализ производительности нейронной сети проводится с учетом расчетов на видеокарте; все полученные результаты по итогам классификации должны быть сохранены для возможности дальнейшей работы с ними.

Рисунок 1 – График зависимости времени классификации от количества записей для различных подходов

Таблица 1 - Результаты классификации различными методами

	Время обработки 10 тыс. записей	Время обработки 50 тыс. записей	Время обработки 100 тыс. записей	Время обработки 150 тыс. записей
Регулярные выражения БД	17 сек	75 сек	165 сек	236 сек
Регулярные выражения C#	4 сек	33 сек	114 сек	234 сек
Регулярные выражения C# с учетом загрузки данных из БД	5 сек	36 сек	117 сек	245 сек
Нейронная сеть	6 сек	21 сек	41 сек	61 сек
Нейронная сеть с учетом загрузки данных из БД	7 сек	24 сек	42 сек	64 сек

На графике зависимости времени классификации от количества записей и типа алгоритма видим значительное преимущество подхода с нейронной сетью. Подходы с регулярными выражениями показывают сходный результат, однако классификация в C#-коде имеет явно выраженную экспоненциальную зависимость, в то время как обработка в базе данных – линейную.

Выводы

По результатам проделанной работы можно сделать следующие выводы:

- в пределах десяти тысяч записей разница во времени между тремя рассмотренными подходами несущественна;
- свыше десяти тысяч названий наилучший результат с большим отрывом показывает нейронная сеть.

До ста пятидесяти тысяч записей также можно использовать подход с реализацией классификации в C#-коде, однако из-за экспоненциального роста времени расчета классификация сверх ста пятидесяти тысяч в C#-коде не целесообразна.

Подход с применением нейронной сети обрабатывает данные значительно быстрее, однако требует дополнительных ресурсов для внедрения:

- предварительная классификация (в том числе ручная для новых типов комнат);
- обучение модели, место в памяти для хранения модели, дополнительная аппаратура (GPU);
- доработка подхода до состояния готового к внедрению сервиса.

Подходы с регулярными выражениями также требуют предварительной классификации в случае появления новых типов комнат или частных случаев, однако не требует дополнительных затрат на внедрение и оптимизацию. Реализация в коде позволит внедрять алгоритм как самостоятельный настраиваемый сервис, реализация в БД не дает существенных преимуществ.

Все три изученные подхода имеют как свои преимущества, так и недостатки, которые необходимо учитывать при достижении поставленных бизнес-целей.

Литература

1. Мартин, Р. С. Чистый код: создание, анализ и рефакторинг. Библиотека программиста / Р. С. Мартин. — СПб.: Питер, 2013. — 464 с.: ил.

2. Суркова, Н. Е. Методология структурного проектирования информационных систем: Монография / Н. Е. Суркова, А. В. Остроух. — Красноярск: Научно-инновационный центр, 2014. — 190 с.

3. Бейли, Л. Изучаем SQL / Л. Бейли. — СПб.: Питер, 2012. — 592 с.: ил.

4. Шилдт, Г. C# 4.0: полное руководство. — М.: Вильямс, 2011. — 1056 с.

5. Стивенс, Э. PyTorch. Освещая глубокое обучение / Э. Стивенс, Л. Антига, Т. Виман. — СПб.: Питер, 2022. – 576 с.: ил.

6. Общие сведения о минимальных API [Электронный ресурс] / Интернет-Ресурс. - Режим доступа: https://learn.microsoft.com/ru-ru/aspnet/core/fundamentals/minimal-apis?view=aspnetcore6.0&viewFallbackFrom=aspnetcore-2.0

7. Введение в PyTorch [Электронный ресурс] / Интернет-Ресурс. - Режим доступа: https:// pythonist.ru/vvedenie-v-pytorch/?ysclid=lv3j7of2o1668414893

8. Официальная документация PyTorch [Электронный ресурс] / Интернет-Ресурс. - Режим доступа: https://pytorch.org/docs/stable/index.html

9. Джонсон, Д. Учебное пособие по PyTorch: регрессия, пример классификации ихобраэений [Электронный ресурс] / Д. Джонсон. / Интернет-Ресурс. - Режим доступа: https://www.guru99.com/ru/pytorch-tutorial.html

10. Лонца, А. Л. Алгоритмы обучения с подкреплением на Python / А. Л. Лонца / пер. с англ. А. А. Слинкина. – М.: ДМК Пресс, 2020. – 286 с.: ил.

Истягин А.О., Рычка О.В. Сравнение производительности различных подходов в задаче классификации. *В статье рассмотрены различные подходы в решении задачи классификации, реализована нейронная сеть для классификации типов номеров отелей, проведен анализ производительности работы регулярных выражений и нейронных сетей, построены графики зависимости скорости классификации больших объемов данных от выбранного алгоритма, сделаны выводы о целесообразности и уместности применения различных подходов. Сделан вывод, что подход с применением нейронной сети обрабатывает данные значительно быстрее, однако требует дополнительных ресурсов для внедрения.*

Ключевые слова: *программная инженерия, задача классификации, Python, C#, анализ производительности*

Istyagin A., Rychka O. Comparing the performance of different approaches in a classification task*. The article considers various approaches to solving the classification problem, implements a neural network for classifying hotel room types, analyzes the performance of regular expressions and neural networks, plots the dependence of the classification rate of large amounts of data on the chosen algorithm, and concludes on the expediency and appropriateness of using various approaches. It is concluded that the neural network approach processes data much faster, but requires additional resources for implementation.*

Keywords: *software engineering, classification problem, Python, C#, performance analysis*

Статья поступила в редакцию 07.12.2023
Рекомендована к публикации профессором Федяевым О. И.

УДК 004.021

Анализ методов преобразования алгоритмов

В. К. Ремизов, А. В. Григорьев
ФГБОУ ВО «Донецкий национальный технический университет», г. Донецк
e-mail: vsevolod.remizov@gmail.com, grigorievalvl@gmail.com

Аннотация
В статье рассмотрены методы преобразования алгоритмов. Описан алгоритм преобразования программ к виду, имеющему ограниченную когнитивную сложность. Определены место и роль алгоритма в рамках существующих подходов. Предложенный алгоритм обеспечивает решение задачи ограничения когнитивной сложности программ, но имеет недостаток, который требует его доработки. Перспективным направлением исследования является доработка и реализация описанного алгоритма преобразования программы к виду, имеющему ограниченную когнитивную сложность.

Введение

При написании программ часто возникает необходимость преобразования алгоритма с целью снижения его вычислительной (временной) или когнитивной (восприятие структурной сложности людьми) сложности. Для этого фрагменты алгоритма заменяются эквивалентными, то есть такими алгоритмами, которые имеют одну и ту же область определения и реализуемые функции, но разную систему правил [1]. Этой необходимостью и обусловлена актуальность данной работы.

Цель предлагаемой статьи – провести анализ основных методов преобразования алгоритмов для последующей реализации преобразования программ к виду, имеющему ограниченную когнитивную сложность.

Задача ограничения когнитивной сложности заключается в снижении сложности структуры алгоритма с целью упрощения понимания этого алгоритма людьми с разным уровнем подготовки. Она включает в себя:

- определение языка для описания моделей;
- построение модели предметной области;
- построение меры когнитивной сложности;
- упрощение построенной модели, ориентируясь на уровень ее когнитивной сложности [2-6].

Для достижение поставленной цели необходимо решить следующие задачи:

– провести анализ основных методов преобразования алгоритмов;

– провести анализ алгоритма преобразования программ к виду, имеющему ограниченную когнитивную сложность;

– определить место и роль алгоритма в рамках существующих подходов;

– определить перспективные направления работы.

Методы преобразования алгоритмов

На данный момент существуют следующие, наиболее характерные среди различных типов, методы преобразования алгоритмов:

– метод, использующий графовую модель алгоритма;

– метод, использующий свойства множеств, предикатов и операций над ними;

– метод, использующий предикативные грамматики;

– метод, использующий разрезание гиперграфа;

– и другие.

Рассмотрим данные методы и опишем их достоинства и недостатки с точки зрения когнитивной сложности.

Метод, использующий графовую модель алгоритма

В [7] рассмотрена задача эквивалентного преобразования алгоритма с предикатами простого типа на парных комбинациях, что позволило автору создать новую классификацию методов преобразования. В качестве модели алгоритма в данном методе используется графовая модель, представленная на рис. 1. В ней вершинам графа соответствуют операторы обработки данных, ветвления и слияния потоков управления. Выполнение оператора ветвления означает, что множество операторов разделяется на две части: одна с предикатом условия «истина», а другая с предикатом «ложь». А выполнение оператора слияния означает, что далее операторы объединяемых множеств выполняются одинаково [7].

Предлагаются следующие преобразования структуры алгоритма, которые выбираются на основании анализа предикатов [7]:

– инверсия условий передачи управления;

– изменение последовательности слияния потоков управления;

– вынесение начала ветвления или условия выхода из цикла из конструкции ветвления;

– разделение потока управления в точке слияния, за которой следует оператор изменения данных;

– разделение потока управления в точке слияния, за которой следует оператор ветвления;

– изменение последовательности ветвления потоков управления;

– вынесение оператора изменения данных из ветвления или изменение данных до выхода из цикла;

– внесение оператора изменения данных в ветвление или проверка условия выхода из цикла до изменения данных.

– Достоинства и недостатки данного метода представлены в табл. 1.

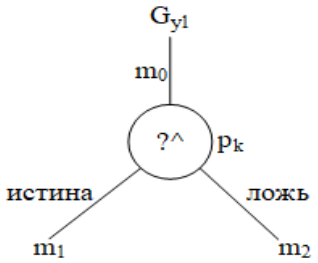


Рисунок 1 – Представление условия передачи управления в графовой модели

Таблица 1 – Достоинства и недостатки метода 1

Достоинства	Недостатки
Предложены всевозможные преобразования алгоритмов, в том числе с циклами	Не доказано, что предложенных преобразований достаточно для любого взятого алгоритма
Предложены механизмы, позволяющие понять эквивалентные ли алгоритмы с разной формой записи	Преобразования не оптимальны, увеличивают структурную сложность

Метод, использующий свойства множеств, предикатов и операций над ними

В [8] рассмотрены эквивалентные преобразования алгоритма, включающего множества и предикаты, с целью снижения его вычислительной сложности. Для этого в алгоритме находятся соответствующие операторы и заменяются на более эффективные.

Выделяются следующие преобразования, использующие свойства множеств и операций над ними [8]:

– удаление элемента множества замещением;

– замена выражения алгебры подмножеств логически эквивалентным и требующим меньшего числа операций;

– выбор порядка выполнения операции пересечения более чем двух множеств;

– использование свойства дистрибутивности операций над множествами.

Преобразования, использующие свойства предикатов и операций над ними [8]:

– задание предикатами связей между множествами;

– определение результата операции над характеристическими множествами предикатов как характеристического множества результата операции над ними;

– использование операции композиции над двухместными предикатами;

– использование свойств логических операций над предикатами.

Достоинства и недостатки данного метода представлены в табл. 2.

Таблица 2 – Достоинства и недостатки метода 2

Достоинства	Недостатки
Представлена классификация способов снижения вычислительной сложности алгоритма, использующих свойства множеств, предикатов и операций над ними	Нет четкого алгоритма преобразования
	Не затрагивается структурная сложность

Метод, использующий предикативные грамматики

В [9] алгоритм представляется в виде структурного графа (см. рис. 2), полученного при помощи предикативной грамматики.

Структурные предикативные грамматики используются для определения и анализа семантической структуры в виде графа, в основе которого лежит семантическое дерево программы. В таких грамматиках для описания структуры программы используются языки первого порядка, в которых объектами являются термы, образованные специальными функциями-конструкторами.

Структурные предикативные грамматики обеспечивают построение конечного ориентированного упорядоченного графа, который называется структурным [9]. После построения структурного графа на его основе строится граф зависимостей по данным (рис. 3). Граф зависимостей по данным отображает зависимости между операторами алгоритма и затем используется для оптимизации программы, например, путем удаления операторов, которые присваивают значение не использующейся переменной [9]. Структурно предикативные грамматики используются как для снижения вычислительной, так и когнитивной сложности. Достоинства и недостатки данного метода представлены в табл. 3.

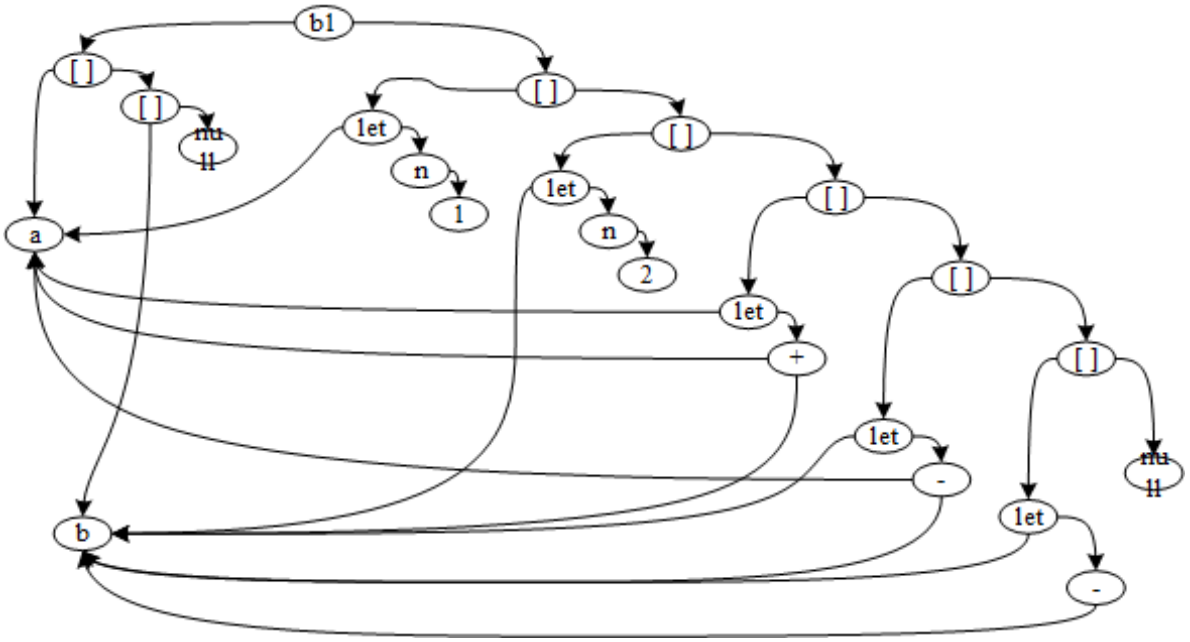


Рисунок 2 – Представление алгоритма в виде структурного графа

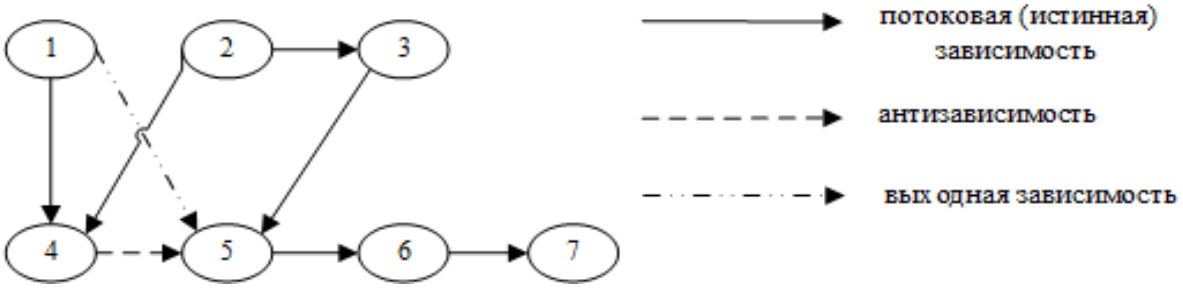


Рисунок 3 – Граф зависимостей по данным

Таблица 3 – Достоинства и недостатки метода 3

Достоинства	Недостатки
Представлен алгоритм унификации.	Нет четкого алгоритма преобразования
Представлен алгоритм построения графа зависимостей по данным	Решается частная задача

Метод, использующий разрезание гиперграфа

В [10] представлены способы преобразования алгоритмов при помощи разрезания гиперграфа, с целью снижения вычислительной сложности алгоритма. Под гиперграфом понимается граф, в котором ребра могут соединять любое множество вершин (см. рис. 4).

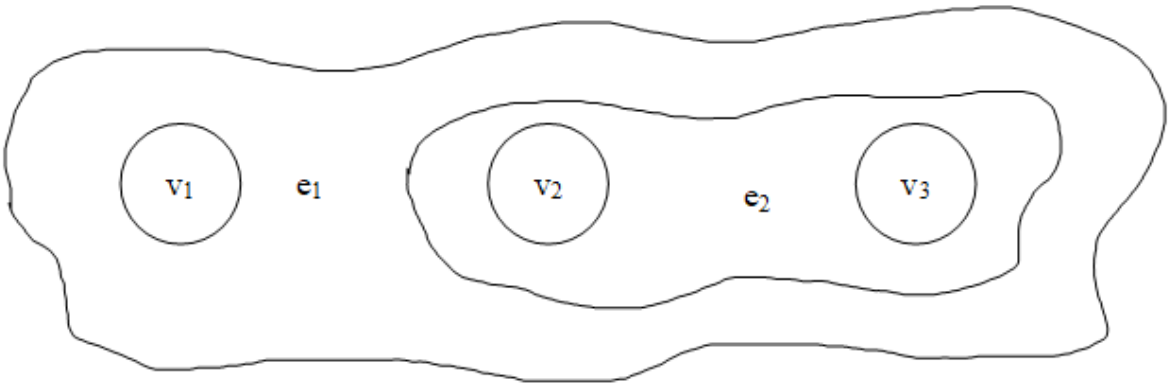


Рисунок 4 – Гиперграф

Предложенный метод представляет собой итерационный алгоритм парного замещения – делается парная перестановка вершин из двух разрезов, а затем выполняется оценка при помощи критерия оптимальности – минимума ребер, попадающих в разрез.

В результате в разрез попадают или из разреза уходят только ребра, связанные с этими вершинами. Достоинства и недостатки данного метода представлены в табл. 4.

Таблица 4 – Достоинства и недостатки метода 4

Достоинства	Недостатки
Представлен способ снижения вычислительной сложности алгоритма при помощи разрезания гиперграфа	Трудоемкость процесса разрезания из-за парных перестановок

Выводы по анализу методов

Ограничение когнитивной сложности является более сложной задачей, чем те задачи, которые были описаны в данном разделе. Это потребовало разработки нового алгоритма, который развил используемые в предыдущих методах идеи.

Алгоритм преобразования программ к виду, имеющему ограниченную когнитивную сложность.

Рассмотрим алгоритм преобразования программ к виду, имеющему ограниченную когнитивную сложность, определим его достоинства и недостатки и наметим перспективы дальнейшего развития.

В [2] описан способ преобразования алгоритма, который снижает его структурную сложность, не затрагивая вычислительную сложность. При этом сам алгоритм не меняется, а меняется лишь форма его подачи.

Данный метод использует структурные предикативные грамматики, и-или дерево и граф связей для представления алгоритма в виде состава блоков и связей между ними.

Затем полученная структура изменяется путем перемещения части подблоков прототипа в новые подблоки. Т.е. часть алгоритма, связанная по смыслу, объединяется в подмодуль, который имеет ограниченную когнитивную сложность, и затем заменяется вызовом этого подмодуля.

Рассмотрим алгоритм работы метода. Данный алгоритм построен на базе описанных ранее методов, с учетом их недостатков. Исходный прототип ***Р***, состоящий из множества подблоков ***Б***, проверяется на допустимую когнитивную сложность.

Если сложность выше заданной, приступаем к преобразованиям. Для этого формируем пустой список ***Sп***, в который затем будем вносить извлекаемые подблоки. На базе этого списка создаем новый «пустой» блок-аккумулятор ***П***, имеющий пустой список свойств, составляющих его внешнюю и внутреннюю границу, и вносим его в ***Б***. Формируем список запрещенных блоков ***Sz*** и вносим в него внутреннюю границу блока ***Б1*** и блока-аккумулятора ***П*** (см. рис. 5).

Далее ищем набор подблоков ***{Бs}***, не относящихся к запрещенным блокам и близких к находящимся в списке ***Sп***.

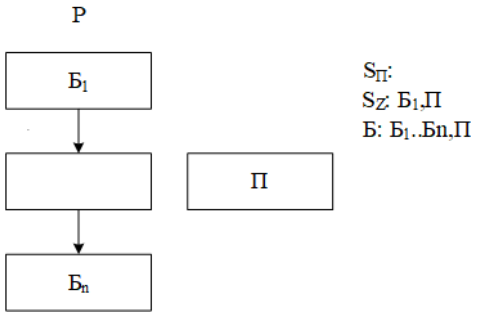


Рисунок 5 – Алгоритм работы метода. Часть 1

Для этого вводится критерий близости, который определяется составом и весом эквивалентных отношений. Состав таких отношений может характеризоваться количеством: связей по свойствам, имен в цепочке идентификации типа блока, наименования типов свойств, значений свойств, заданных в порядке взаимного включения типов свойств.

Если список ***Sп*** еще пустой, а найденных подблоков несколько, тогда определяем блок ***Бs***, удаление которого снизит когнитивную сложность больше остальных (см. рис. 6).

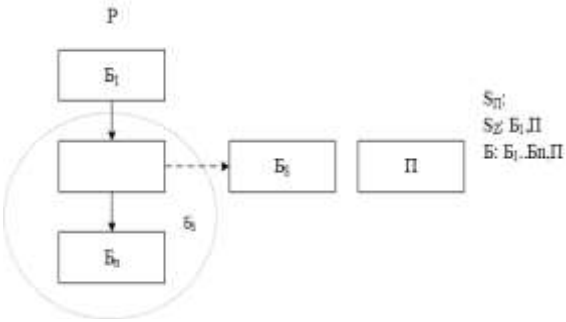


Рисунок 6 – Алгоритм работы метода. Часть 2

Найденный блок вносим в состав списка ***Sп***, формируем новый блок-аккумулятор ***П*** со списком свойств, составляющих его границу, и вносим его в ***Б***. Формируем связи блока ***Бs*** с подблоками блока ***П***. Границы блока ***П*** пополняем свойствами, посредством которых блок ***Бs*** связан с внешней средой. Удаляем из исходного прототипа ***Р*** блок ***Бs*** и включаем связи с новым блоком ***П*** (см. рис. 7).

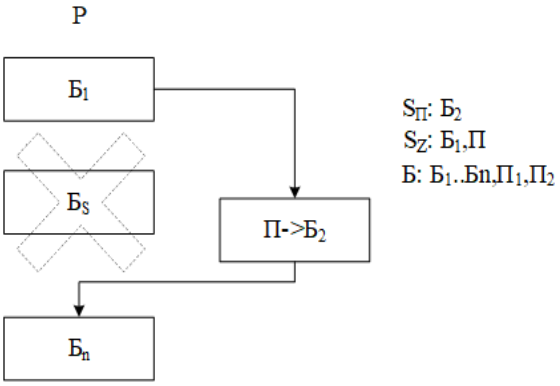


Рисунок 7 – Алгоритм работы метода. Часть 3

Если ***Sz=P***, то новый блок ***П*** – искомый, а значит алгоритм заканчивается. В противном случае оцениваем когнитивную сложность блока ***П***, если она не превышает заданную, то выбираем следующий вторичный подблок (см. рис. 8).

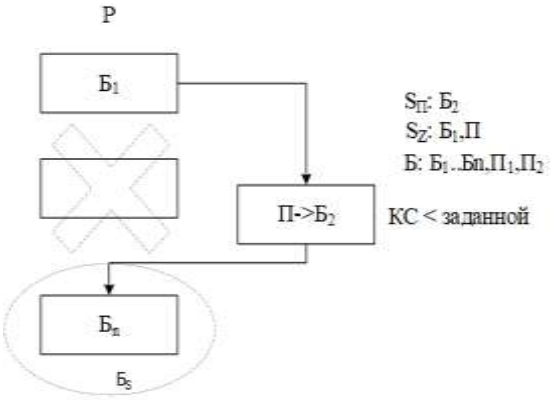


Рисунок 8 – Алгоритм работы метода. Часть 4

Если же подблок ***П*** превысил необходимый уровень когнитивной сложности, то возвращаемся к предыдущему варианту ***Sn***, ***P*** и ***П***, вносим блок ***Бs*** в список запрещенных блоков ***Sz*** и переходим к выбору следующего вторичного блока.

Финальным этапом алгоритма является создание нового «пустого» блока ***П***, имеющего пустой список подблоков и свойств, составляющих его границы, и внесение его в ***Б*** (см. рис. 9-10).

Достоинства и недостатки данного метода представлены в табл. 5.

Место алгоритма показано на рис. 11.

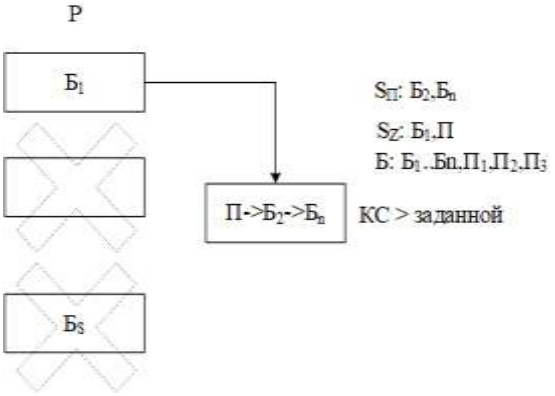


Рисунок 9 – Алгоритм работы метода. Часть 5

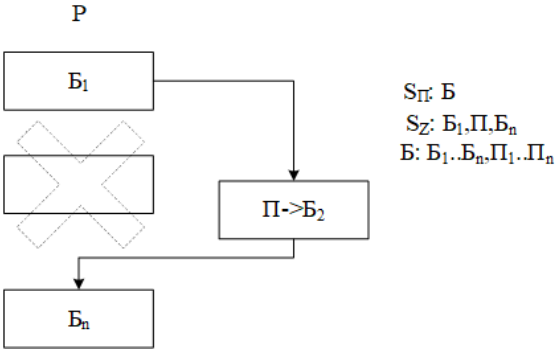


Рисунок 10 – Алгоритм работы метода. Часть 6

Таблица 5 – Достоинства и недостатки метода 5

Достоинства	Недостатки
Представлен агрегатный подход преобразования алгоритма к виду, имеющему ограниченную когнитивную сложность	Не гарантируется, что вынесенный подблок будет иметь необходимый уровень когнитивной сложности
Не используются парные перестановки, т.к. необходимый блок выбирается сразу благодаря введенному критерию близости	
Все части алгоритма, включая циклы, представляются в виде состава блоков, имеющих свои входы и выходы	

Описанный алгоритм использует фрагменты описанных ранее методов (графовое представление алгоритма, свойства предикатов, структурные предикативные грамматики, граф зависимостей по данным и разрезание гиперграфа).

Рисунок 11 – Место алгоритма в рамках существующих подходов

Однако, он разрабатывался с учетом устранения их недостатков. В результате:

- понижается структурная сложность;
- решается более сложная задача;
- преобразования более оптимальные, это достигается за счет ухода от парных перестановок.

Выводы

В статье проведен анализ основных методов преобразования алгоритмов, а также анализ алгоритма преобразования программ к виду, имеющему ограниченную когнитивную сложность. Определены место и роль алгоритма в рамках существующих подходов. Описанный алгоритм обеспечивает решение задачи ограничения когнитивной сложности программ, но имеет недостаток, который требует его доработки.

Перспективным направлением исследований является доработка и реализация описанного алгоритма преобразования программы к виду, имеющему ограниченную когнитивную сложность, так как реализация данного метода может существенно упростить программный код и сделать его более понятным для восприятия.

Литература

1. Алферова, З. В. Теория алгоритмов: учебное пособие по специальности "Организация механизированной обработки экономической информации" / З. В. Алферова. – М.: Статистика, 1973. – 164 с.

2. Григорьев, А. В. Ограничение когнитивной сложности моделей / А. В. Григорьев // Прогрессивные технологии и системы машиностроения: Международный сб. научных трудов. - Донецк: ДонГТУ, 2000. - Вып. 10. - С. 49-58.

3. Григорьев, А. В. Методика тестирования для определения когнитивной сложности моделей различных предметных областей / А. В. Григорьев // Научные труды Донецкого государственного технического университета. Серия: Информатика, кибернетика и вычислительная техника. - Донецк: ДонГТУ, 1999. – Вып. 6 (ИКВТ-99). - С. 246-251.

4. Григорьев, А. В. Оценка когнитивной сложности моделей / А. В. Григорьев // Научные труды Донецкого государственного технического университета. Серия: Информатика, кибернетика и вычислительная техника. - Донецк: ДонГТУ, 1999. – Вып. 6 (ИКВТ-99). – С. 252-259.

5. Григорьев, А. В. Адаптивная система ограничений на сложность при синтезе новых решений в интеллектуальных САПР / А. В. Григорьев // Искусственный интеллект. – Донецк, 2001. – N 2. – С. 152–167.

6. Григорьев, А. В. Комплекс средств и методов работы с формальными грамматиками в семиотической концептуальной модели предметной области интеллектуальных САПР / А. В. Григорьев // Информатика и кибернетика. – Донецк: ДонНТУ, 2017. - №1(7). – С. 46-72.

7. Иванова, Г. С. Эквивалентные преобразования структур алгоритмов [Электронный ресурс] / Г. С. Иванова // Машиностроение и компьютерные технологии, 2009. - №11. - URL: https://cyberleninka.ru/article/n/ekvivalentnye-preobrazovaniya-struktur-algoritmov (дата обращения: 19.10.2023).

8. Овчинников, В. А. Оптимизирующие преобразования алгоритмов, использующие свойства множеств, предикатов и операций над ними [Электронный ресурс] / В. А. Овчинников, Г. С. Иванова // Вестник МГТУ им. Н.Э. Баумана. Серия «Приборостроение», 2013. - №4 (93). - URL: https://cyberleninka.ru/article/n/optimiziruyuschie-preobrazovaniya-algoritmov-ispolzuyuschie-svoystva-mnozhestv-predikatov-i-operatsiy-nad-nimi (дата обращения: 19.10.2023).

9. Крицкий, С. П. Реализация оптимизирующих преобразований программ с помощью структурных предикативных грамматик [Электронный ресурс] / С. П. Крицкий, Б. Ю. Тапкинов // Известия вузов. Северо-Кавказский регион. Серия: Естественные науки, 2006. - № S1. - URL: https://cyberleninka.ru/article/n/realizatsiya-optimiziruyuschih-preobrazovaniy-programm-s-pomoschyu-strukturnyh-predikativnyh-grammatik (дата обращения: 19.10.2023).

10. Овчинников, В. А. Способы снижения вычислительной сложности алгоритмов, вытекающие из принципа формирования решений [Электронный ресурс] / В. А. Овчинников // Инженерный журнал: наука и инновации, 2013. - Вып. 11. - URL: http://engjournal.ru/catalog/it/hidden/1046.html (дата обращения 09.11.2023).

Ремизов В.К., Григорьев А.В. Анализ методов преобразования алгоритмов. *В статье рассмотрены методы преобразования алгоритмов. Описан алгоритм преобразования программ к виду, имеющему ограниченную когнитивную сложность. Определены место и роль алгоритма в рамках существующих подходов. Предложенный алгоритм обеспечивает решение задачи ограничения когнитивной сложности программ, но имеет недостаток, который требует его доработки. Перспективным направлением исследования является доработка и реализация описанного алгоритма преобразования программы к виду, имеющему ограниченную когнитивную сложность.*

Ключевые слова: *преобразование алгоритма, вычислительная сложность, когнитивная сложность, граф, множество, предикативная грамматика*

Remizov. V.K., Grigoriev A.V. Analysis of algorithm conversion methods. *The article discusses the methods of converting algorithms. An algorithm for converting programs to a form with limited cognitive complexity is described. The place and role of the algorithm within the framework of existing approaches are determined. The proposed algorithm provides a solution to the problem of limiting the cognitive complexity of programs, but has a drawback that requires its improvement. A promising area of research is the refinement and implementation of the described algorithm for converting a program to a form with limited cognitive complexity.*

Key words: *algorithm conversion, computational complexity, cognitive complexity, graph, sets, predicative grammar*

Статья поступила в редакцию 08.12.2023
Рекомендована к публикации профессором Зори С. А.

УДК 004.853

Анализ специфики и области применения архитектурных шаблонов в развернутых экосистемах

К. А. Терещенко, Р. В. Мальчева
ГОУВПО «Донецкий национальный технический университет»
E-mail: malcheva.raisa@yandex.ru

Аннотация
В статье рассмотрены основные шаблоны системной архитектуры. Сформулирована система оценки их применения в развёрнутых экосистемах. Проведена оценка применимости в рамках сформированной системы. Выбран наиболее применимый шаблон. Описаны дополнительные шаблоны для повышения соответствия определенным обязательным критериям шаблона. Сформулированы критерии оценки архитектур: безопасность, масштабируемость, оптимальность, интегрируемость, сложность и адаптивность, на основании которых сделан вывод, что наиболее применимой является микросервисная архитектура.

Введение

Как отмечают многие авторы, индустриальная фаза роста мировой экономики закончилась в 2018-2020 гг., и ее дальнейшее развитие будет осуществляться под все большим воздействием всеобщей цифровизации [1, 2]. И, если первоначально этот процесс затрагивал отдельные производственные, банковские, управленческие или иные процессы, то современное внедрение цифровых систем происходит повсеместно. Следует отметить, что эффективность их применения не всегда отвечает ожиданиям инвесторов.

Типичным представителем современных цифровых систем являются экосистемы, которые предоставляют крайне разнообразный набор услуг, включающих в себя широкий спектр связанных систем, погружающих клиента в цикл взаимодействующих продуктов. Примером может служить реализация продукта, предоставляющего функционал по поиску и покупке недвижимости, который будет интегрирован с банковскими продуктами для упрощенной процедуры выдачи ипотеки на выбранные клиентом объекты. Таких продуктов, может быть, масса и все они направлены на мотивацию клиента в использовании различных услуг. Поэтому актуальной проблемой является проектирование эффективной архитектуры экосистем с учетом таких ключевых требований как масштабируемость, открытость, неоднородность, безопасность, доступность, и производительность [3-6].

Целью данной статьи является анализ особенностей и области применения шаблонов системной архитектуры с точки зрения эффективности их применения в развернутых экосистемах.

Общая постановка проблемы

Формирование масштабной сети взаимно интегрированных продуктов и систем порождает ряд существенных трудностей, сопряженных в первую очередь с планировкой и реализацией системной архитектуры, которая должны обладать рядом обязательных свойств [7]:

Безопасность. Взаимодействующие системы обменивается критичной для клиента персональной информацией: персональными данными, платежными данными и т.д. Очевидно, что сохранность данных должна быть гарантирована, так как помимо репутационных, компания несет юридические потери.

Масштабируемость. В условиях постоянного роста интеграций между различными системами вследствие увеличения числа информационных потоков, а также потенциальным расширением их объемов, например, в связи с ростом числа клиентов, система должна иметь возможность кратно увеличивать свою производительность, функциональность и объем обрабатываемых данных без значительного снижения эффективности или производительности.

Надежность. Крупные связанные системы зачастую являются высоконагруженными, так как обмениваются большим объемом данных. Это вызывает необходимость формирования надежных отказоустойчивых систем. Также надежность является фактором обеспечения сохраннности данных, утеря которых несет существенные репутационные и юридические риски.

Оптимальность. Система должна быть сконфигурирована таким образом, чтобы обеспечить возможность осуществления функционала с максимальной производитель-

ностью и эффективностью в рамках заданных ограничений, связанных со значительными объемами передаваемой и обрабатываемой информации.

Интегрируемость. В связи со значительной развернутостью и необходимостью взаимодействовать с большим числом других узлов экосистемы, система должна иметь доступный интеграционный интерфейс или доступ к эффективным интеграционным технологиям обмена данными.

Сложность. Сложные системные шаблоны требуют существенных финансовых издержек для реализации и поддержки, что может стать препятствием для реализации. Кроме того поддержка и внедрение сложных системных шаблонов влечет за собой повышение рисков сбоев и ошибок.

Адаптивность. Система должна быть эластичной и иметь возможность к адаптации изменяющихся технических и юридических условий функционирования.

Подбор архитектурного фундамента для достижения данных свойств является комплексной задачей, которая требует выполнения как обзора существующих решений, так и реализации специфичных, направленных на устранение локальных противоречий с изменяемыми требованиями, разработок.

Анализ ключевых архитектурных шаблонов

Рассмотрим основные шаблоны архитектуры, которые сейчас используются в различных системах и приложениях [8, 9].

Слоистая архитектура. Согласно модели слоистой архитектуры, компоненты системы структурированы в горизонтальные слои, каждый из которых отвечает за свою определенную функцию (рисунок 1). Хотя количество и типы слоев не фиксированы, обычно слоистая архитектура включает четыре основных слоя:

- слой представления;
- слой бизнес-логики;
- слой доступа к данным;
- слой абстракции баз данных.

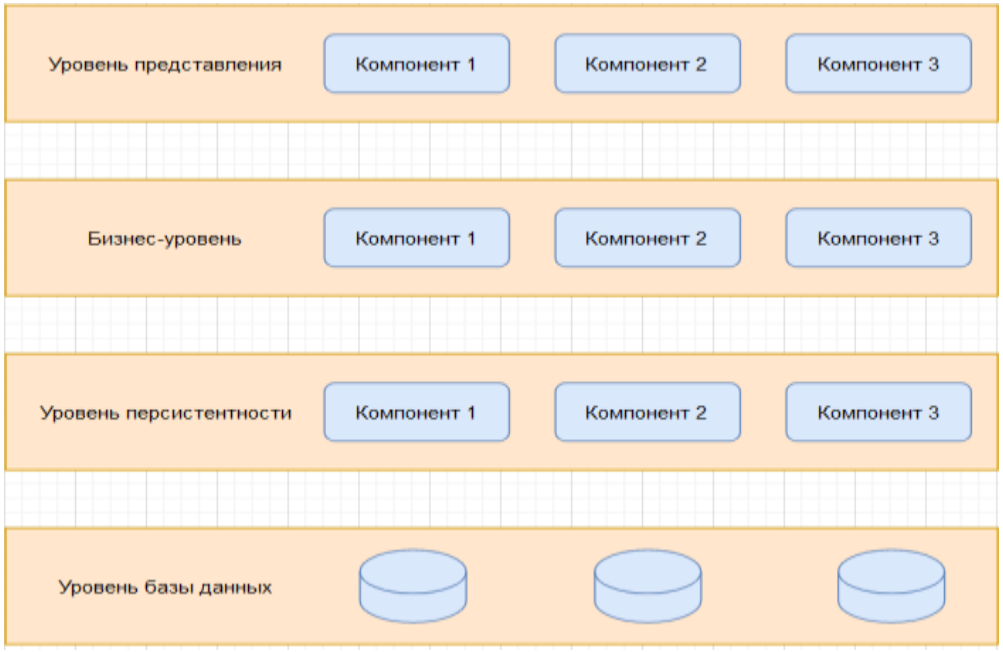


Рисунок 1 – Слоистая архитектура

Иногда слой бизнес-логики и слой доступа к данным объединяются в один, особенно если логика доступа к данным встроена в компоненты бизнес-логики. Таким образом, количество слоев может варьироваться в зависимости от размера и сложности приложения - от трех слоев для небольших проектов до пяти и более для крупных и сложных бизнес-приложений.

Каналы и фильтры. В данной архитектуре (рисунок 2) существует 4 ключевые компоненты:

- генератор (источник) - начальная точка процесса;
- преобразователь (сопоставление) - выполняет преобразование всех или некоторых данных;
- испытатель (редуцирование) - проверяет один или несколько критериев;
- потребитель (приемник) - конечная точка.

Фильтры взаимодействуют друг с другом через коммуникационные каналы.

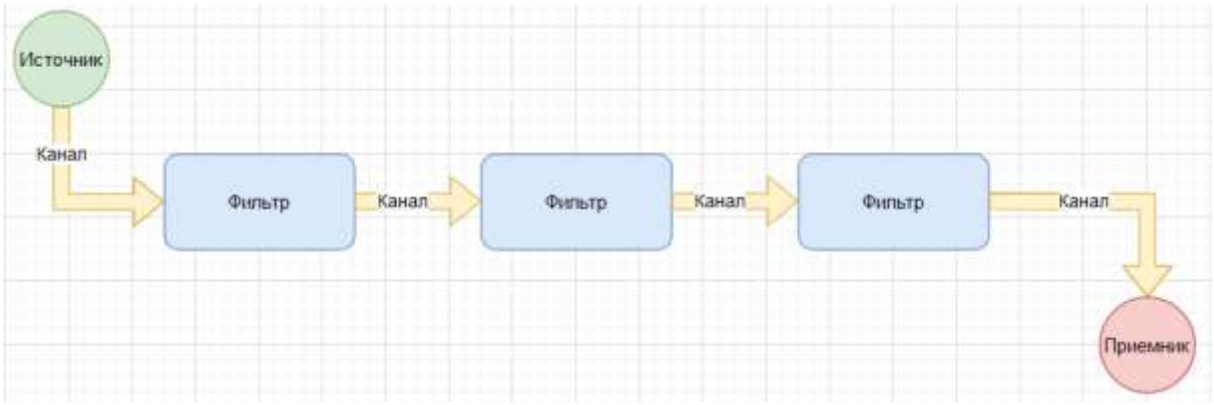


Рисунок 2 – Каналы и фильтры

Основная концепция заключается в том, что каждый канал типа "точка-точка" для повышения производительности и является однонаправленным, принимая данные от одного источника и направляя их к приемнику.

Микроядерная архитектура. Архитектура микроядра состоит из двух видов компонентов (рисунок 3): основной системы и модулей-плагинов. Функциональная часть приложения разделена между независимыми модулями-плагинами и основной системой, что позволяет обеспечить гибкость, расширяемость и изоляцию функций приложения и пользовательской логики обработки данных.

Рисунок 3 – Микроядерная архитектура

Модульный монолит. Это концепция, при которой в рамках одной системы кодовая база разделяется на модули, часто реализующие различные области функциональности. Между объектами функционального модуля формируется «сильная» связанность (объекты модуля созависимы друг от друга и изменения одного зачастую влечет цепочку изменений в других). Между самими же модулями слабый тип связанности, что позволяет трансформировать конкретный модуль без необходимости трансформировать остальные (Рисунок 4.)

Микросервисная архитектура. Архитектура микросервисов (рисунок 5) состоит из нескольких отдельных сервисов, каждый из которых работает в собственном процессе и выполняет единственную функцию. Каждый сервис взаимодействует с другими сервисами через API. Это позволяет разработчикам создавать и обновлять приложения независимо друг от друга, улучшая масштабируемость и гибкость разработки.

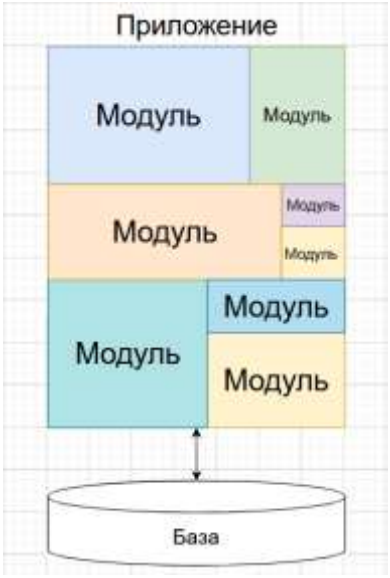


Рисунок 4 - Модульный монолит

Сравнение шаблонов дано в таблице.

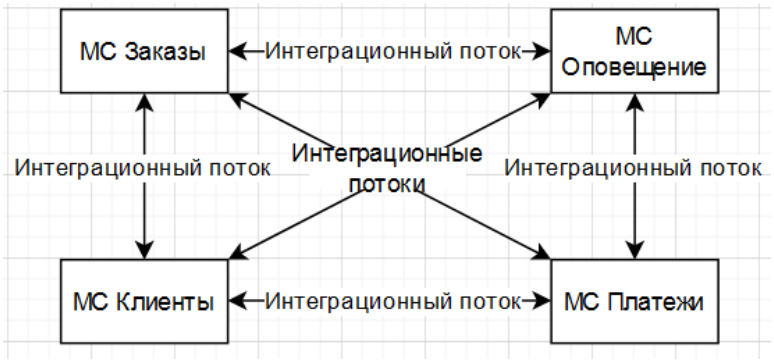


Рисунок 5 – Микросервисная архитектура

Таблица - Сравнение шаблонов системной архитектуры

Название шаблона	Значимые Метрики	Соответствие метрики	Обоснование
Слоистая архитектура	1. Безопасность	Средняя	Архитектура позволяет изолировать чувствительные данные и функциональности в отдельных слоях, что уменьшает риск их компрометации, тем не менее в рамках реализации слоенной архитектуры могут возникнуть дополнительные точки для атаки злоумышленников, так как каждый слой представляет собой потенциальную уязвимую точку.
	2. Масштабируемость	Низкая	Из-за тесной связанности и монолитности данного Шаблона системы, созданные с его использованием обычно затруднительно масштабировать. Слоистую архитектуру можно масштабировать, разделяя слои на отдельные физические экземпляры или копируя всё приложение на несколько узлов. Однако, в целом, излишняя детализация такой архитектуры приводит к высоким затратам на масштабирование.
	3. Надежность	Средняя	С увеличением числа слоев возрастает сложность конфигурации и взаимодействия между ними, что может привести к ошибкам и уязвимостям, тем не менее упрощенная структура шаблона на малых по объему приложениях минимизирует количество излишних интеграционных потоков и потенциальных точек отказа.
	4. Оптимальность	Низкая	Несмотря на то, что некоторые слоистые архитектуры могут хорошо выполнять свою работу, этот Шаблон не эффективен для высокопроизводительных приложений. Это связано с необходимостью проходить через несколько слоёв архитектуры для выполнения бизнес-запроса.
	5. Интегрируемость	Низкая	Шаблон позволяет выделить отдельный слой под интеграционное взаимодействие, однако донесение запроса до последующих слоев может занять значительный объем времени, так же реализации оркестрации распределение бизнес-запросов не предусматривается в данном шаблоне, что может вызвать трудности в реализации.
	6. Простота	Высокая	Этот шаблон отличается легкостью в разработке, в основном потому, что он хорошо известен и не слишком сложен в реализации. Большинство компаний разрабатывают приложения, опираясь на разделение компетенций по уровням (например, представление, бизнес-логика, база данных), поэтому данный шаблон естественным образом становится предпочтительным выбором для создания многих типов бизнес-приложений.
	7. Адаптивность	Средняя	Хотя возможно изолировать изменения с использованием концепции изолированных слоев, внесение изменений в структуре архитектуры затруднено из-за того, что реализации монолитны и компоненты в определенной степени сопряжены между собой.

Продолжение таблицы

Каналы и фильтры	1. Безопасность	Средняя	Система фильтров позволяет устанавливать проверки безопасности, фильтрующие данные соответствующим образом, тем не менее, каналы, как интеграционные потоки, представляют собой дополнительные узлы риска. Также система нуждается в комплексном конфигурировании взаимодействия: настройка фильтров, установка необходимого типа взаимодействия с поставщиком данных и т.д., что повышает риск допущения уязвимостей.
	2. Масштабируемость	Низкая	Шаблон используется для ограниченного ряда решений. В основном для формирования систем, организующих обмен данными между другими системами. Сама структура решения не подразумевает значительного диапазона вариаций масштабирования.
	3. Надежность	Средняя	Шаблон обладает надежностью в контексте правильной его конфигурации и использования подходящей инфраструктуры. Система каналов позволяет устанавливать доступный мониторинг на способность канала отслеживать уязвимости, потерю данных и проводить необходимый аудит. Тем не менее, сложная система фильтрации и установленных на ней проверок может приводить к отсеиванию критически важной информации, а также последующей ее потери, Поэтому надежность шаблона сильно зависит от реализации.
	4. Оптимальность	Высокая	В целом, шаблон достигает высокой оптимальности и производительности благодаря асинхронным возможностям (передача сообщения источником без необходимости получения ответа от потребителя), простоте реализации и доступности решений в рамках современной инфраструктуры.
	5. Интегрируемость	Высокая	Шаблон используется как система регулирования интеграции между системами, Поэтому подразумевает использование удобного для интеграции интерфейса.
	6. Сложность	Средняя	Разработка может быть трудной из-за асинхронной структуры шаблона, а также из-за необходимости формулирования контрактов и предусмотрительного обработчика ошибок в коде для сбойных обработчиков событий и брокеров. Шаблоны сильно зависят от конфигурирования, потому требуют тщательного подхода к реализации. Тем не менее простейший концепт шаблона можно создать без значительных ресурсных затрат.
	7. Адаптивность	Низкая	Шаблон используется для ограниченного ряда решений. В основном для формирования систем, организующих обмен данными между другими системами. Решение не может быть адаптировано для реализации значительного ряда концепций.
Микроядерная архитектура	1. Безопасность	Высокая	Обеспечивает высокий уровень безопасности, поскольку все драйверы устройств работают в отдельном от ядра системы адресном пространстве. Это снижает риск повреждения и взлома системы.
	2. Масштабируемость	Низкая	Большинство реализаций микроядерной архитектуры предназначены для использования в продуктовых приложениях и обычно имеют более компактный размер. Они представляются как единый модуль развертывания, что ограничивает их способность к масштабированию. В зависимости от того, как реализованы подключаемые модули, время от времени масштабируемость может быть обеспечена на уровне этих модулей.

Продолжение таблицы

	3. Надежность	Средняя	Основной функционал реализован в ядре системы, а подключаемые плагины представляют собой отдельные плагины, содержащие опциальные возможности, поэтому нарушение работы плагинов не несет в себе критической угрозы для системы. Тем не менее, нарушения в работе непосредственно ядра, могут привести к значительным проблемам и полной потере части критически важной информации.
	4. Оптимальность	Низкая	Микроядерная архитектура требует больше времени для выполнения операций, так как каждый запрос к ядру должен проходить через слой абстракции. Это зачастую приводит к снижению производительности системы.
	5. Интегрируемость	Низкая	Каждая интеграция дополнительной опциональности происходит только в определенном, ограниченном формате «ядро -> плагин», что значительно затрудняет межсистемную интеграцию.
	6. Сложность	Высокая	Для успешной реализации архитектуры необходимо тщательное планирование и контроль за контрактами, что делает этот процесс довольно сложным. Учет версий контрактов, обновление реестров модулей, детализация модулей и широкий выбор методов их подключения являются основными факторами, которые добавляют сложности при внедрении данного подхода
	7. Адаптивность	Высокая	Изменения могут быть изолированы и оперативно внедрены за счет слабо связанных модулей. В общем, система в большинстве микроядерных архитектур быстро стабилизируется и требует лишь минимальных изменений со временем.
Модульный монолит	1. Безопасность	Средняя	В рамках реализации модульного монолита есть возможность изолировать критически важные данные в различных модулях, однако реализация модульного монолита зачастую не подразумевает значительной сепарации данных, в результате, зачастую, компрометация данных БД приводит к потери значимого объема критически важных данных.
	2. Масштабируемость	Средняя	Дополнительная функциональность реализуется через внедрение дополнительных модулей, что в данной архитектуре является шаблонной реализацией. Тем не менее, значительное увеличение числа модулей усложняет процесс стратегирования связей по ходу масштабирования системы, а также усложняет процесс развертывания системы на сервисе.
	3. Надежность	Средняя	Основной функционал реализован в рамках монолитной структуры, что, с одной стороны, снижает число потенциальных точек отказа (интеграционных потоков), а, с другой стороны, из-за незначительной степени сепарации данных нарушение процессов функциониро-вания БД может привести к потери значимого объема критически важных данных.
	4. Оптимальность	Средняя	Взаимодействие между модулями не обременено комплексными потоками данными, что позволяет реализовывать функциональность с высокой производительностью. Однако, при значительном расширении опциональности приложения, количество связей между модулями будет расти, что может привести к падению производительности
	5. Интегрируемость	Средняя	Модульность позволяет создавать отдельные точки для интеграции, однако последующее распределение данных может быть затруднено из-за слабой связанности определенных компонент.
	6. Сложность	Низкая	Данная архитектура является одним из наиболее популярных и естественных способов реализации монолитной архитектуры потому существует значительное число как технических, так и стратегических методологий реализации.

Продолжение таблицы

	7. Адаптивность	Высокая	Новая функциональность внедряется путем создания дополнительных модулей и сопряжения их с необходимыми компонентами, что в контексте обсуждаемой концепции является относительно простым способом.
Микросервисная архитектура	1. Безопасность	Средняя	Микросервисная архитектура подразумевает сепарацию информации по отдельным БД для сервисов, что снижает риск компрометации, тем не менее, сервисы взаимосвязаны между собой интеграционными потоками, каждый из которых может стать точкой кражи информации.
	2. Масштабируемость	Высокая	Приложение разбито на отдельные компоненты, которые могут быть масштабированы по отдельности. Это позволяет точно настроить масштабирование всего приложения в зависимости от его потребностей.
	3. Надежность	Средняя	Микросервисная архитектура подразумевает сепарацию информации по отдельным БД для сервисов, что снижает риск потери данных. Тем не менее, сервисы взаимосвязаны между собой интеграционными потоками, каждый из которых может стать точкой кражи отказа.
	4. Оптимальность	Низкая	В связи с необходимостью формировать дополнительные точки нагрузки в виде интеграционных потоков, связывающих микросервисы, подход не является наиболее оптимальным решением с точки зрения производительности.
	5. Интегрируемость	Высокая	Помимо возможности выделить отдельный сервис для интеграции со сторонними системами, который будет оркестровать данные от других систем, каждый микросервис может иметь свой собственный интеграционный интерфейс, что позволяет выстраивать необходимые потоки данных направленые в требуемые узлы.
	6. Сложность	Низкая	Благодаря выделению функциональных возможностей в отдельные сервис-компоненты, разработка становится проще, поскольку объем работы уменьшается, и каждый компонент изолирован от других. Это уменьшает риск внесения изменений в один компонент, которые могут повлиять на другие, и, следовательно, уменьшает необходимость согласования между разработчиками и командами разработки.
	7. Адаптивность	Высокая	Благодаря концепции отдельных развернутых модулей, изменения вносятся в изолированные сервис-компоненты, что ускоряет процесс развертывания. Приложения, разработанные с учетом этого подхода, имеют низкую взаимосвязь друг с другом, что упрощает внесение изменений.

Анализ таблицы показывает, что наиболее эффективным шаблоном реализации архитектуры является микросервисная архитектура по совокупности признаков, важнейшими из которых являются масштабируемость, интегрируемость и адаптивность.

Следует отметить, что данный шаблон также обладает рядом недостатков и проблем, часть из которых могут привести к критическим инцидентам в масштабных экосистемах, например, относительно низкая оптимальность, средняя надежность и безопасность. В результате экосистемы устанавливают дополнительные условия реализации в той или иной конфигурации, призванные решить возникающие проблемы. Также компании накладывают дополнительные условия реализации, применяя, а иногда и создавая, топические системные шаблоны.

Одним из таких шаблонов является **Circuit Breaker («Автоматический выключатель»)** приведенный на рисунке 6 [10]. При взаимодействии микросервисов возможны ситуации, когда один из сервисов перестает функционировать. Для борьбы с временными сбоями, вызванными, например, медленным сетевым соединением, используются повторные вызовы. Однако при серьезных сбоях, приводящих к полному отказу сервиса, повторные вызовы будут просто расходовать ресурсы.

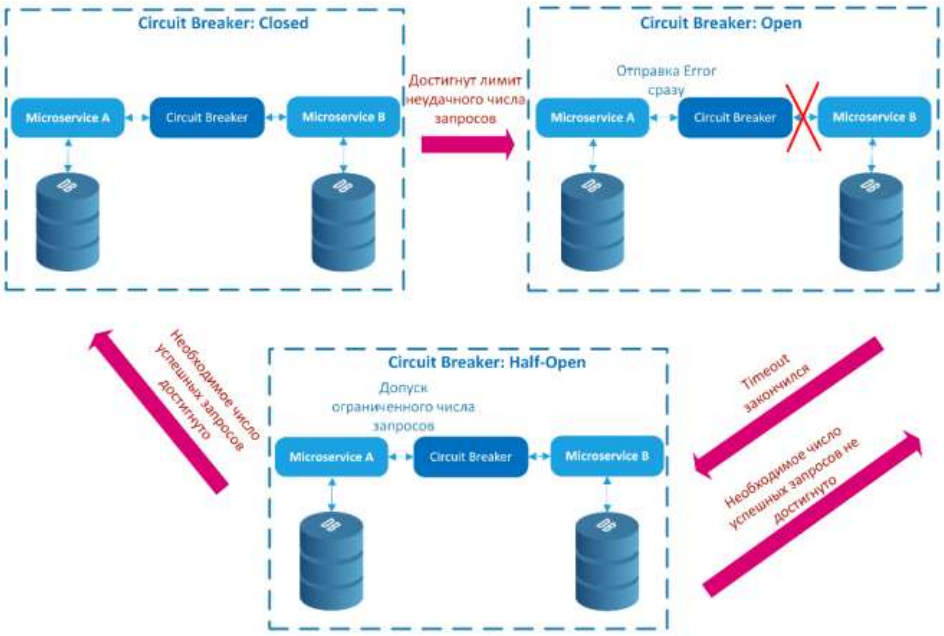


Рисунок 6 – Автоматический выключатель

Чтобы уменьшить вероятность каскадных сбоев в таких случаях, рекомендуется использовать данный шаблон. При его применении микросервис обращается к другому микросервису через прокси-сервер, который контролирует количество сбоев и принимает решение о том, следует ли продолжать отправку запросов или вернуть ошибку. Прокси-сервер может находиться в трех состояниях:

Закрытое. Происходит обмен данными и подсчет сбоев. В случае превышения порога сбоев за определенный период времени, прокси-сервер переключается в режим "Открытое".

Открытое. Все запросы сразу возвращаются с ошибкой. После истечения заданного временного интервала прокси-сервер переходит в режим "Полуоткрытое".

Полуоткрытое. Прокси-сервер ограничивает количество запросов и подсчитывает успешные попытки. Если достигнуто необходимое количество, прокси-сервер переключается в режим "Закрытое", иначе возвращается в режим "Открытое".

Использование шаблона "Circuit Breaker" помогает повысить отказоустойчивость системы и предотвратить распространение сбоев, но требует тщательной настройки и мониторинга.

MDM – повышение безопасности. Как было отмечено, шаблон микросервисной архитектуры обладает средними показателями безопасности. В значительной степени это связано с тем, что зачастую крупные системы для оптимизации работоспособности и повышения отказоустойчивости, применяют шаблон, в рамках которого для каждого микросервиса используется своя БД. Это сильно повышает риск компрометации данных, так как каждая дополнительная база является критическим узлом. Для разрешения данной проблемы периодически применяется шаблон, основанный на подходе MDM (master data management).

Этот шаблон характеризуется сохранением критически значимой информации исключительно в мастер-системах без хранения их в качестве характеристик других сущностей.

Рассмотрим упрощенный абстрактный пример (рисунок 7). Предположим, что есть микросервис, отвечающий за работу с клиентом. Информация о банковских картах клиента является одной из его характеристик, она необходима для формирования отчетности, сохранения чеков, повышения удобства пользователя (чтобы ему не приходилось при каждой покупке заново вводить данные карты) и т.д. Однако, также эти данные являются критически важными и хранение их во всех базах, которые с этой информацией взаимодействуют, приведет к значительному повышению сопутствующих рисков.

В результате сохраняется не конкретная информация о банковской карте клиента, а ссылка на эту информацию в мастер-системе, условный идентификатор данных в базе другого микросервиса. Теперь, когда сервису, отвечающему за работу с клиентами, требуется совершать действия с данными карт, он обращается по сохраненному Id в мастер-систему, и, после проверки на наличие необходимых прав, система передает данные, которые применяются для конкретной операции.

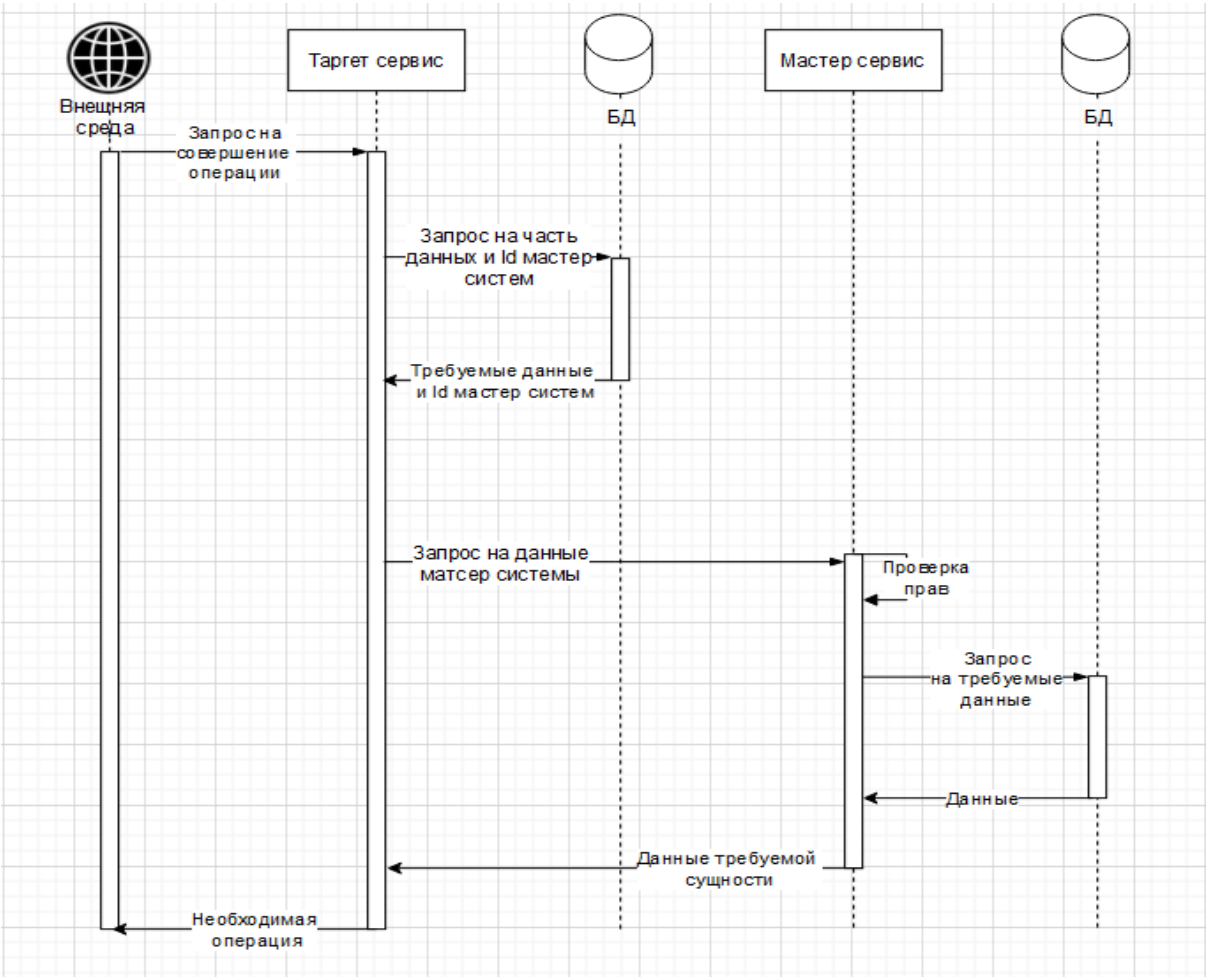


Рисунок 7 – Sequence-диаграмма процесса взаимодействия с мастер-системой

Шаблон «API-шлюз» (API Gateway) – повышение оптимальности *(рисунок 8) [11]*

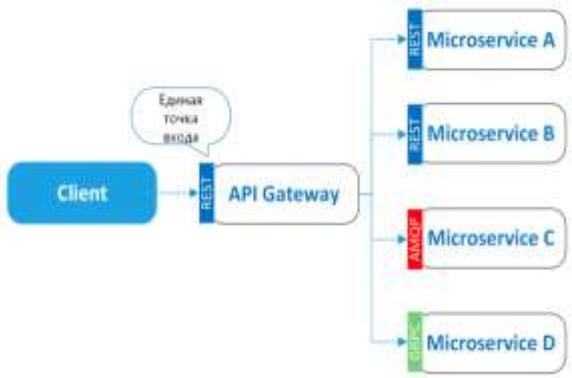


Рисунок 8 - API Gateway

В зависимости от цели использования имеет следующие модификации:

- Gateway Routing. Шлюз работает как прокси-сервер, который направляет запросы клиента к соответствующему сервису.
- Gateway Aggregation. Шлюз используется для разделения клиентского запроса на несколько микросервисов и объединения ответов, возвращаемых клиенту.

Gateway Offloading. Шлюз выполняет общие для сервисов задачи, такие как аутентификация, авторизация, настройка SSL, ведение журналов и другие.

Как было отмечено шаблон микросервисной архитектуры обладает низкими показателями оптимальности. Более того применение различных дополнительных шаблонов, как например MDM может еще сильнее снизить оптимальность работы микросервисов.

Наиболее доступный метод взаимодействия между микросервисами — прямое обращение между ними. В крупных экосистемах с большим числом микросервисов число обращений может быть значительным, более того нередкими становятся ситуации формирования целых цепочек последовательных вызовов от сервиса к сервису значительно повышающих время обработки задач из внешней среды и увеличивающих нагрузку на систему (рисунок 9).

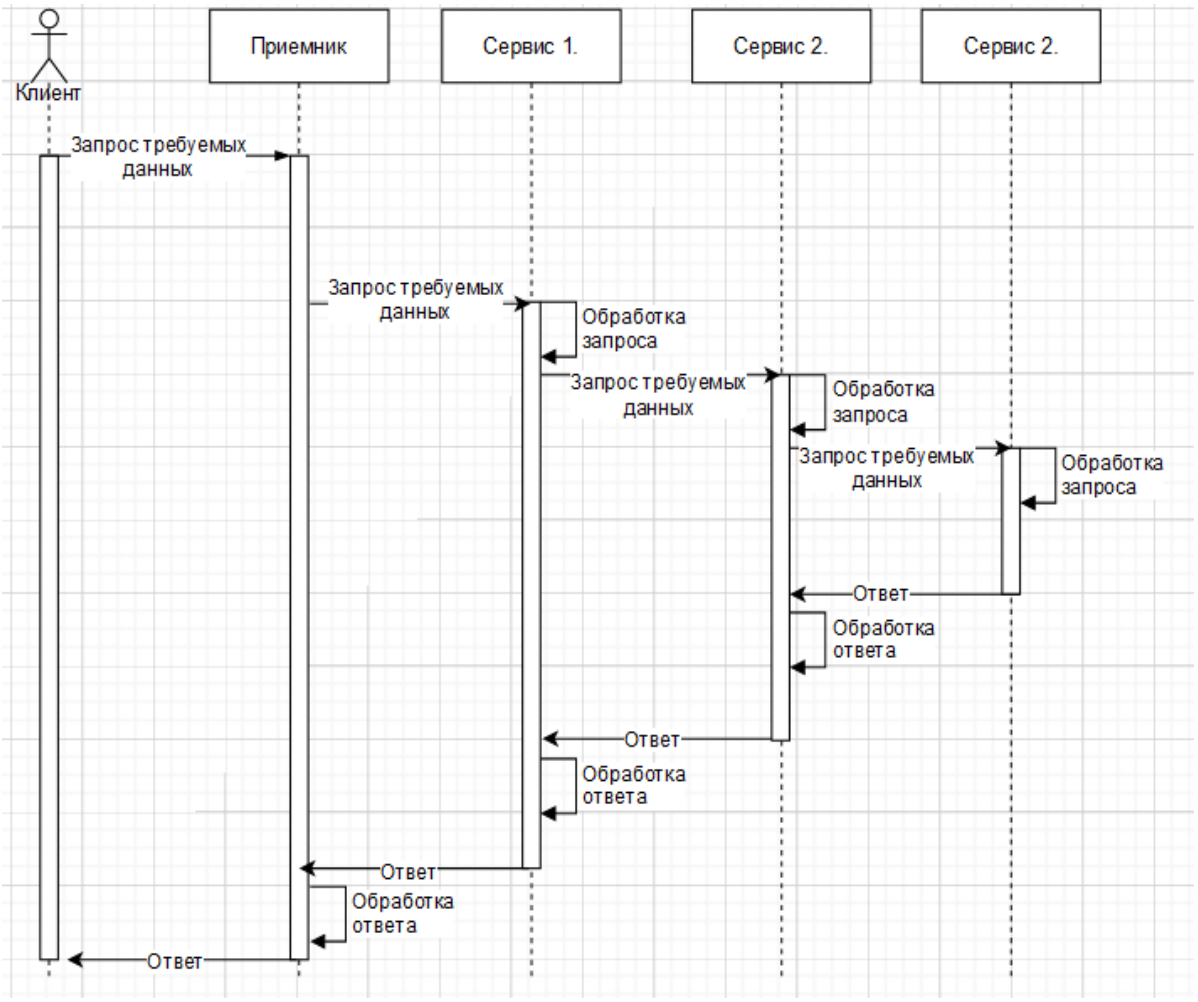


Рисунок 9 – Sequence-диаграмма цепочек последовательных вызовов

Выводы

В ходе рассмотрения популярных шаблонов системной архитектуры с точки зрения их применения их в развёрнутых экосистемах:

- сформулированы критерии оценки безопасности, масштабируемости, оптимальности, интегрируемости, сложности и адаптивности систем;
- сделан вывод, что наиболее применимой является микросервисная архитектура.

Также выявлены проблемы, возникающие при применении микросервисной архитектуры с точки зрения удовлетворения надежности, безопасности и оптимальности. Предложены шаблоны, призванные решить данные проблемы. Рассмотрены их особенности.

Литература

1. Мальчева, Р. В. Компьютерные технологии – основа цифровой экономики // Бизнес-инжиниринг сложных систем: модели, технологии, инновации. Сборник материалов III международной научно-практической конференции. – Донецк: ДОННТУ, 2018. - С. 102-105.

2. Паньшин, Б. Цифровая экономика: особенности и тенденции развития [Электронный ресурс]. – Режим доступа: https://cyberleninka.ru/article/n/tsifrovaya-ekonomika-osobennosti-i-tendentsii-razvitiya

3. Мальчева, Р. В. Проектирование архитек-туры системы электронных денег / Р. В. Мальчева, К. А. Терещенко // Информатика и кибернетика. – Донецк: ДонНТУ, 2023. - № 1(31). – С. 23-29.

4. Терещенко, К. А. Проектирование распределенной архитектуры компьютерной сети банка / К. А. Терещенко, Р. В. Мальчева // Информационное пространство Донбасса: проблемы и перспективы : материалы V Респ. С междунар. Участием науч.-практ. Конф., 27 окт. 2022 г. – Донецк : ГОУ ВПО «ДонНУЭТ», 2022. – С. 131 -134.

5. Терещенко, К. А. Анализ особенностей функционирования и развития объектов и процес-сов компьютерной сети банка / К. А. Терещенко, Р. В. Мальчева // Информатика, управляющие системы, математическое и компьютерное моделирование (ИУСМКМ-2022): Материалы XIII Международной научно-

технической конференции в рамках VIII Международного Научного форума Донецкой Народной Республики. Донецк, 2022. – Донецк: ДонНТУ, 2022. – С.392-295.

6. Service-oriented architecture [Electronic resource] – URL: htpps://learn.microsoft.com/en-us/dotnet/architecture/microservices/architect-microservice-container-applications/service-oriented-architecture

7. Миндалёв, И. В. Электронный учебно-методический комплекс для направления 09.03.03 (230700.62) «Прикладная информатика» : Раздел 1.3 Требования [Электронный ресурс]. – Режим доступа: http://enisey.name/umk/upr/ch01s03.html

8. Raj, P. Architectural Patterns: Uncover essential patterns in the most indispensable realm of enterprise architecture [Электронный ресурс] / P. Raj, A. Raman, H. Subramanian. – Packt Publishing - ebooks Account, 2017. – 468 p. Режим доступа: - htpps://ru.z-library.se/book/3493736/28a234/architectural-patterns-uncover-essential-patterns-in-the-most-indispensable-realm-of-enterprise-arc.html?dsource=recommend

9. Richards, M. Software Architecture Patterns: Understanding Common Architectural Styles and When to Use Them, 2nd Edition [Электронный ресурс]. - O'Reilly Media, Inc., 2022. – 92 p. — ISBN-13: 978-1-098-13427-3. - Режим доступа: https://vtome.ru/knigi/programming/595543-software-architecture-patterns-2nd-edition.html

10. Circuit Breaker pattern [Электронный ресурс]. – Режим доступа: - https://learn.microsoft.com/en-us/azure/architecture/patterns/circuit-breaker

11. Использование шлюзов API в микрослужбах [Электронный ресурс]. – Режим доступа: - https://learn.microsoft.com/ru-ru/azure/architecture/microservices/design/gateway

Терещенко К. А., Мальчева Р. В. Анализ специфики и области применения архитектурных шаблонов в развернутых экосистемах. *В статье рассмотрены основные шаблоны системной архитектуры. Сформулирована система оценки их применения в развёрнутых экосистемах. Проведена оценка применимости в рамках сформированной системы. Выбран наиболее применимый шаблон. Описаны дополнительные Шаблоны для повышения соответствия определенным обязательным критериям шаблона. Сформулированы критерии оценки архитектур: безопасность, масштабируемость, оптимальность, интегрируемость, сложность и адаптивность, на основании которых сделан вывод, что наиболее применимой является микросервисная архитектура.*

Ключевые слова: *экоситема, шаблон, архитектура, критерии, система оценок*

Tereshchenko K. A., Malcheva R. V. Analysis of the specifics and scope of architectural patterns in deployed ecosystems. *The article discusses the main patterns of the system architecture. A system for evaluating their application in deployed ecosystems has been formulated. The assessment of applicability within the framework of the formed system is carried out. The most applicable template has been selected. Additional patterns are described to enhance compliance with certain mandatory template criteria. The criteria for evaluating architectures are formulated: security, scalability, optimality, integrability, complexity and adaptability, on the basis of which it is concluded that the microservice architecture is the most applicable.*

Key words: *ecosystem, pattern, architecture, criteria, evaluation system*

Статья поступила в редакцию 10.12.2023
Рекомендована к публикации профессором Зори С. А.

Об авторах

Боднар Алина Валериевна - кандидат технических наук, доцент, доцент кафедры программной инженерии им. Л. П. Фельдмана факультета интеллектуальных систем и программирования ФГБОУ ВО «Донецкий национальный технический университет».

Григорьев Александр Владимирович - кандидат технических наук, доцент, доцент кафедры программной инженерии им. Л. П. Фельдмана факультета интеллектуальных систем и программирования ФГБОУ ВО «Донецкий национальный технический университет».

Завадская Татьяна Владимировна - кандидат технических наук, доцент, доцент кафедры компьютерной инженерии факультета интеллектуальных систем и программирования ФГБОУ ВО «Донецкий национальный технический университет».

Звягинцев Дмитрий Евгеньевич – магистрант группы МИДм-22, кафедры компьютерного моделирования и дизайна факультета информационных систем и технологий ФГБОУ ВО «Донецкий национальный технический университет».

Истягин Алексей Олегович - магистрант кафедры программной инженерии им. Л. П. Фельдмана факультета интеллектуальных систем и программирования ФГБОУ ВО «Донецкий национальный технический университет».

Креков Олег Игоревич - магистрант кафедры компьютерной инженерии факультета интеллектуальных систем и программирования ФГБОУ ВО «Донецкий национальный технический университет».

Криводубский Олег Александрович - доктор технических наук, доцент, профессор кафедры программной инженерии им. Л. П. Фельдмана факультета интеллектуальных систем и программирования ФГБОУ ВО «Донецкий национальный технический университет».

Кучер Татьяна Викторовна - старший преподаватель кафедры математических дисциплин, ФГКОУ ВО «Донецкий институт ГПС МЧС России».

Мальчева Раиса Викторовна - кандидат технических наук, доцент, доцент кафедры компьютерной инженерии факультета интеллектуальных систем и программирования ФГБОУ ВО «Донецкий национальный технический университет».

Ремизов Всеволод Константинович – студент кафедры программной инженерии им. Л. П. Фельдмана факультета интеллектуальных систем и программирования ФГБОУ ВО «Донецкий национальный технический университет».

Руденко Мария Павловна - кандидат технических наук, доцент кафедры компьютерного моделирования и дизайна факультета информационных систем и технологий ФГБОУ ВО «Донецкий национальный технический университет».

Рычка Ольга Валентиновна – кандидат технических наук, доцент кафедры программной инженерии им. Л. П. Фельдмана факультета интеллектуальных систем и программирования ФГБОУ ВО «Донецкий национальный технический университет».

Теплова Ольга Владимировна – аспирант ФГБНУ «Институт проблем искусственного интеллекта»

Терещенко Кирилл Александрович - аспирант кафедры компьютерной инженерии факультета интеллектуальных систем и программирования ФГБОУ ВО «Донецкий национальный технический университет».

Ходарев Даниил Федорович - магистрант кафедры программной инженерии им. Л. П. Фельдмана факультета интеллектуальных систем и программирования ФГБОУ ВО «Донецкий национальный технический университет».

**Требования к статьям,
направляемым в редакцию научного журнала
«Информатика и кибернетика»**

Редколлегией принимаются к рассмотрению статьи, в которых рассматриваются важные вопросы в области информатики и кибернетики. Научный журнал издаётся с 2015 года, периодичность издания – 4 раза в год.

В журнале предусмотрены следующие рубрики:

- информатика и вычислительная техника;
- компьютерные и информационные науки;
- инженерное образование.

В соответствии с номенклатурой специальностей научных работников МОН ДНР первые две рубрики соответствуют следующим укрупненным группам специальностей научных работников:

05.01 – «Инженерная геометрия и компьютерная графика»,

05.13 – «Информатика, вычислительная техника и управление».

С 01.02.2019 Научный журнал включён в Перечень рецензируемых научных изданий, в которых должны быть опубликованы основные научные результаты диссертаций на соискание учёной степени кандидата наук, на соискание учёной степени доктора наук (приказ МОН ДНР № 135) по группам специальностей 05.01.00 и 05.13.00.

Рубрика «Инженерное образование» предназначена опубликования сотрудниками научно-методических статей.

Журнал также включён в базу данных РИНЦ (Российский индекс научного цитирования) (лицензионный договор № 425-07/2016 от 14.07.2016).

Статьи, представляемые в данный сборник, должны отвечать следующим требованиям. ***Содержание статьи*** должно быть посвящено актуальным научным проблемам и включать следующие необходимые элементы:

- постановку проблемы в общем виде, её связь с важными научными и практическими задачами;
- анализ последних исследований и публикаций, в которых решается данная задача и на которые опирается автор, выделение нерешенных ранее частей общей проблемы, которым посвящается статья;
- формулировка цели статьи и постановка задач, решаемых в ней;
- изложение основного материала с полным обоснованием полученных научных результатов;
- выводы и перспективы последующих исследований в данном направлении.

Каждый элемент должен быть выделен соответствующим названием раздела, например, «введение», «постановка задачи», «цель и задачи работы», «цель статьи», «цель исследования», «цель разработки», «анализ … », «сравнительная оценка … », «разработка … », «проектирование … », «программная реализация», «тестирование … », «полученные результаты», «выводы», «литература». Разделы «введение», «выводы», «литература» являются обязательными. Включать в названия разделов нумерацию не разрешается.

В основном тексте статьи формулируются и обосновываются полученные авторами утверждения и результаты. Выводы должны полностью соответствовать содержанию основного текста. Языки публикаций: русский, английский.

Объём статьи, формат страницы

Для оформления статьи следует использовать листы формата А4 (210х297 мм) с полями по 2,5 см со всех сторон. Нумерацию страниц выполнять не нужно.

Рекомендуемый объём статьи – 6-12 страниц. Рукописи меньшего объёма могут быть рекомендованы к публикации в качестве коротких сообщений.

Последняя страница текста статьи должна быть заполнена не менее чем на две трети, но содержать не менее трёх пустых строк в конце.

Форматирование текста

Подготовка статьи осуществляется в текстовом редакторе Microsoft Office Word.

Весь текст статьи оформляется шрифтом Times New Roman 10 пт с одинарным междустрочным интервалом, если ниже в требованиях не сказано иного. Абзацный интервал «перед» – 0 пт, «после» – 0 пт.

На первой строке с выравниванием по левому краю располагается УДК.

Заголовок (название) статьи оформляется шрифтом Times New Roman 14 пт, полужирное начертание, с выравниванием по центру (без абзацных отступов). Заголовок статьи следует печатать с прописной буквы без точки в конце, переносы слов не допускаются. Абзацный интервал «перед» – 12 пт, «после» – 12 пт.

После названия статьи следует информация об авторах, которая выравнивается по центру (без абзацных отступов). На одной строке указываются инициалы и фамилии всех авторов через запятую. Между двумя инициалами ставится пробел. С новой строки указывается название вуза (организации) и город (для каждого автора, если не совпадают). На следующей строке указываются адреса электронной почты (один адрес либо каждого автора – по желанию). Адрес электронной почты оформляется в виде гиперссылки.

К тексту аннотации применяется курсивное начертание, с выравниванием по ширине, отступы слева и справа по 1 см. Заголовок «Аннотация» выделяется полужирным начертанием. Объём аннотации – 450-550 символов (без пробелов). Абзацный интервал «перед» – 12 пт, «после» – 12 пт.

Основной текст статьи разбивается на две колонки шириной по 7,5 см (промежуток между столбцами – 0,99 см), выравнивается по ширине. Абзацный отступ первой строки – 1 см. Автоматический перенос слов не применяется.

Заголовки разделов выполняются шрифтом Arial 10 пт, полужирное курсивное начертание. Абзацный отступ отсутствует, интервал перед абзацем – 12 пт, после абзаца – 6 пт. Для заголовка «Введение» установить интервал «перед» – 0 пт, «после» – 6 пт.

Таблицы в тексте статьи

Название следует помещать над таблицей с абзацного отступа (1 см) в формате: слово «Таблица», пробел, номер таблицы, пробел, тире, пробел, название таблицы. Название таблицы записывают с прописной буквы без точки в конце строки и выравнивают по ширине. В ячейках таблицы устанавливается выравнивание текста по центру по вертикали. По горизонтали текст выравнивается по центру либо по левому краю. Границы ячеек таблицы должны быть только чёрного цвета, толщина линии – 1 пт. На все таблицы должны быть приведены ссылки в тексте статьи, при ссылке следует писать слово «табл.» с указанием её номера, например, « … данные приведены в табл. 5». Таблицы нумеруются в пределах статьи. Таблица располагается сразу после ссылки на неё, если это возможно (например, после окончания абзаца). Если же таблица не помещается на текущей странице, то она должна быть расположена в начале следующей страницы (или колонки). При необходимости допускается включение в статью таблицы, ширина которой превышает ширину колонки. В этом случае таблица и её название размещаются по центру страницы. Таблица не должна выступать за границы полей страницы. Таблица и её название отделяются от основного текста статьи одной пустой строкой до и после.

Рисунки в статье

Ссылки на иллюстрации по тексту статьи обязательны и оформляются в виде « ... на рис. 2» и т. п. Рисунок и его подпись выравниваются по центру колонки (без абзацных отступов), положение рисунка – «в тексте». Размещается рисунок после его первого упоминания в тексте, если это возможно (например, после окончания абзаца). Если же иллюстрация не помещается на текущей странице, то она должна быть расположена в начале следующей страницы (или колонки). При необходимости допускается включение в статью рисунка, ширина которого превышает ширину колонки. В этом случае рисунок и его подпись выравниваются по центру страницы. Иллюстрация не должна выступать за границы полей страницы. Подпись рисунка оформляется в формате: слово «Рисунок», пробел, номер иллюстрации, пробел, тире, пробел, название рисунка. Название рисунка записывают с прописной буквы без точки в конце строки. Для подписи иллюстрации применяют курсивное

начертание. Иллюстрация и её подпись отделяются от основного текста статьи одной пустой строкой до и после. Не допускается выполнять рисунки с помощью встроенного графического редактора Microsoft Office Word. Если на иллюстрации имеется текст, размер шрифта должен быть не менее чем аналогичный текст, набранный шрифтом Times New Roman 10-го размера. Иллюстрация не должна содержать много незаполненного пространства.

Формулы

Формулы и уравнения рекомендуется набирать с использованием MathType (предпочтительно) или MS Equation. Формулы и математические символы не должны существенно отличаться по размеру от основного текста. Обязательной является нумерация формул, на которые имеется ссылка в тексте статьи. Ссылки в тексте на порядковые номера формул дают в скобках, например, « ... согласно формуле (2)». Формулы размещаются по центру колонки, а их номера – по правому краю. Как для строки с формулой, так и для первой строки пояснений (при наличии), абзацный отступ убирается. Первая строка пояснения начинается со слова «где», после которого следует поставить табуляцию на 1 см, затем само пояснение в формате: символ, подлежащий объяснению, пробел, тире, пробел, поясняющий текст, запятая, обозначение единицы измерения физической величины. Пояснения перечисляются через точку с запятой, выравниваются по ширине. Вторая и последующие строки пояснений начинаются с абзацного отступа (1 см). Весь блок текста, связанный с формулой (только формула, несколько формул подряд или формула с пояснениями), отделяется от основного текста одной пустой строкой до и после. Переносить формулы на следующую строку допускается только на знаках выполняемых операций, причем знак в начале следующей строки повторяют. При переносе формулы на знаке умножения применяют знак «×». Формулы и математические уравнений могут быть записаны в тексте документа, если их высота не превышает высоту строки. При этом следует учитывать, что знаки математических операций отделяются от чисел или символов пробелами с обеих сторон. Например, «Если учесть, что $y < 0$ и $2x + y = 1$, то из формулы (3) можно выразить x…». К символам, которые приведены в формуле, при дальнейшем их употреблении (в том числе в пояснениях к формуле) должно применяться курсивное начертание. При этом к любым числам (верхние и нижние индексы, содержащие цифры и т. п.), а также к математическим знакам курсивное начертание не применяется. Не допускается вставлять формулы, выполненные в виде рисунков.

Перечисления: оформление списков

Основной текст статьи может содержать перечисления, оформленные в виде маркированного списка. В качестве маркера элемента списка разрешается использовать только короткое тире «–». Каждый элемент перечисления записывается с новой строки с абзацного отступа, равного 1 см. После символа короткого тире текст располагается с отступом в 1,5 см от левой границы строки, выравнивается по ширине, при переносе на новые строки располагается без отступов. Нумерованные и многоуровневые списки включать в статью не разрешается.

Литература

В тексте статьи обязательны ссылки на все литературные источники, номер источника указывается в квадратных скобках. Ссылки на неопубликованные работы не допускаются. Рекомендуемое количество источников, на которые ссылается автор, не менее 10. Перечень источников приводится в порядке их упоминания в статье. Библиографическое описание каждого литературного источника оформляется в соответствии с ГОСТ Р 7.0.100–2018. Перечень литературных источников оформляется в виде нумерованного списка. В качестве маркеров элементов списка используют порядковые арабские цифры с точкой. Каждый источник представляет собой отдельный элемент перечисления, записывается с новой строки с абзацного отступа, равного 1 см. После порядкового номера с точкой текст располагается с отступом в 1,5 см от левой границы строки, выравнивается по ширине, при переносе на новые строки располагается без отступов.

В конце статьи обязательно приводятся аннотации на русском и английском языках, каждая заканчивается перечнем 5-6 ключевых слов.

К тексту аннотации применяется курсивное начертание, с выравниванием по ширине, отступы слева и справа по 1 см. Слово «Аннотация» опускается. Текст аннотации начинается с ФИО авторов и названия статьи, выделяемых полужирным начертанием. Аннотация на русском языке совпадает с аннотацией, приведенной в начале статьи. В тексте аннотации на английском языке после фамилии автора указывается только первая буква имени с точкой. Абзацный интервал «перед» – 12 пт, «после» – 12 пт. Ключевые слова оформляются с новой строки аналогично тексту аннотации. Заголовок «Ключевые слова:» (англ. «Keywords:») выделяется полужирным начертанием. Ключевые слова перечисляются через запятую.

Порядок представления статьи и сопроводительные документы

В редакцию необходимо представить:

– файл с текстом статьи;

– файл, содержащий фамилию, имя и отчество авторов полностью; ученую степень, ученое звание; место работы с полным указанием должности, подразделения и наименования организации, города (страны); номера телефонов и e-mail для связи;

– экспертное заключение о возможности публикации статьи, подписанное руководителем и заверенное печатью организации, в которой работает автор статьи;

– выписка из заседания кафедры или письмо организации с просьбой об опубликовании и указанием, что изложенные в статье результаты ранее не публиковались.

Статьи и сопроводительные документы следует высылать на электронный адрес infcyb.donntu@yandex.ru.

К сведению авторов

Если статья оформлена с нарушением указанных выше требований и правил, редакция после предварительного рассмотрения может отклонить статью.

На рецензирование статьи направляются членам редакционной коллегии журнала. Все статьи публикуются при наличии положительной рецензии.

В статью могут быть внесены изменения редакционного характера без согласования с автором. Ответственность за содержание статьи и качество перевода аннотаций несут авторы.

Публикация статей в научном журнале «Информатика и кибернетика» осуществляется на некоммерческой основе.

Все номера Научного журнала размещаются в электронной библиотечной системе ФГБОУ ВО «ДонНТУ» и на сайте http://infcyb.donntu.ru/.

.

CONTENT

Informatics and computer engineering

Электронное периодическое издание

Научный журнал

ИНФОРМАТИКА И КИБЕРНЕТИКА

(на русском, английском языках)

№ 4 (34) - 2023

Ответственный за выпуск Р. В. Мальчева

Технический редактор Р. В. Мальчева

Компьютерная верстка Р. В. Мальчева

Подписано к выпуску 12.12.2023. Усл. печ. лист. 7,5. Уч.-изд. лист.4,3.
Адрес редакции: ДНР, 283001, г. Донецк, ул. Артема, 58, ФГБОУ ВО «ДонНТУ»,
4-й учебный корпус, к. 36, ул. Кобозева, 17.
Тел.: +7 (856) 301-07-35, +7 (949) 334-89-11
E-mail: infcyb.donntu@yandex.ru, URL: http://infcyb.donntu.ru