

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

Факультет
вычислительной техники

Кафедра
МОиПЭВМ

Направление подготовки 09.03.02 «Информационные системы и технологии»

БАКАЛАВРСКАЯ РАБОТА

на тему

КОМПЬЮТЕРНАЯ МОДЕЛЬ ФУНКЦИОНИРОВАНИЯ
ИНФОРМАЦИОННО-УПРАВЛЯЮЩЕЙ СИСТЕМЫ
МОБИЛЬНОГО ВИДЕОНАБЛЮДЕНИЯ

Студент _____ Кудикова Д.В. _____
(подпись, дата) (ФИО полностью)

Руководитель _____ Козлов А.Ю. _____
(подпись, дата) (фамилия, инициалы)

Нормоконтролёр _____ Такташкин Д.В. _____
(подпись, дата) (фамилия, инициалы)

Работа допущена к защите (протокол заседания кафедры от _____ № _____)

Заведующий кафедрой _____ Козлов А.Ю. _____
(подпись) (фамилия, инициалы)

Работа защищена с отметкой _____ (протокол заседания ГЭК от _____ № _____)

Секретарь ГЭК _____ Попова Н.А. _____
(подпись) (фамилия, инициалы)

Пенза, 2023

Федеральное государственное бюджетное образовательное учреждение
высшего образования
ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

Факультет

Вычислительной техники

Кафедра

МО и ПЭВМ

«Утверждаю»
Заведующий кафедрой
_____ А.Ю. Козлов
«__» _____ 2023 г

**ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
БАКАЛАВРА**

1. Студент Кудикова Даяна Владимировна гр. 19ВИ1 факультета ВТ
<Фамилия, имя, отчество полностью>

2. Тема работы Компьютерная модель функционирования информационно-
управляющей системы мобильного видеонаблюдения

Тема утверждена приказом ПГУ № _____ от " _____ " _____ 20 _____

2. Руководитель работы профессор МОиПЭВМ д.т.н. Козлов А.Ю.

3. Задание на работу (назначение разработки, исходные данные и т.п.)

Компьютерная модель функционирования информационно-управляющей системы
мобильного видеонаблюдения должна включать математическую модель и программное
обеспечение, позволяющее решать задачу определения вероятностных параметров
функционирования информационно-управляющей системы мобильного
видеонаблюдения; математическая модель должна строиться на основе использования
положений теории полумарковских процессов; программное обеспечение компьютерной
модели должно быть разработано на языке программирования C++; программное
обеспечение должно функционировать под управлением ОС Windows 10.0.

4. Перечень подлежащих разработке вопросов

1. Характеристика информационно-управляющей системы мобильного видеонаблюдения.

1.1. Применение мобильного наблюдения. 1.2. Характеристика функционирования

информационно-управляющей системы мобильного видеонаблюдения. 1.3 Постановка

задачи исследования. 2. Разработка математической модели информационно-управляющей

системы мобильного видеонаблюдения. 2.1. Схема построения математической модели функционирования информационно-управляющей системы мобильного видеонаблюдения. 2.2. Моделирование функционирования информационно-управляющей системы мобильного видеонаблюдения на основе теории полумарковских процессов. 3. Программное обеспечение компьютерной модели информационно-управляющей системы мобильного видеонаблюдения. 3.1. Процесс проектирования программных средств и структур данных для задачи определения вероятностных параметров функционирования информационно-управляющей системы мобильного видеонаблюдения. 3.2. Программное обеспечение решения задачи определения вероятностных параметров функционирования информационно-управляющей системы мобильного видеонаблюдения. 3.3. Валидация разработанной компьютерной модели. 3.4. Расчет бюджета на разрабатываемый продукт

5. Календарный график выполнения работы

№ п/п	Наименование этапов работы	Объем работы	Срок выполнения	Подпись руководителя
1	Анализ предметной области и требований к системе	10%	12.04.2023 – 16.04.2023	
2	Проектирование программных средств	20%	17.04.2023 – 26.04.2023	
3	Реализация программных средств	30%	27.04.2023 – 11.05.2023	
4	Тестирование программных средств	20%	12.05.2023 – 21.05.2023	
5	Оформление пояснительной записки	20%	22.05.2023 – 01.06.2023	

Дата выдачи задания «22» декабря 2023г.

Руководитель бакалаврской работы _____
(подпись, дата)

Задание к исполнению принял студент _____
(подпись, дата)

Работу к защите допустить
Декан ФВТ _____ Л.Р. Фионова
(подпись, дата)

Реферат

Пояснительная записка содержит 62 страниц, 17 рисунков, 20 использованных источника и 1 приложение.

СИСТЕМА МОБИЛЬНОГО ВИДЕОНАБЛЮДЕНИЯ,
МАТЕМАТИЧЕСКАЯ МОДЕЛЬ, ВЕРОЯТНОСТНЫЕ ПАРАМЕТРЫ, С

Объект разработки: система видеонаблюдения.

Цель работы: разработка приложения и требований к вероятным временным параметрам функционирования мобильных систем видеонаблюдения при проектировании, которые можно решить только на основе математического моделирования.

Результаты работы: разработано приложение, которое определяет к вероятным временным параметрам функционирования мобильных систем видеонаблюдения при проектировании, которые можно решить только на основе математического моделирования.

Средства разработки: C++.

					ПГУ 09.03.02 – 8ВР191.09 ПЗ			
Изм.	Лист	№ докум.	Подпись	Дата				
Разраб.		Кудикова Д.В			Компьютерная модель функционирования информационно-управляющей системы мобильного видеонаблюдения Пояснительная записка	Лит.	Лист	Листов
Провер.		Козлов А.Ю.					4	62
Реценз.						группа 19ВИ1		
Н. Контр.		Такташкин Д.В						
Утверд.		Козлов А.Ю.						

Содержание

Введение.....	2
1. Характеристика информационно-управляющей системы мобильного видеонаблюдения	3
1.1. Применение мобильного видеонаблюдения	3
1.2. Характеристика функционирования информационно-управляющей системы мобильного видеонаблюдения	3
2. Математическая модель функционирования информационно-управляющей системы мобильного видеонаблюдения.....	14
2.1. Схема построения математической модели функционирования информационно-управляющей системы видеонаблюдения.....	14
2.2. Моделирование функционирования информационно-управляющей системы мобильного видеонаблюдения	20
3. Программное обеспечение компьютерной модели информационно-управляющей системы мобильного видеонаблюдения.....	27
3.1. Процесс проектирования программных средств и структур данных для задачи определения вероятностных параметров функционирования информационно-управляющей системы мобильного видеонаблюдения	27
3.2. Программное обеспечение для решения вероятностных параметров функционирования информационно-управляющей системы мобильного видеонаблюдения	29
3.3. Валидация разработанной компьютерной модели.....	32
3.4. Расчет бюджета на разрабатываемый продукт	36
Заключение.....	39
Список используемых источников	40
Приложение А. Листинг программы.....	43

Введение

В современном мире видеонаблюдение стало неотъемлемой частью комплексных системы безопасности, так как современные системы видеонаблюдения имеют возможность не только показывать и записывать видео, но и программировать реакцию всей системы безопасности в случае возникновения непредвиденных ситуаций. Видеонаблюдение (Closed Circuit Television, CCTV – система телевидения замкнутого периметра) – это процесс визуального контроля (наблюдения) за объектом и событиями, происходящими на его территории осуществляется с помощью специализированного оборудования объединенного в определенную систему. Объектом видеонаблюдения является контролируемая системой видеонаблюдения территория, в том числе помещения зданий и сооружений, прилегающая территория и др.

Технология видеонаблюдения не стоит на месте, а наоборот развивается и стремительно. Разрабатываются новые виды камер с различными функциями: поворотная камера (PTZ), передача информации по движению, делая снимок (GSM) и другие.

В случаях, когда нужно организовать скрытое видеонаблюдение на территории используется мобильное видеонаблюдение.

Компьютерная модель – это описание математической модели и создание программного обеспечения.

Цель и задачи исследования: разработка приложения и требований к вероятным временным параметрам функционирования мобильных систем видеонаблюдения при проектировании, которые можно решить только на основе математического моделирования.

Объект и предмет исследования: объектом исследования является система видеонаблюдения. Предметом исследования – модель функционирования.

1. Характеристика информационно-управляющей системы мобильного видеонаблюдения

1.1. Применение мобильного видеонаблюдения

Самая главная из всех причин, по которой используют системы видеонаблюдения – это стремление повысить уровень безопасности. Можно сказать, что видеонаблюдение продемонстрировало большой прогресс в обеспечении безопасности. Благодаря системам видеонаблюдения можно обеспечить безопасность не только на территории объекта. Большие предприятия нуждаются в надежных и контролируемых средствах контроля, и управления технологическими процессами. Люди также нуждаются в большом количестве автоматизированных средств для осуществления своего труда. Системы видеонаблюдения дают возможность контролировать и управлять большим количеством технологических операций, которые не контролируются напрямую человеком.

1.2. Характеристика функционирования информационно-управляющей системы мобильного видеонаблюдения

Вся современная система видеонаблюдения включает в себя не только телевизор и видеокамеру для просмотра передаваемого с нее изображения. Система видеонаблюдения на данный момент представляет из себя целую сеть элементов, которые связываются друг с другом при помощи передачи сигнала посредством радиоволн или кабеля. Какие именно приборы входят в систему видеонаблюдения, определяют разные типы систем.

С конца 60-х годов прошлого века стали широко применяться аналоговые системы видеонаблюдения (CCTV, closed circuit television). Камера, проводная сеть для передачи видеосигнала и монитор – вот четыре компонента системы видеонаблюдения (рисунок 1).

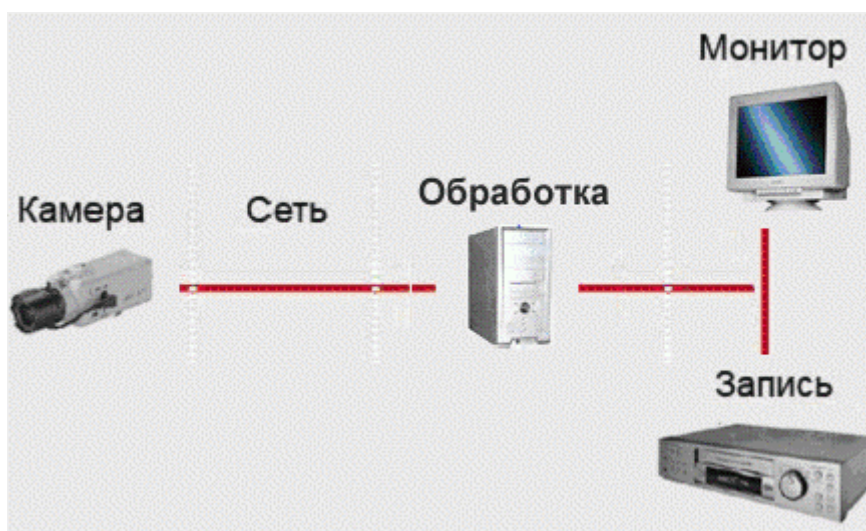


Рисунок 1 – Общая схема системы видеонаблюдения

В 1990 году был переход на цифровые технологии и аналоговые телекамеры с ПЗС-матрицей (предметами для зарядки). Такие камеры стали вытеснять аналоговые телекамеры из видеонаблюдения. Цифровые видеорегистраторы (DVR – digital video recorder) имели вход для подключения коаксиального кабеля и выход, аналогичный выходу монитора. Именно эти ПЗС-телекамеры были частично цифровыми: у них был аналоговый выход для подключения монитора. Сигнал с камер продолжали записываться на аналоговых кассетных видеомagnetофонах. Это период времени был началом эры цифрового видеонаблюдения.

Системы, построенные по новому принципу (гибридному) появились во второй половине 1990-х годов вместе с появлением видеорегистраторов нового типа. Смысл состоял в том, что видеосигнал имел цифровую форму на отдельных участках межэлементарной системы. С введением в цифровых видеорегистраторах интерфейса для TCP/IP (Transmission Control Protocol / Internet protocols) – соединения стала возможна работа систем видеонаблюдения совместно с локальными сетями. Но была возможность работать совместно с локальной сетью в них было всего лишь вспомогательной опцией, никак не влияющей на принцип построения системы. Таким образом, уже в 2005 году начали выпускаться системы

видеонаблюдения нового типа – основанные на новейших цифровых технологиях. Цифровой видеосигнал передается не только между элементами системы, но и за их пределами. Данные системы имеют название «IP-видеонаблюдение». Состав таких систем включает в себя: сетевые камеры или IP-камеры, программно – аппаратные комплексы для анализа видеофрагментов и обработки сигнала. IP-видеонаблюдение отличается от гибридных и аналоговых систем возможностью использования разных сетевых устройств – свичей, роутеров. Рассмотрим детальнее о видах видеокамер видеонаблюдения.

Аналоговое видеонаблюдение.

Для систем видеонаблюдения до сих пор выпускаются большими производителями аналоговые устройства. Хотя с появлением первых IP камер, казалось что в скором времени все видеосистемы будут переключены на цифровые камеры и устройство аналогового типа перестанет быть популярным. Несмотря на это, аналоговые системы смогли сохранить свою популярность. Есть несколько основных причин, по которым осуществляется монтаж аналогового видеонаблюдения на ряде объектов. Это следующее:

- стоимость аналоговых устройств на 20-50% ниже стоимости IP камер, имеющих такие же характеристики и аналогичную функциональность;
- сочетание передовых технологий, функциональных и конструктивных решений аналоговых видеоустройств позволяют сочетать друг с другом изделия от различных производителей;
- аналоговое видеооборудование, которое используется в видеосистемах не изнашивается морально и физически;
- у аналоговых устройств более высокий уровень по сравнению с IP камерами – такие технические характеристики как светочувствительность, автоматическая настройка и т. д.;
- роботизированная камера может легко сопровождать движущийся объект, но в аналоговой системе это почти невозможно из-за того что более высокое время отклика видеосистемы;

— в случаи, когда оператор нуждается в быстром реагировании на постоянно меняющуюся ситуацию необходимо использовать аналоговые камеры.

Комплект аналогово видеонаблюдения включает в себя видеокамеру, видеорегистратор, монитор или телевизор, вспомогательные устройства. Аналоговое видеонаблюдение включает в себя камеры, которые по своим техническим характеристикам делятся на цветные и черно-белые, с корпусом или без корпуса, оснащены ИК подсветкой либо нет. Основные функции данных камер предоставляют возможность надежно контролировать территорию. Именно по этой причине спрос на них продолжает возрастать и является достаточно высоким (рисунок 2).



Рисунок 2 – Схема системы аналогового видеонаблюдения

Принцип действия: вначале свет проходит через объектив и попадает на линзы. С помощью матриц свет преобразовывается в видеосигнал, и именно на этом этапе его путь завершается. Витая пара или коаксиальный кабель служат для передачи сигнала от видеорегистратора к монитору. В

определенных условиях возможно кодирование аналогового сигнала в цифровой при помощи специального оборудования (рисунок 3).

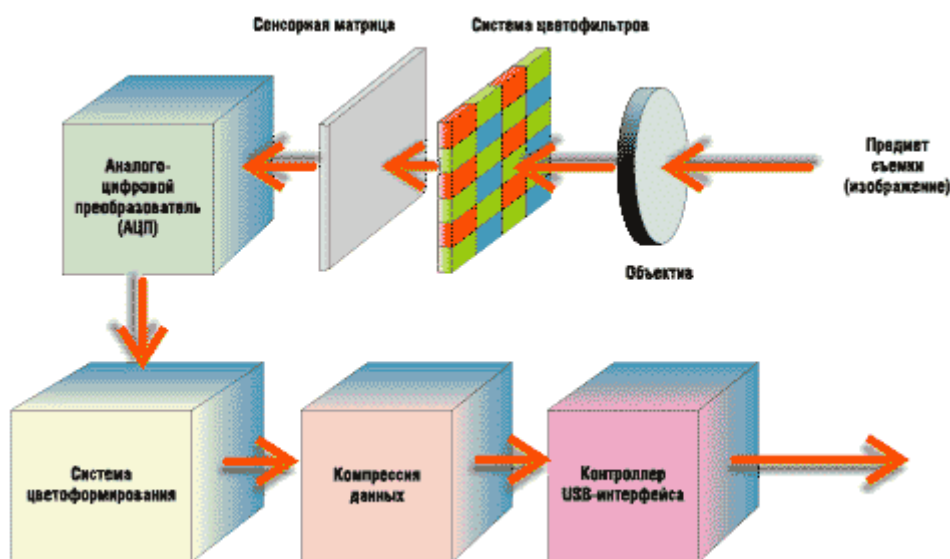


Рисунок 3 – Принцип действия

Плата видеозахвата применяется для вывода информации от регистратора на монитор. Очень часто оборудование состоит из нескольких аналоговых мониторов или телевизоров. Вывод изображения на все устройства можно осуществить при помощи коммутатора, установленного дополнительно. Коммутирующее оборудование чаще всего устанавливают для больших объектов с большим количеством помещений. Также с помощью аналоговых систем видеонаблюдения и контроля можно решать такие задачи:

- осуществление визуального контроля над событиями, которые происходят на объекте и его окрестностях. Приведенная на экран информация информирует о текущем положении дел. Это поможет вовремя отреагировать и принять быстрые решения в той или иной нестандартной ситуации;

- непрерывная видеозапись полученной информации на видеорегистраторе или компьютере. Данная функция дает возможность документально подтвердить все нарушения;

— у видеокамер и датчиков движения есть общий источник сбора информации – оператор, через них поступает сигнал о всех перемещениях по объекту.

Цифровое видеонаблюдение.

Обзор основных функций и технических характеристик аналоговых систем рекомендуется изучать в сравнении с цифровым видеонаблюдением. Это оборудование появилось несколько позже, чем аналоговое. Оно сразу стало популярно и быстро достигло своего пика. Цифровое видеонаблюдение стоило очень дорого еще тогда, когда оно только появилось на рынке. Несмотря на это уже в то время эти системы были более качественными и функциональными. В первую очередь, потребители чаще всего использовали проверенный и недорогой аналоговый вариант (рисунок 4).



Рисунок 4 – Схема цифрового видеонаблюдения

IP- видеонаблюдение.

Однако, с развитием технологий возможности цифрового оборудования возросли, а его стоимость значительно снизилась вместе со стоимостью аналоговых систем. Качество современных видеокамер нельзя сравнить даже с их предшественниками, имеющими уже достаточно большой срок

эксплуатации. По своему устройству они также имеют существенные отличия. При работе IP-камер задействуется совершенно другая схема. На матрицу, обладающую высокой светочувствительностью попадает световой поток при прохождении через объектив и линзы. Она позволяет преобразовать световой сигнал в электрический. Вслед за обработкой сигнал идет по витой паре или через Wi-Fi на мультиплексор и далее – уже непосредственно в видеорегистратор. Данная информация записывается на внешний жесткий диск и транслируется в монитор (рисунок 5).



Рисунок 5 – Схема IP видеонаблюдения

Главные достоинства:

- с помощью удаленного визуального контроля (с компьютера, телефона и т. п.), также возможно управление системой видеонаблюдения;
- небольшие затраты на установку видеонаблюдения и простота в ее выполнении. Чаще всего такие системы не требуют прокладки дополнительного кабеля;
- максимальное разрешение изображения. В основе IP-систем лежит видеоформат MPEG4;

- также, возможно создание распределенных систем видеомониторинга на достаточно большой площади;

- IP видеосистемы выбирают, когда нужно большое разрешение изображения и удаленный контроль над системой;

- масштабируемость системы является основным показателем. Число видеокамер колеблется между одной и несколькими тысячами;

- многопользовательский доступ к одной видеокамере через WEB браузер;

- надежность – это самое главное. Отсутствует ключевая часть, которая отвечает за работу всей системы в целом;

- возможность передачи данных по радиоканалу, используя технологию беспроводной связи WiFi.

Недостатки:

- завышенная цена объясняется;

- относительно невысокая удаленность: длина построения ЛВС на скорости 100/1000 Мбит / с более 100 м;

- за сложность системы, отвечающие настройками в IP технологиях и передача данных от провайдера требуют определенных навыков;

- налаженные ЛВС на объекте не смогут вынести большие объемы данных, передаваемых IP камерами;

- несокрушимая защита от хакерских атак. Злоумышленник-хакер со стремянкой и ноутбуком получает доступ к информационной сети, не проходя при этом на объект;

- задержки в отображении могут достигать до 1 и 2 секунды;

- несоответствие стандартам и несовместимость компонентов.

Беспроводное видеонаблюдение.

На российский рынок продвигается беспроводное видеонаблюдение. Люди всё чаще выбирают беспроводное видеонаблюдение. При монтаже системы видеонаблюдения стандартным проводным способом возможны

некоторые технические неполадки, связанные с монтажом кабеля для подключения. Схема работы беспроводных систем видеонаблюдения представлена на рисунке 6.



Рисунок 6 – Схема беспроводных систем видеонаблюдения

Недостатки беспроводного видеонаблюдения: если проводить монтаж системы видеонаблюдения, используя стандартный кабель (Ethernet), то можно получить: высококачественное изображение с высокой четкостью и скоростью 25 к/сек., звуковое сопровождение. Так же видеокамеру возможно подключить напрямую через локальную сеть LAN. Плюсы беспроводного видеонаблюдения утрачены, или они полностью отсутствуют. Чтобы обеспечить беспроводное видеонаблюдение, необходим источник питания. В случае если нужна постоянная замена или зарядка аккумуляторов, то подходят выносные и сетевые блоки питания. Системы видеонаблюдения, поддерживающие беспроводную сеть не могут защититься от перепадов напряжения. Первое, это то что беспроводное видеонаблюдение обладает двумя своими важными достоинствами. Если есть желание сделать систему видеонаблюдения, не придется сверлить стены для того чтобы проложить линии. В-третьих, беспроводное видеонаблюдение является мобильной

системой. В случае необходимости вы имеете возможность менять местонахождение видеокамеры в произвольном месте. Так же современные системы имеют миниатюрные размеры, а их работа может быть осуществлена с приемником оснащенным ЖК дисплеем. Основное применение – бытовое. Но существует и беспроводная система видеонаблюдения на улице. Для ночной съемки видеокамеры имеют специальный кожух и инфракрасные светодиоды, которые автоматически включаются при недостаточном освещении. Стоит отметить, что монтаж системы видеонаблюдения оправдан в случаях отсутствия технических трудностей при прокладке кабеля.

Для того что бы описать бизнес-процессы для системы мобильного видеонаблюдения используется IDEF0 (рисунок 7). Это особый графический язык который позволяет подробно и понятно описывать работу по выполнению разных функций между различными подразделениями компании. Именно бизнес-процессы описываются данным образом.

Контекстная диаграмма является наиболее абстрактным уровнем описания системы, поэтому при создании процесса моделирования в IDEF0 необходимо создать ее.

Понятие субъекта является синонимом понятия системы. Здесь можно четко определить, что входит в систему и присутствует за ее пределами (как компоненты или внешнее воздействие).

К контекстной диаграмме в пояснительном тексте нужно указывать цель (Purpose) построения и фиксировать точку зрения(Viewpoint).

Цель устанавливает конкретные области в исследуемой системе, на которых следует фокусироваться первостепенно.

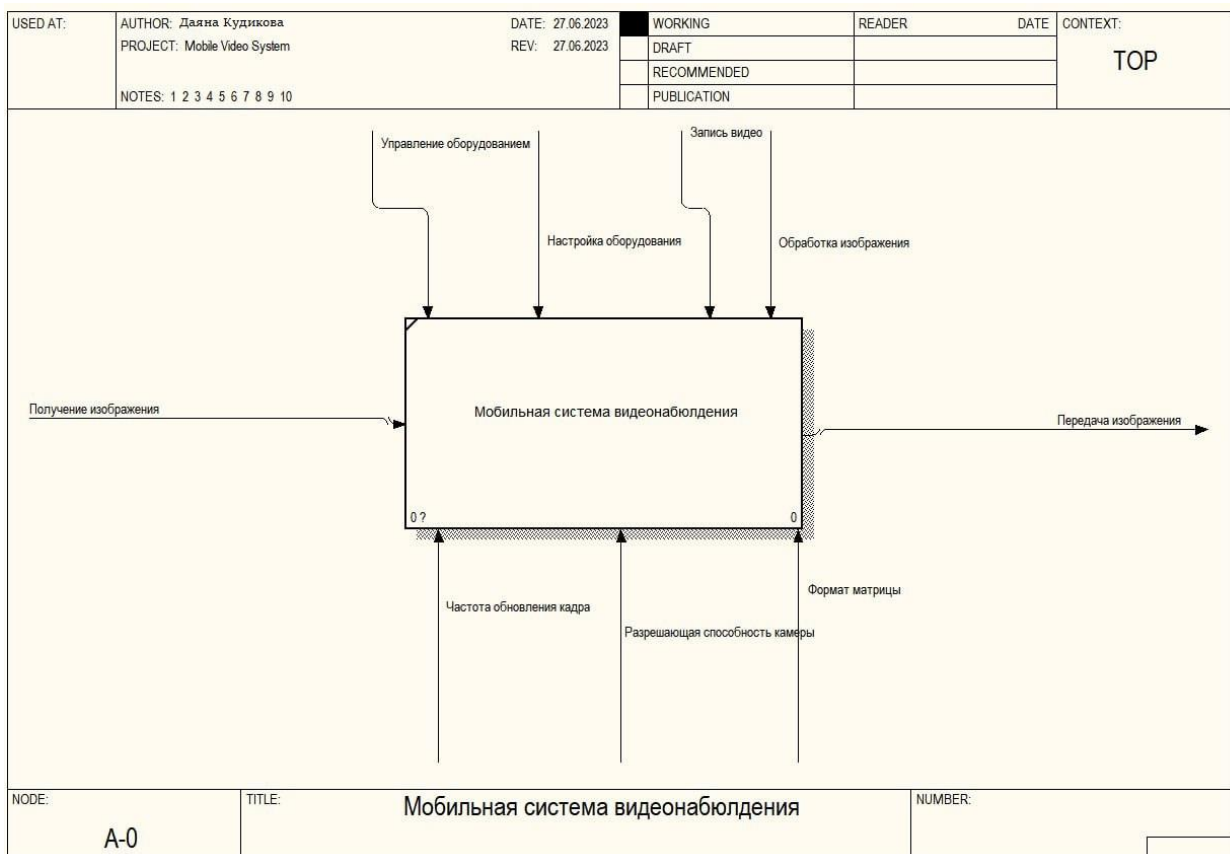


Рисунок 7 – Диаграмма IDEF0

На рисунке 7 более понятным языком изображена мобильная система видеонаблюдения

1.2. Постановка задачи исследования

Нужно разработать приложение и требования к вероятным временным параметрам функционирования мобильных систем видеонаблюдения при проектировании, которые можно решить только на основе математического моделирования.

2. Математическая модель функционирования информационно-управляющей системы мобильного видеонаблюдения

2.1. Схема построения математической модели функционирования информационно-управляющей системы видеонаблюдения

Для решения задач исследования функционирования МСВ в переходном и установившемся режиме наиболее адекватным является такое представление ПМП, когда фазовый портрет исследуемого процесса задан графом состояний $G(P, Q)$ (рисунок 8), возможными переходами $\{i, j\}$, матрицей независимых функций распределения времени пребывания МСВ в i -м состоянии перед переходом в j -е состояние $Q(t) = \|Q_{ij}(t)\|$, т. е. таких функций, которые имели бы место, если бы данный переход из состояния i был единственным, начальным состоянием процесса в момент времени $t=0$.

Состояния МСВ могут быть следующими:

S_1 – МСВ занимает указанное место и разворачивается для ведения наблюдения;

S_2 – МСВ ведет наблюдение и обрабатывает получаемую информацию;

S_3 – МСВ свертывается и производит смену места наблюдения;

S_4 – МСВ обнаружена и находится под негативным воздействием;

S_5 – МСВ неисправна и ремонтируется.

Таким образом, число состояний K рассматриваемой МСВ равно 5.

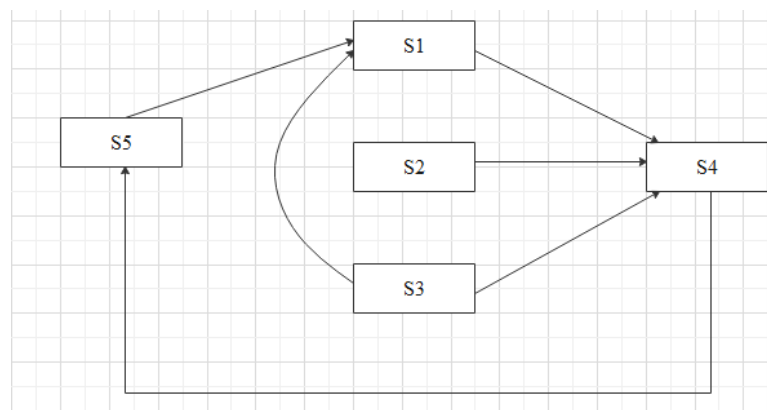


Рисунок 8 – Граф состояний МСВ

Для того чтобы определить вероятностные значения мобильного видеонаблюдения, следует задать функции независимые $Q_{ij}(t)$. Статический вывод аналитического вида независимых функций распределения возможно совершить на основе выдвижения и проверки статических гипотез о видах законов распределений.

Научные основы исследования были положены на основе теории полумарковских процессов. Он базируется из случайного процесса с конечным или численным множеством состояний, который имеет ступенчатые траектории со скачками во времени.

Статистические характеристики мобильной системы видеонаблюдения в переходном и устоявшемся режимах были получены на основе решения задачи оптимизации методом квадратичного программирования с программной реализацией MatLab.

Для рассматриваемого мобильного видеонаблюдения получим матрицу функций $Q_{ij}(t)$:

$$\mathbf{Q}(t) = \begin{pmatrix} 0 & Q_{12}(t) & 0 & Q_{14}(t) & 0 \\ 0 & 0 & Q_{23}(t) & Q_{24}(t) & 0 \\ Q_{31}(t) & 0 & 0 & Q_{34}(t) & 0 \\ 0 & 0 & 0 & 0 & Q_{45}(t) \\ Q_{51}(t) & 0 & 0 & 0 & 0 \end{pmatrix}. \quad (1)$$

Тогда при наличии смежных состояний вероятность перехода $p_{ij}(t)$ из состояния i в состояние j вычисляется по формулам:

$$p_{ij}(t) = \int_0^t \prod_{k \neq j} (1 - Q_{ik}(t)) dQ_{ij}(t), \quad (2)$$

где $\prod_{k \neq j} (1 - Q_{ik}(t))$ – вероятность невыхода из i -го состояния за время t по направлению $k \neq j$; $dQ_{ij}(t)$ – вероятность перехода по направлению j в окрестности t .

Для представленного графа матрица нестационарных вероятностей перехода будет иметь вид:

$$\mathbf{P}(t) = \begin{pmatrix} 0 & p_{12}(t) & 0 & p_{14}(t) & 0 \\ 0 & 0 & p_{23}(t) & p_{24}(t) & 0 \\ p_{31}(t) & 0 & 0 & p_{34}(t) & 0 \\ 0 & 0 & 0 & 0 & p_{45}(t) \\ p_{51}(t) & 0 & 0 & 0 & 0 \end{pmatrix}. \quad (3)$$

Установившееся (стационарное) значение переходной вероятности p_{ij} получим из выражения:

$$p_{ij} = p_{ij}(t \rightarrow \infty) = \lim_{t \rightarrow \infty} p_{ij}(t) = \int_0^{\infty} p_{ij}(t) dt = \int_0^{\infty} \prod_{k \neq j} (1 - Q_{ik}(t)) dQ_{ij}(t). \quad (4)$$

Полученная матрица стационарных значений переходных вероятностей имеет вид:

$$\mathbf{P} = \begin{pmatrix} 0 & p_{12} & 0 & p_{14} & 0 \\ 0 & 0 & p_{23} & p_{24} & 0 \\ p_{31} & 0 & 0 & p_{34} & 0 \\ 0 & 0 & 0 & 0 & p_{45} \\ p_{51} & 0 & 0 & 0 & 0 \end{pmatrix}. \quad (5)$$

Вероятность (2) есть вероятность сложного события: p_{ij} - перехода в состояние j и $F_{ij}(t)$ - пребывания в i -м состоянии в течение времени t , т. е. $p_{ij}(t) = p_{ij} F_{ij}(t)$, откуда можно определить условную функцию распределения времени ожидания перехода

$$F_{ij}(t) = \frac{p_{ij}(t)}{p_{ij}}. \quad (6)$$

В итоге получим следующую матрицу:

$$\mathbf{F}(t) = \begin{pmatrix} 0 & F_{12}(t) & 0 & F_{14}(t) & 0 \\ 0 & 0 & F_{23}(t) & F_{24}(t) & 0 \\ F_{31}(t) & 0 & 0 & F_{34}(t) & 0 \\ 0 & 0 & 0 & 0 & F_{45}(t) \\ F_{51}(t) & 0 & 0 & 0 & 0 \end{pmatrix}. \quad (7)$$

Следовательно, начальная матрица независимых функций распределения времени ожидания переходов $Q(t)$ трансформируется в две ячейки: переходную ячейку $P = \|dQ_{ij}(t)\|$, $i, j = 1, \dots, K$, и часть условных функционалий выбора перехода $F(t) = \|F_{ij}(t)\|$, $i, j = 1, \dots, K$, которые определяют функционирование полумарковской модели МСВ.

Если $F_{ij}(t) = 0$ при $t < 1$ и $F_{ij}(t) = 1$ при $t > 1$ скачки чередуются через определенную единицу времени (то есть когда они происходят регулярно), то ПМП превращается в марковскую цепь с дискретным временем. Если экспоненциальные, то ПМП становится марковским процессом с непрерывным временем.

Причиной интервально-переходной вероятности $P_{ij}(t)$ является вероятность того, что в момент времени t возможно состояние j при условии, если по окончании время 0 она находилась состоянии i .

Если $i=j$, то она имеет возможность не покидать состояния в течение промежутка времени и или же выйдя из состояний, она все-таки возвращается к состоянию j до того момента t .

Кроме того, МСВ может оказаться в произвольном состоянии j и занимает промежуточное состояние n во время своего существования.

При суммировании вероятностей вероятны обеих взаимоисключающих возможностей. Исходя из этого получаем уравнение для вероятности $P_{ij}(t)$:

$$P_{ij}(t) = V_i(t) + \sum_{n=1}^K p_{in} \int_0^t P_{nj}(t-\tau) dF_{in}(\tau) \quad (8)$$

После дифференцирования подынтегральное выражение примет вид

$$P_{ij}(t) = V_i(t) + \sum_{n=1}^K p_{in} \int_0^t f_{in}(\tau) P_{nj}(t-\tau) d\tau \quad (9)$$

где $f_{in}(\tau) = \frac{dF_{in}(\tau)}{d\tau}$ — функция плотности вероятности времени пребывания МСВ в состоянии i в направлении n .

Первый член в (9) $V_i(t)$ - вероятность того, что МСВ не покинет состояние i в момент времени t

$$V_i(t) = (1 - F_i(t))\delta_{ij}, \quad (10)$$

где $\delta_{ij} = \begin{cases} 1, & \text{при } i = j, \\ 0, & \text{при } i \neq j, \end{cases}$ - символ Кронекера; $F_i(t)$ - безусловная функция распределения времени пребывания ПМП в состоянии i , она может быть получена непосредственно через Q_{ij} по формуле

$$F_i(t) = 1 - \prod_j (1 - Q_{ij}(t)). \quad (11)$$

Тогда выражение (10) будет иметь вид

$$V_i(t) = \delta_{ij} \prod_j (1 - Q_{ij}(t)), \quad (12)$$

где $\prod_j (1 - Q_{ij}(t))$ - вероятность того, что ПМП за время t не перейдет из i -го состояния ни в какое j -е состояние; $1 - \delta_{ij} \prod_j (1 - Q_{ij}(t))$ - вероятность перехода за время t хотя бы в одно j -е состояние.

Вероятность последовательных событий в выражении (9) представлена вторым членом - вероятностью. Он описывает вероятность превращения из состояния i во время n к моменту j и затем обратно перехода из состояния I на момент N за оставшееся время $(t - \tau)$.

Все возможные частные переходы суммируются по всем промежуточным состояниям n , в которые возможны превращения из начального состояния i . И интегрируется вероятность перехода между 0 и t после такого объединения всех возможных переходов.

И в конце концов, искомые интервально-переходные вероятности объединяются с матрицей $P(t) = \|P_{ij}(t)\|$.

Помимо вероятностных параметров модель ПМП функционирования МСВ имеет возможность определить временные параметры.

Кроме вероятностных параметров модель ПМП функционирования МСВ позволяет определять временные параметры:

— безусловные математические ожидания (МОЖ) времени пребывания МСВ в каждом состоянии:

$$t_i = \int_0^{\infty} [1 - F_i(t)] dt = \int_0^{\infty} \prod_j [1 - Q_{ij}(t)] dt. \quad (13)$$

— условные МОЖ времени пребывания МСВ в состоянии:

$$t_{ij} = \int_0^t (1 - F_{ij}(t)) dt. \quad (14)$$

Система линейных интегральных уравнений (9) является основной для анализа функционирования МСВ. Она позволяет определять вероятностные параметры исследуемого процесса функционирования МСВ через основные характеристики ПМП.

Традиционный способ решения системы основан на преобразовании Лапласа. Но, получить чёткое решение в явном виде удаётся лишь только при численном составе состояний не превышающем трёх. Потенциальные проблемы, встречающиеся при исследовании реальных систем с большим количеством состояний и не только связанные со стремительным ростом вычислительных мощностей но так же потому что получаемые в данном случае лапласовы изображения функций описывающих переходной процесс получаются настолько сложны, что их открытие оригинальных оригиналов не представляется возможным. Итерационные методы эффективны при нахождении приближенного решения, если сформулировать задачу в виде оптимизации.

При функционировании МСВ задача определения интервально-переходных вероятностей основывается на решении системы линейно интегрируемых уравнений (9) каждого временного шага.

В связи с этим сформулируем следующую оптимизационную задачу условной нелинейной минимизации, решаемую в каждый момент времени t

функционирования МСВ.

Требуется найти глобальный минимум целевой функции $f(\mathbf{P}(t))$ на множестве допустимых решений $\mathbf{P}_D$, т. е. такую точку $\mathbf{P}(t)^* \in \mathbf{P}_D$, что:

$$f(\mathbf{P}(t)^*) = \min_{\mathbf{P}(t) \in \mathbf{P}_D} f(\mathbf{P}(t)) = \min_{\mathbf{P}(t) \in \mathbf{P}_D} \frac{\sqrt{\sum_{i=1}^K \sum_{j=1}^K \left(P_{ij}(t) - V_i(t) - \sum_{n=1}^K p_{in} \int_0^t f_{in}(\tau) P_{nj}(t-\tau) d\tau \right)^2}}{K^2}, \quad (15)$$

$$\text{где } \mathbf{P}_D = \left\{ \mathbf{P}(t) \left| g_l(\mathbf{P}(t)) = \sum_{j=1}^K P_{lj}(t) - 1 = 0, \quad l = 1, \dots, K \right. \right\}. \quad (16)$$

В качестве начального приближения решения оптимизационной задачи при $t=0$ принимается единичная матрица порядка K :

$$\mathbf{P}(0)^{(0)} = \mathbf{E}. \quad (17)$$

В качестве начального приближения при $t > 0$ принимается квадратная матрица интервально-переходных вероятностей, полученная на предыдущем временном шаге:

$$\mathbf{P}(t)^{(0)} = \mathbf{P}(t - \Delta t). \quad (18)$$

Итерационный процесс останавливается при одновременном выполнении следующих условий:

$$\begin{cases} \|\mathbf{P}(t)^{(k)} - \mathbf{P}(t)^{(k-1)}\| \leq \varepsilon_1, \\ \|f(\mathbf{P}(t))^{(k)} - f(\mathbf{P}(t))^{(k-1)}\| \leq \varepsilon_2, \\ \|c(\mathbf{P}(t))^{(k)} - c(\mathbf{P}(t))^{(k-1)}\| \leq \varepsilon_3, \end{cases} \quad (19)$$

где $\varepsilon_1, \varepsilon_2, \varepsilon_3$ — наперед заданная точность вычислений.

В качестве полученного решения принимается вектор $\mathbf{P}(t)^* \approx \mathbf{P}(t)^{(k)}$.

2.2. Моделирование функционирования информационно-управляющей системы мобильного видеонаблюдения

На основе описанной модели разработан алгоритм, диаграмма активности которая представлена на рисунке 9. Алгоритм основан на

решении указанной выше оптимизационной задачи методом последовательного квадратичного программирования (SQP), который программно реализован в Matlab функцией `fmincon(...)`.

Реализация метода SQP состоит из трех основных стадий:

1. Корректировка матрицы Гессе для Лагранжевой функции.
2. Решение задачи квадратичного программирования.
3. Вычисление линейного поиска и функции выгоды.

В таблице 1 представлены принятые параметры функции `fmincon` для решения обозначенной задачи оптимизации.

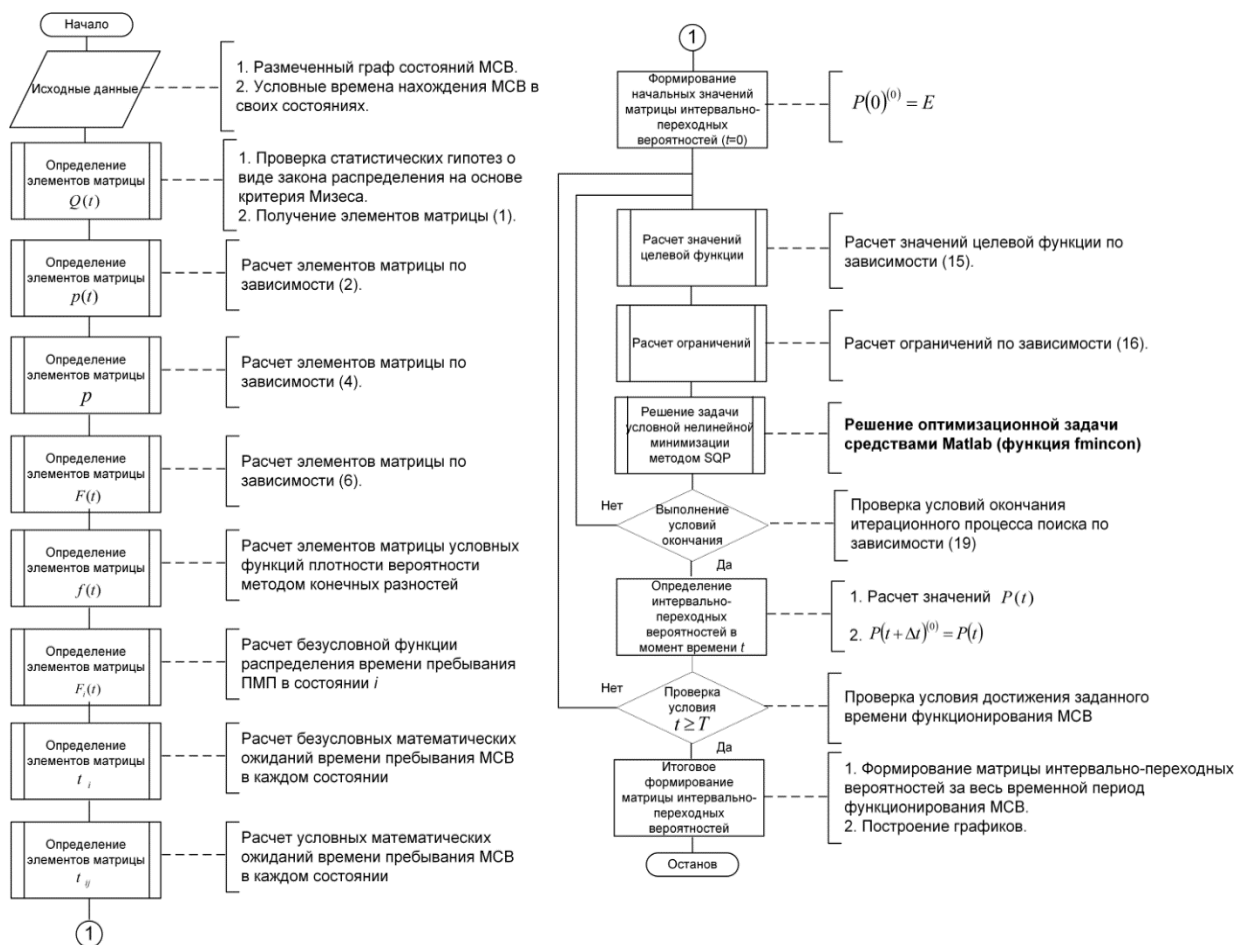


Рисунок 9 – Блок-схема алгоритма определения интервально-переходных вероятностей МСВ

Рассмотрим более подробно алгоритм определения интервально-переходных вероятностей МСВ:

- исходные данные включают в себя: размеченный граф состояний

МСВ. Условные времена нахождения МСВ в своих состояниях;

- определение элементов матрицы $Q(t)$: проверка статических гипотез о виде закона распределения на основе критерия Мизеса. Получение элементов матрицы (1);

- определение элементов матрицы $p(t)$: расчет элементов матрицы по зависимости (2);

- определение элементов матрицы p : расчет элементов матрицы по зависимости (4);

- определение элементов матрицы $F(t)$: расчет элементов матрицы по зависимости (6);

- определение элементов матрицы $f(t)$: расчет элементов матрицы условных функций плотности вероятности методом конечных разностей;

- определение элементов матрицы $F_i(t)$: расчет безусловной функции распределения времени пребывания ПМП в состоянии i ;

- определение элементов матрицы t_i : расчет безусловных математических ожиданий времени пребывания МСВ в каждом состоянии;

- определение элементов матрицы t_{ij} : расчет условных математических ожиданий времени пребывания МСВ в каждом состоянии;

- формирование начальных значений матрицы интервально-переходных вероятностей ($t=0$): $P(0)^{(0)} = E$;

- расчет значений целевой функции: расчет значений целевой функции по зависимости (15);

- расчет ограничений: расчет ограничений по зависимости (16);

- решение задачи условной нелинейной минимизации методом SQP: решение оптимизационной задачи средствами Matlab (функция `fmincon`);

- выполнение условий окончания: проверка условий окончания итерационного процесса поиска по зависимости (19);

- определение интервально-переходных вероятностей в момент

времени t : расчет значений $P(t)$, $P(t+\Delta t)^{(0)} = P(t)$;

— проверка условия $t \geq T$: проверка условия достижения заданного времени функционирования МСВ;

— итоговое формирование матрицы интервально-переходных вероятностей: формирование матрицы интервально-переходных вероятностей за весь временной период функционирования МСВ, построение графиков.

Таблица 1 –Параметры функции `fmincon`

Параметр	Описание параметра	Значение параметра
Algorithm	Алгоритм, используемый для решения оптимизационной задачи	'active-set'
Display	Уровень отображения. 'off' отображение не производится, 'iter' отображение проводится на каждой итерации, 'final' (принимается по умолчанию) отображение только конечной информации	'off'
MaxFunEvals	Максимально число допустимых расчетов функции	6000
MaxIter	Максимально число допустимых итераций	1500
TolFun	Конечное допустимое отклонение по значению функции (ε_2)	10^{-4}
TolCon	Конечное допустимое отклонение по нарушению условий ограничения (ε_3)	10^{-4}
TolX	Конечное допустимое отклонение по значению x (ε_1)	10^{-6}

Разработанный алгоритм программно реализован в Matlab.

Так как получение опытных значений интервально-переходных вероятностей для МСВ невозможно, то оценку адекватности разработанной полумарковской модели функционирования МСВ проведем на основе сравнения с результатами, полученными на хорошо апробированном

аппарате марковских процессов (МП). Для проверки адекватности будем использовать следующее положение: при условии экспоненциальности времен нахождения МСВ в каждом состоянии и равенстве МОЖ времен нахождения в состояниях, финальные вероятности МП, полученные на основе решения уравнений Колмогорова и интервально-переходные вероятности, рассчитанные в установившемся режиме на основе ПМП, должны быть равны между собой.

Решим сначала задачу определения вероятностей состояний МСВ, принимая в качестве допущений экспоненциальность времен нахождения МСВ в каждом состоянии и равенство их МОЖ, т. е. рассмотрим МП с дискретными состояниями и непрерывным временем. Для решения этой задачи составим систему уравнений Колмогорова [6]. Размеченному графу, который представлен на рисунке 1, соответствует следующая система уравнений Колмогорова:

$$\left\{ \begin{array}{l} \frac{dp_1(t)}{dt} = \lambda_{31}p_3(t) + \lambda_{51}p_5(t) - (\lambda_{12} + \lambda_{14})p_1(t), \\ \frac{dp_2(t)}{dt} = \lambda_{12}p_1(t) - (\lambda_{23} + \lambda_{24})p_2(t), \\ \frac{dp_3(t)}{dt} = \lambda_{23}p_2(t) - (\lambda_{31} + \lambda_{34})p_3(t), \\ \frac{dp_4(t)}{dt} = \lambda_{14}p_1(t) + \lambda_{24}p_2(t) + \lambda_{34}p_3(t) - \lambda_{45}p_4(t), \\ \frac{dp_5(t)}{dt} = \lambda_{45}p_4(t) - \lambda_{51}p_5(t), \end{array} \right. \quad (20)$$

Начальные условия для решения системы уравнений (20) следующие:
при $t = 0$

$$p_1(0)=1, \quad p_2(0)=0, \quad p_3(0)=0, \quad p_4(0)=0, \quad p_5(0)=0. \quad (21)$$

Система линейных дифференциальных уравнений (20) при начальных условиях (21) имеет единственное решение, если заданы интенсивности перехода из состояния в состояние λ . Учитывая допущение о равенстве математических ожиданий времен нахождения в состояниях, примем $\lambda=0,2$.

Систему уравнений Колмогорова (20) с начальными условиями (21) решим методом численного интегрирования Рунге-Кутты, который реализован в Matlab функцией ode45(...).

Теперь решим эту же задачу на основе использования математического аппарата ПМП. При задании ПМП МСВ примем экспоненциальный закон для всех независимых функций $Q_{ij}(t)$ с $\lambda=0,2$. В качестве начального состояния системы примем состояние «1». Решение системы (9) реализовано в системе Matlab на основе использования разработанного алгоритма.

Полумарковские вероятности состояний МСВ в установившемся режиме можно определить двумя способами. Первый способ основан на том факте, что с увеличением времени функционирования, в системе наступает стационарный режим, когда полумарковские вероятности становятся не зависимыми от начального состояния. Второй способ базируется на непосредственном определении стационарных полумарковских вероятностей состояний через вероятности вложенной марковской цепи по следующей зависимости:

$$P_i = \frac{z_i t_i}{\sum_{i=1}^5 z_i t_i}, \quad (22)$$

где $t_i = \int_0^{\infty} \prod_j [1 - Q_{ij}(t)] dt$ - безусловное МОЖ времени пребывания МСВ в i -м состоянии перед переходом в любое другое состояние; z_i - финальные вероятности вложенной цепи Маркова.

Для нахождения финальных вероятностей вложенной цепи Маркова составляется система алгебраических уравнений

$$\mathbf{z} = \mathbf{zP} \quad \text{при} \quad \sum_{i=1}^k z_i = 1, \quad (23)$$

где $\mathbf{z} = (z_1, z_2, \dots, z_5)$ - вектор-строка; k - общее число состояний процесса функционирования МСВ, определяющее стационарные вероятности z_i

застать вложенный МП в момент произвольного перехода в состояние i ;
 $\mathbf{p} = \|p_{ij}\|$ - стационарные переходные вероятности.

Программно реализовав вышеуказанные зависимости в Matlab, получены численные значения полумарковских вероятностей состояний в установившемся режиме.

3. Программное обеспечение компьютерной модели информационно-управляющей системы мобильного видеонаблюдения

3.1. Процесс проектирования программных средств и структур данных для задачи определения вероятностных параметров функционирования информационно-управляющей системы мобильного видеонаблюдения

Процесс проектирования программных средств и структур данных обычно включает в себя несколько этапов. Рассмотрим их основные этапы:

1. Анализ требований – этот этап включает в себя описание того, как система должна работать, и какие функции должны быть доступны пользователю.

2. Проектирование базы данных – этот этап включает в себя описание схемы базы данных, которая будет использоваться для хранения информации.

3. Проектирование архитектуры – этот этап включает в себя определение архитектурного решения, т. е. как будут взаимодействовать компоненты системы, какие технологии будут использоваться и т. д.

4. Проектирование пользовательского интерфейса – этот этап включает в себя проектирование интерфейса, который будет использоваться пользователями для работы с системой.

5. Разработка кода – этот этап включает в себя написание кода, который будет использоваться для реализации проекта.

6. Тестирование – этот этап включает в себя проверку, что система работает так, как было задумано на этапе анализа требований.

7. Внедрение и сопровождение – этот этап включает в себя выпуск системы в эксплуатацию и поддержку ее работоспособности.

Каждый этап включает в себя перечень задач, которые должны быть выполнены для обеспечения корректной работы системы и обслуживания клиентов. Тактика конструирования программных средств и структур данных чаще всего включает в себя применение формализованной методологии,

строгого соблюдения стандартов при разработке. Это нужно для обеспечения качества системной работы.

Для показа действий одного объекта в течении программы, от создания до его уничтожения применяется диаграмма деятельности (рисунок 10).

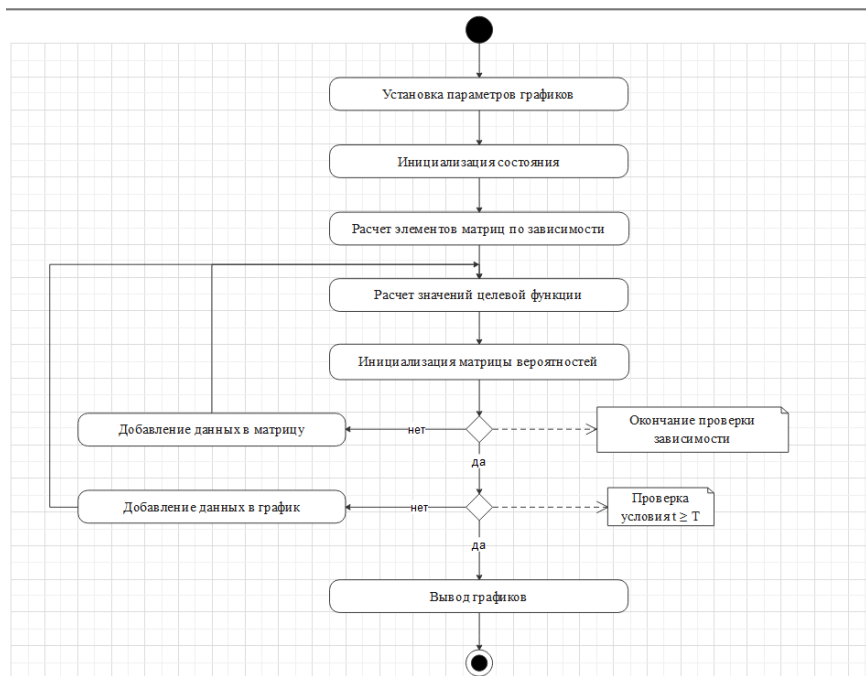


Рисунок 10 – Диаграмма деятельности

Диаграмма деятельности (activity diagram) – это графическое представление действий и задач, которые связаны с определённым проектом или деятельностью. На данной диаграмме можно увидеть, как происходит анализ и оптимизация бизнес-процессов. Она помогает принимать решения по различным вопросам управления проектами или планировать работу компании. Набор символов и линий, являющихся составляющими диаграммы деятельности имеет свое значение (назначение) в соответствии с полученной информацией.

Диаграмма деятельности помогает найти последовательность действий, которые следует выполнять в соответствии с определенным проектом или процессом. Она может визуализировать последовательность действий и операций, которые необходимо выполнить для получения желаемого результата.

Принцип диаграммы деятельности – графическое отображение последовательностей действий. Каждая задача на диаграмме деятельности имеет табличный состав, в котором указываются кто выполняет задачу, какие ресурсы необходимы для ее выполнения и какие итоги будут получены после завершения каждого этапа процесса. Кроме того представлена последовательность действий по которой можно успешно завершить проект.

В целом, диаграмма деятельности – сильный инструмент для планирования и управления проектами. Она способствует выявлению проблем и определить области для совершенствования процессов, что в свою очередь приводит к повышению эффективности деятельности.

3.2. Программное обеспечение для решения вероятностных параметров функционирования информационно-управляющей системы мобильного видеонаблюдения

Я выбрала язык программирования C++, так как он простой и понятный. Много программных средств создано для написания на этом языке. Например:

Visual Studio Code.

Общая структура программы является удобным и современным интерфейсом. Данная программа поддерживает подсветку синтаксиса языка по умолчанию, автоматическое дополнение кода и IntelliSense. Также в ней есть встроенная система для нахождения ошибок: она помогает взаимодействовать с API, добавлять код элементами из проекта созданных программ. Это удобство дополняется функцией компиляции языка с помощью специального плагина. В одном интерфейсе можно найти все инструменты, которые нужны для разработки.

Microsoft Studio C++.

При разработке с использованием языка C++ используется проприетарная программная платформа. Visual Studio поддерживает Node.js, Python и C# языки. Необходимые приложения можно создать прямо в

программе. Это доступно через удобный интерфейс для управления проектами, многооконный текстовый редактор скриптов и встроенный быстрый компилятор. А также имеет место быть удобная система тестирования кода со всеми необходимыми инструментами на консоли и прочими атрибутами. Системы IntelliSense является аналогичной Vs Code. Она также помогает писать код быстрее и эффективней. Она может автоматически дополнять и проверять строки кода, не допускать ошибок в коде при работе со сторонними API.

G++.

Программа умеет работать на Linux, macOS. Стандартные компиляторы g++ не имеют среды для разработки в отличие от IDE с его отсутствием возможности создания приложений. Они занимаются всего лишь переводом текста с языка C++ на язык машин, который понятен для компьютерного процессора. G++ – это специальная утилита для Unix, запускаемая через командную строку.

Dev – C++.

Dev-C++ является полноформатной IDE для написания и компиляции кода на C, который основан в языке C. Уже с 1998 года утилита является одной из самых главных программных продуктов. Многие разработчики, студенты и исследователи используют ее.

Если исходить из программ, которые я привела для реализации приложений на языке C++. Я остановилась на программе Microsoft Studio C++. Так как в этой программе имеется облачное хранилище данных, удобный интерфейс и множество встроенных языков. К тому же я имею опыт написания программ на данном приложении.

Я использую специализированное приложение для разработки пользовательских интерфейсов (GUI) и мульти-платформенных приложений, которые работают на большинстве настольных компьютеров с их вариациями до мобильных устройств или встраиваемых систем. В большинстве

программ, созданных с помощью Qt, используется собственный интерфейс. В данном случае он классифицируется как инструментарий виджетов.

Qt включает в себя библиотеку классов C++, а также инструменты для разработки кросс-платформенных приложений с графическим интерфейсом (GUI). Для других языков есть вариации: PyQt для языка программирования Java, QtoRubi на Rabby и т. д.

В большинстве случаев Qt используют для создания очень быстрых и высокопроизводительных приложений. Это могут быть мессенджеры, игры или трудоемкие и сложные программные продукты. Он популярен в областях, с повышенной потребностью во внимании к безопасности ПО.

У Qt есть свои преимущества. Рассмотрим их:

1. Кросс-платформенность. Qt – кросс-платформенный фреймворк. То есть он актуален для всех операционных систем: Windows, Linux, iOS и Android. При разработке под устройства различного назначения, от микроконтроллеров до суперкомпьютеров применяют фреймворк.

2. Высокая скорость. Программы, написанные на C++, легко запускаются и быстро обрабатываются. Также C++ является компилируемым языком программирования. Следовательно, компилятор сохраняет исходный код в C++ и транслирует его во исполняемый файл. Этот набор инструкций – то самое машинное описание – влияет на скорость работы; также это значит трансляцию командной строки из общего кода (программы) в исполнительную строку;

3. Удобная среда разработки. Благодаря Qt Creator, в котором легко разобраться даже новичку. Она содержит в себе все необходимое, нужные компоненты находятся рядом и сам инструмент понятен. Разработчик может легко найти проблемные участки кода, так как там удобно организована отладка.

4. Быстрое создание GUI. Инструменты и дополнительное программное обеспечение помогают быстро спроектировать интерфейс, а

также создать дизайн. Qt Creator – отличный фреймворк для разработки приложений, ориентированных на графический интерфейс.

5. Взаимодействие процессов. Также благодаря системе метаобъектов Qt имеет возможность более гибко управлять межпроцессным взаимодействием, чем C++. Однако в настоящее время это преимущество не так актуально, поскольку актуальны версии C++ 11 и выше. Но очень большое количество проектов пользуется легаси-кодом, написанным на старых версиях языка. Кроме того, все так же важно соблюдать особенности взаимодействия.

3.3. Валидация разработанной компьютерной модели

Результаты графических расчетов, реализованные на языке C++ и приложения Qt

Рассмотрим результаты представления более подробно. На рисунке 11 реализовано состояние S1 – МСВ занимает указанное место и разворачивается для ведения наблюдения.

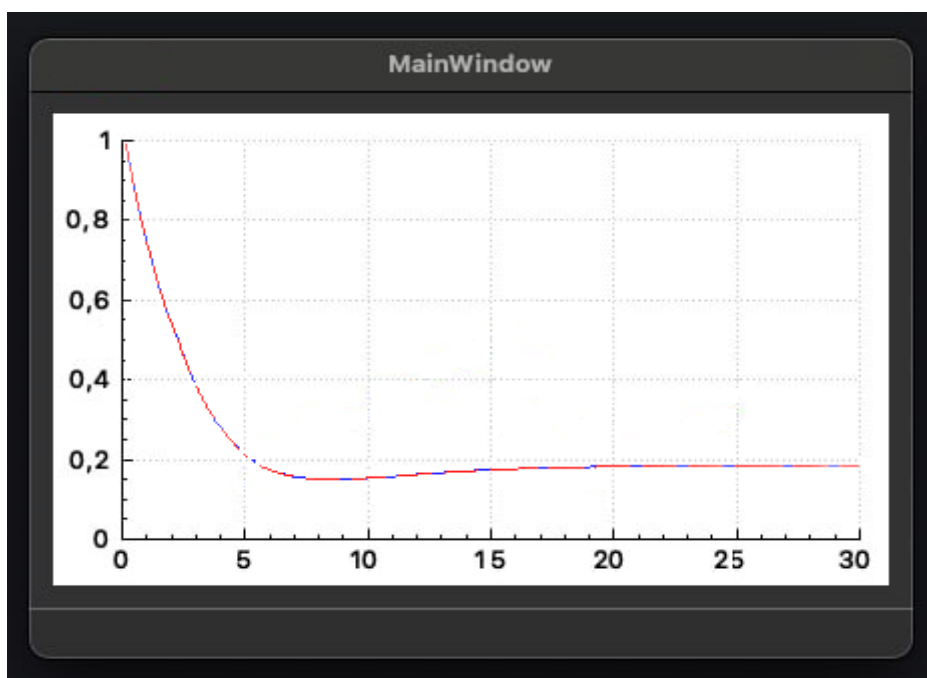


Рисунок 11 – Вероятность нахождения МСВ в состоянии «1»

На рисунке 12 реализовано состояние S2 – МСВ ведет наблюдение и обрабатывает получаемую информацию.

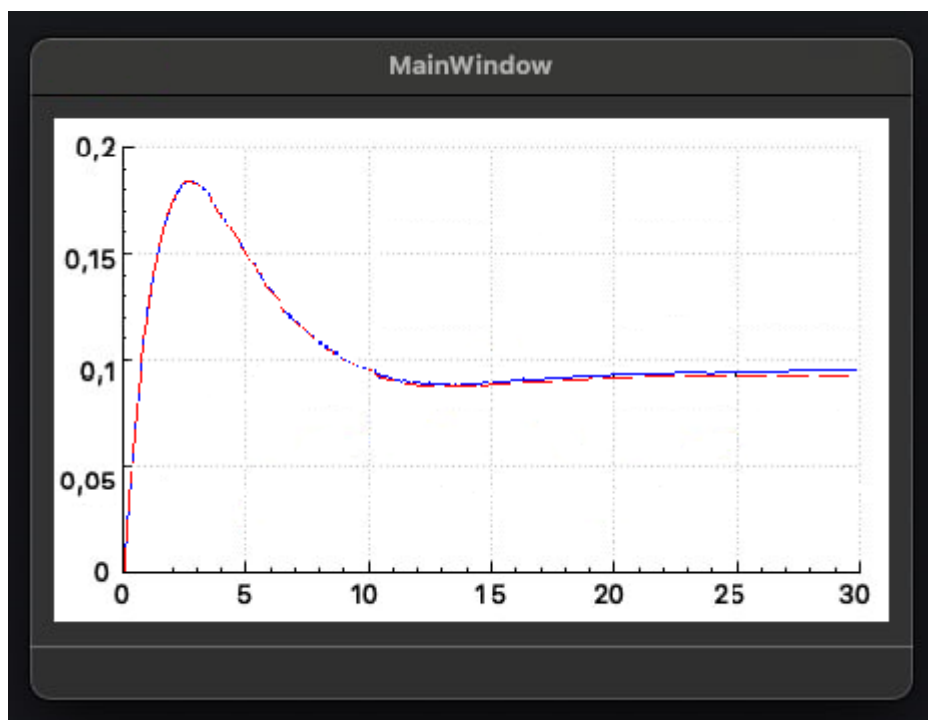


Рисунок 12 – Вероятность нахождения МСВ в состоянии «2»

На рисунке 13 реализовано состояние S3 – МСВ свертывается и производит смену места наблюдения.

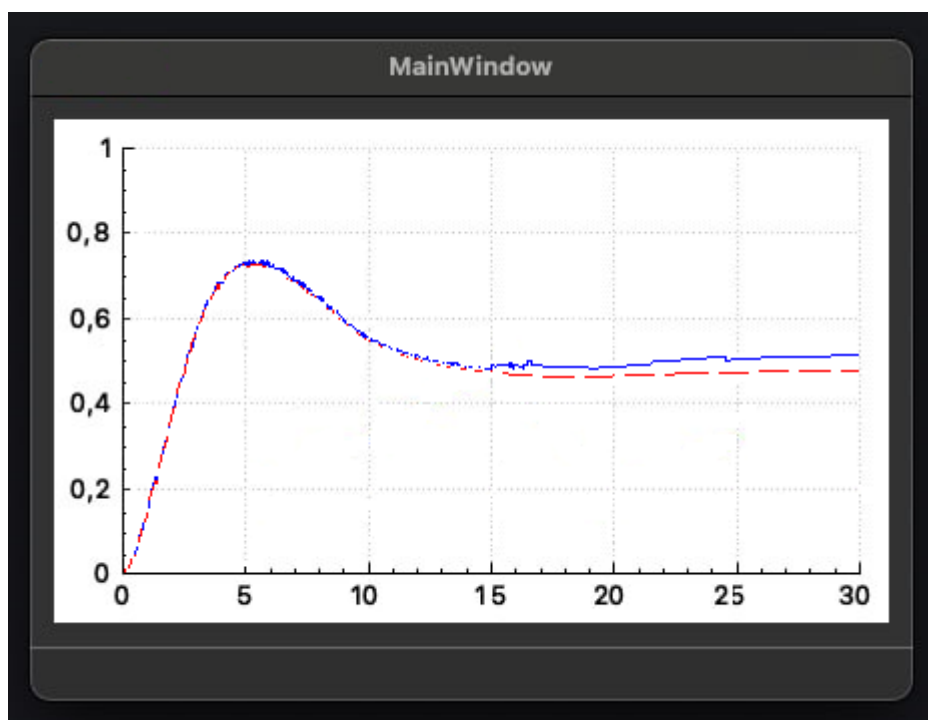


Рисунок 13 – Вероятность нахождения МСВ в состоянии «3»

На рисунке 14 реализовано состояние S4 – МСВ обнаружена и находится под негативным воздействием.

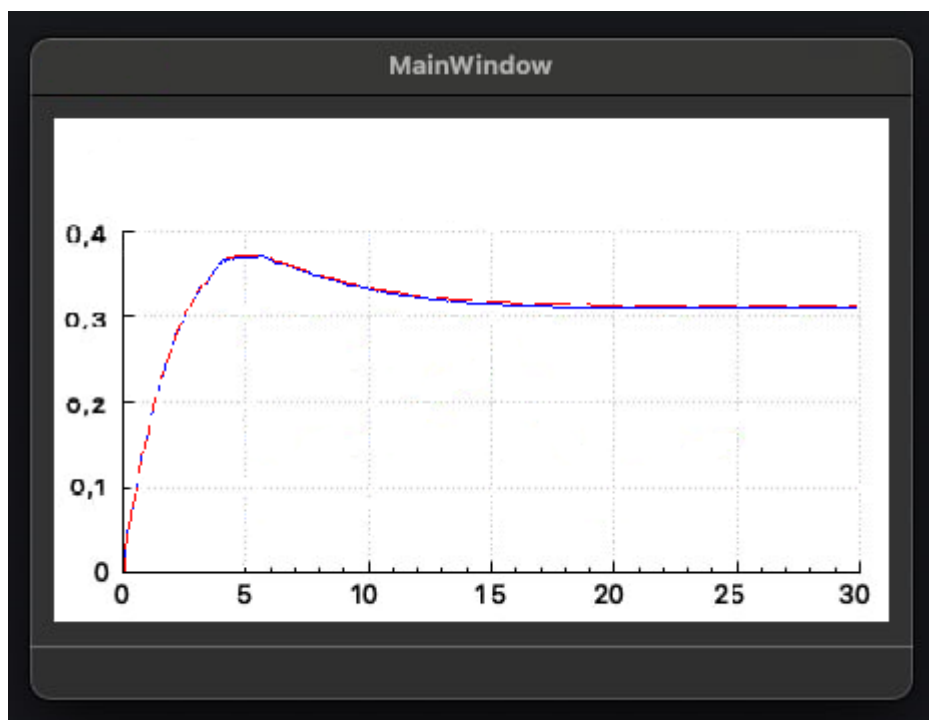


Рисунок 14 – Вероятность нахождения МСВ в состоянии «4»

На рисунке 15 реализовано состояние S5 – МСВ неисправна и ремонтируется.

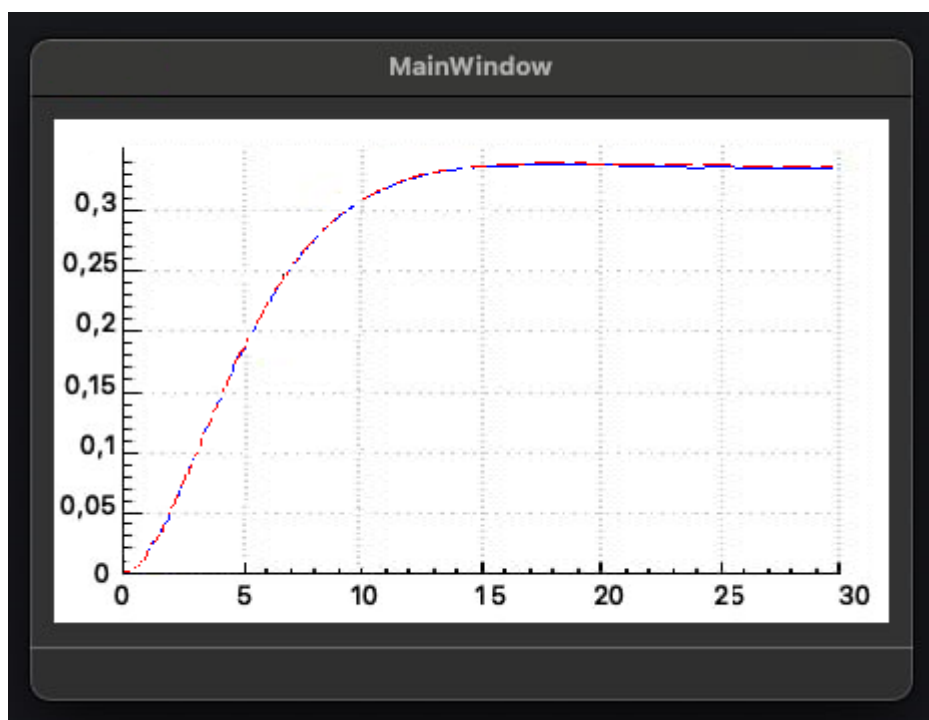


Рисунок 15 – Вероятность нахождения МСВ в состоянии «5»

На рисунке 16 представлена гистограмма, которая сравнивает все стационарные полумарковские вероятности.

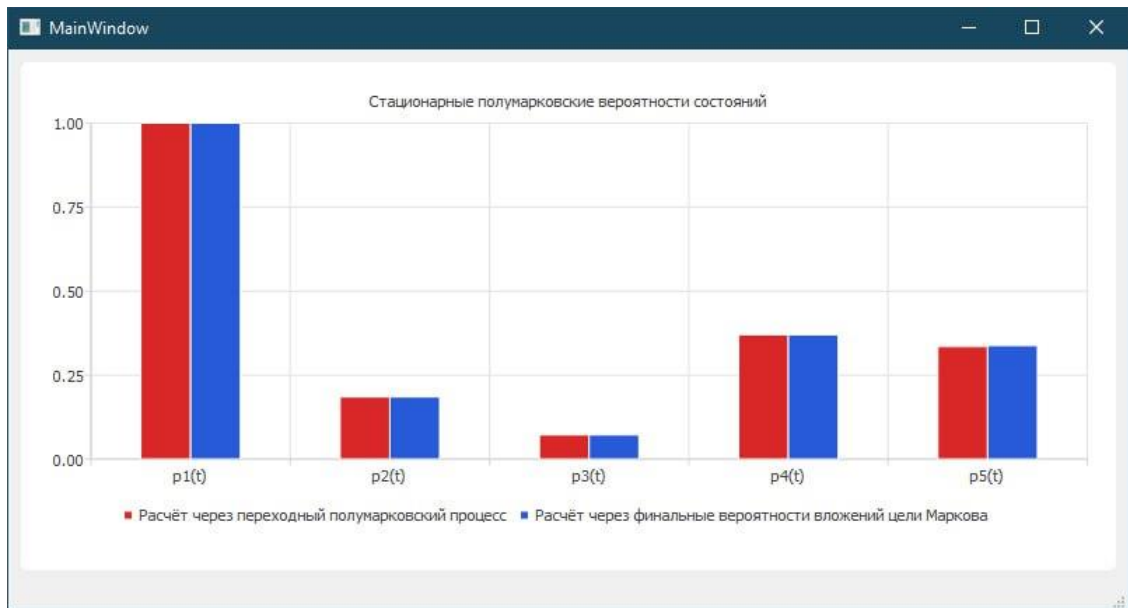


Рисунок 16 – Стационарные полумарковские вероятности состояний

Каждый график показывает вероятность нахождения в этом состоянии, где 0 это не может, а 1 это находится, между ними то что вероятно может скорее произойти или наоборот не произойти, в зависимости от текущего состояния, в этом суть марковской цепи – последовательность действий влияет на дальнейшее развитие событий. На каждом из графиков было изображено полумарковский процесс и марковский процесс. Отличие марковского процесса от полумарковского заключается в том, что для марковского имеет значение только текущее состояние, а для полумарковского ещё и предыдущие. Так как определяет вероятность будущего события. И графики показывают примерно одинаковые значения.

Эти 5 состояний отличаются начальными значениями, но всё равно приходят к одному.

График выпрямляется, потому что система приходит в стационарный режим с увеличением времени, графики это отражают. Можно заметить что вероятности полумарковских и марковских процессов очень близки или прям равны, хотя полумарковские не зависимы от начального состояния.

3.4. Расчет бюджета на разрабатываемый продукт

Для планирования времени на разрабатываемый продукт я использовала программу ProjectLibre.

ProjectLibre является кроссплатформенным программный обеспечением, созданном для осуществления планирования и управления проектами в качестве альтернативы платной программе Microsoft Project, и замены предшествующей программы OpenProj.

Предоставляет возможность составлять проекты, в которые записывается иерархический перечень задач внутри данного проекта. Цель задачи может быть ограничена во времени, иметь ответственных исполнителей и показывать процент завершения. В этой программе есть возможность делать отчёты по проектам, строить различные графики и организовывать выборки, используя фильтры.

ProjectLibre имеет ряд особенностей, которые предназначены для операционной системы Windows

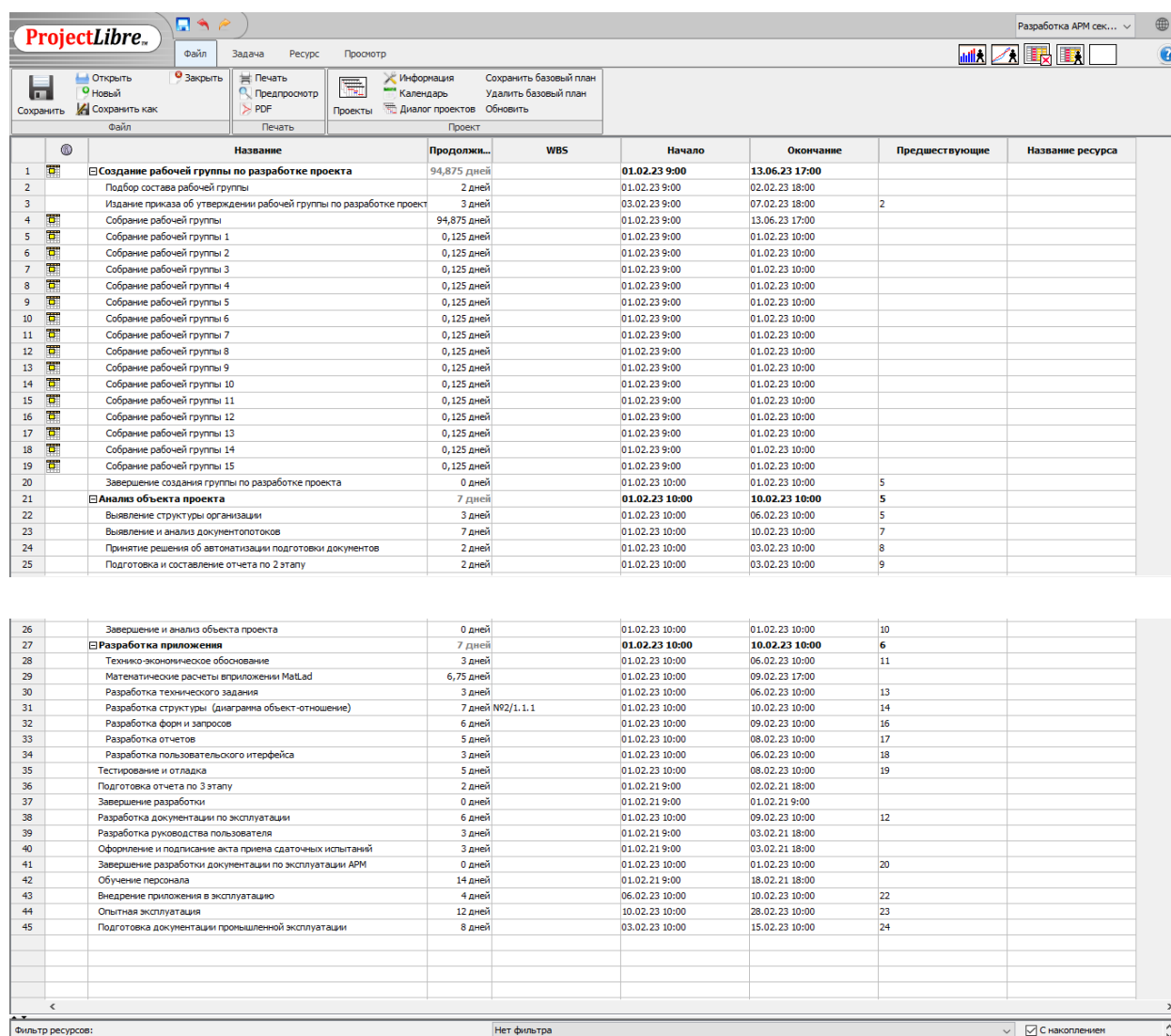
- Формат файлов Microsoft Project 2010 поддерживает, а также диаграммы Ганта.
- Данный способ предоставляет возможность производить импорт и экспорт проектов, а также их отправку в печать.
- Оборудование: ленточный интерфейс;
- Разработка списков задач.
- Формирование чартов и диаграмм;
- В том числе создание отчётов, подготовка графиков.
- Главным окном находится список задач.

Любой желающий может установить ProjectLibre и использовать его в своих целях. Создание документов стало проще, не нужно больше разделять составляющие проекта. Фразой «формулируйте задачи, располагайте их в иерархическом порядке и достигайте поставленных целей» можно построить

иерархию задач по которым планируется достичь цели. Работать над каждой задачей можно, как с отдельной единицей: назначать ответственных за порученное дело и оценивать процент завершения.

Этот способ декомпозиции состоит в том, чтобы составить список дел по созданию ИТ-продукта, который детализируется настолько насколько это возможно. Более точный результат даст детальнее сделанная декомпозиция.

План для разрабатываемого продукта мобильного видеонаблюдения представлен на рисунке 17.



	Название	Продолжи...	WBS	Начало	Окончание	Предшествующие	Название ресурса
1	Создание рабочей группы по разработке проекта	94,875 дней		01.02.23 9:00	13.06.23 17:00		
2	Подбор состава рабочей группы	2 дней		01.02.23 9:00	02.02.23 18:00		
3	Идание приказа об утверждении рабочей группы по разработке проект	3 дней		03.02.23 9:00	07.02.23 18:00	2	
4	Собрание рабочей группы	94,875 дней		01.02.23 9:00	13.06.23 17:00		
5	Собрание рабочей группы 1	0,125 дней		01.02.23 9:00	01.02.23 10:00		
6	Собрание рабочей группы 2	0,125 дней		01.02.23 9:00	01.02.23 10:00		
7	Собрание рабочей группы 3	0,125 дней		01.02.23 9:00	01.02.23 10:00		
8	Собрание рабочей группы 4	0,125 дней		01.02.23 9:00	01.02.23 10:00		
9	Собрание рабочей группы 5	0,125 дней		01.02.23 9:00	01.02.23 10:00		
10	Собрание рабочей группы 6	0,125 дней		01.02.23 9:00	01.02.23 10:00		
11	Собрание рабочей группы 7	0,125 дней		01.02.23 9:00	01.02.23 10:00		
12	Собрание рабочей группы 8	0,125 дней		01.02.23 9:00	01.02.23 10:00		
13	Собрание рабочей группы 9	0,125 дней		01.02.23 9:00	01.02.23 10:00		
14	Собрание рабочей группы 10	0,125 дней		01.02.23 9:00	01.02.23 10:00		
15	Собрание рабочей группы 11	0,125 дней		01.02.23 9:00	01.02.23 10:00		
16	Собрание рабочей группы 12	0,125 дней		01.02.23 9:00	01.02.23 10:00		
17	Собрание рабочей группы 13	0,125 дней		01.02.23 9:00	01.02.23 10:00		
18	Собрание рабочей группы 14	0,125 дней		01.02.23 9:00	01.02.23 10:00		
19	Собрание рабочей группы 15	0,125 дней		01.02.23 9:00	01.02.23 10:00		
20	Завершение создания группы по разработке проекта	0 дней		01.02.23 10:00	01.02.23 10:00	5	
21	Анализ объекта проекта	7 дней		01.02.23 10:00	10.02.23 10:00	5	
22	Выявление структуры организации	3 дней		01.02.23 10:00	06.02.23 10:00	5	
23	Выявление и анализ документопотоков	7 дней		01.02.23 10:00	10.02.23 10:00	7	
24	Принятие решения об автоматизации подготовки документов	2 дней		01.02.23 10:00	03.02.23 10:00	8	
25	Подготовка и составление отчета по 2 этапу	2 дней		01.02.23 10:00	03.02.23 10:00	9	
26	Завершение и анализ объекта проекта	0 дней		01.02.23 10:00	01.02.23 10:00	10	
27	Разработка приложения	7 дней		01.02.23 10:00	10.02.23 10:00	6	
28	Техико-экономическое обоснование	3 дней		01.02.23 10:00	06.02.23 10:00	11	
29	Математические расчеты в приложении MatLab	6,75 дней		01.02.23 10:00	09.02.23 17:00		
30	Разработка технического задания	3 дней		01.02.23 10:00	06.02.23 10:00	13	
31	Разработка структуры (диаграмма объект-отношение)	7 дней	NF2/1.1.1	01.02.23 10:00	10.02.23 10:00	14	
32	Разработка фори и запросов	6 дней		01.02.23 10:00	09.02.23 10:00	16	
33	Разработка отчетов	5 дней		01.02.23 10:00	08.02.23 10:00	17	
34	Разработка пользовательского итерфейса	3 дней		01.02.23 10:00	06.02.23 10:00	18	
35	Тестирование и отладка	5 дней		01.02.23 10:00	08.02.23 10:00	19	
36	Подготовка отчета по 3 этапу	2 дней		01.02.21 9:00	02.02.21 18:00		
37	Завершение разработки	0 дней		01.02.21 9:00	01.02.21 9:00		
38	Разработка документации по эксплуатации	6 дней		01.02.23 10:00	09.02.23 10:00	12	
39	Разработка руководства пользователя	3 дней		01.02.21 9:00	03.02.21 18:00		
40	Оформление и подписание акта приема сдаточных испытаний	3 дней		01.02.21 9:00	03.02.21 18:00		
41	Завершение разработки документации по эксплуатации АРМ	0 дней		01.02.23 10:00	01.02.23 10:00	20	
42	Обучение персонала	14 дней		01.02.21 9:00	18.02.21 18:00		
43	Внедрение приложения в эксплуатацию	4 дней		06.02.23 10:00	10.02.23 10:00	22	
44	Опытная эксплуатация	12 дней		10.02.23 10:00	28.02.23 10:00	23	
45	Подготовка документации промышленной эксплуатации	8 дней		03.02.23 10:00	15.02.23 10:00	24	

Рисунок 17– План работы

Бюджет для разрабатываемого программного обеспечения рассчитывается опираясь на специалистов, которые необходимы чтобы реализовать данный продукт. Нам потребуются:

1. Специалист по MatLab, который должен спроектировать математическую модель и произвести расчеты. Получают зарплату в диапазоне от 100 тысяч рублей до 120 тыс.рублей;

2. Программист разрабатывает код для языка C++, опираясь на математические расчеты специалиста MatLab. Общая сумма зарплаты для такого проекта составляет 140 тысяч рублей;

3. Тестировщик. Человек, который будет стараться «сломать» данную программу или его коллеги по тестированию. Т.Данный специалист должен искать ошибки во всех расчетах, а также в кодах. Заработная плата такого специалиста может доходить до 100 тысяч рублей.

Ну что же, подведем итоги. Бюджет такого проекта составит примерно от 360 тысяч рублей.

Заключение

В ходе выполнения выпускной квалификационной работы мной были рассмотрена и создана компьютерная модель функционирования информационно-управляющей системы мобильного видеонаблюдения, на основе использования положений теории полумарковских процессов.

Список используемых источников

1. Мобильное видеонаблюдение – 2012. Тренды и перспективы // [Электронный ресурс]: URL: <http://www.secuteck.ru>
2. Алгоритм оценивания и исследования готовности системы управления судном морского транспорт// [Электронный ресурс]: URL: <https://cyberleninka.ru/article/n/algoritm-otsenivaniya-i-issledovaniya-gotovnosti-sistemy-upravleniya-sudnom-morskogo-transporta>
3. Теория вероятностей и математическая статистик // [Электронный ресурс]. URL: <https://elib.pnzgu.ru/library/10313200>
4. Марковские процессы // [Электронный ресурс]. URL: https://www.studmed.ru/tihonov-an-mironov-ma-markovskie-processy_e8f5593bcba.html
5. Оптимизация при наличии ограничений [Электронный ресурс] // [Электронный ресурс]. URL: http://matlab.exponenta.ru/optimiz/book_1/15.php.
6. Имитационное моделирование систем и сетей массового обслуживания средствами приложения к пакету Mathcad// [Электронный ресурс]. URL: <https://cyberleninka.ru/article/n/imitatsionnoe-modelirovanie-sistem-i-setey-massovogo-obsluzhivaniya-sredstvami-prilozheniya-k-paketu-mathcad>
7. Метод последовательных упрощений полумарковского процесса: // [Электронный ресурс]. URL: <https://cyberleninka.ru/article/n/metod-posledovatelnyh-uproscheniy-polumarkovskogo-protssesa>
8. Полумарковский процесс описания алгоритмов // [Электронный ресурс]. URL: <https://cyberleninka.ru/article/n/polumarkovskiy-protsess-kak-model-dlya-opisaniya-algoritmov>
9. Временные характеристики полумарковских процессов обработки информации специализированной ЭВМ// [Электронный ресурс]. URL: <https://cyberleninka.ru/article/n/vremennye-harakteristiki-polumarkovskih-protsessov-obrabotki-informatsii-spetsializirovannoy-evm>

- 10.Обобщенная полумарковская модель алгоритма управления цифровыми устройствами// [Электронный ресурс]. URL: <https://cyberleninka.ru/article/n/obobschennaya-polumarkovskaya-model-algoritma-upravleniya-tsifrovymi-ustroystvami>
11. Моделирование распределенной системы видеонаблюдения // [Электронный ресурс]. URL: <https://cyberleninka.ru/article/n/modelirovanie-raboty-raspredelennoy-sistemy-videonablyudeniya/viewer>
12. Современные системы видеонаблюдения этапы развития // [Электронный ресурс]. URL: <https://cyberleninka.ru/article/n/sovremennye-sistemy-videonablyudeniya-etapy-razvitiya/viewer>
13. Современные системы видеонаблюдения и их составляющие // [Электронный ресурс]. URL: <https://cyberleninka.ru/article/n/sovremennye-sistemy-videonablyudeniya-i-ih-sostavlyayuschie/viewer>
14. Системы безопасности их типы и структуры // [Электронный ресурс]. URL: http://fors.gtechs.ru/sites/all/forsdocs/2014Zadorojn/Strelec_doc140409.pdf
- 15.SystemView + Matlab + Simulink. Возможности программы Matlab для моделирования в программе Simulink структуры // [Электронный ресурс]. URL: <https://cyberleninka.ru/article/n/systemview-matlab-simulink-vozmozhnosti-programmy-matlab-dlya-modelirovaniya-v-programme-simulink>
- 16.О применении Matlab к решению статистических задач// [Электронный ресурс]. URL: <https://cyberleninka.ru/article/n/o-primenenii-matlab-k-resheniyu-statisticheskikh-zadach>
- 17.Matlab — новые возможности в технологии осциллографии// [Электронный ресурс]. URL: <https://cyberleninka.ru/article/n/matlab-novye-vozmozhnosti-v-tehnologii-ostsillografi>
- 18.Промышленное видеонаблюдение// [Электронный ресурс]. URL: <https://cyberleninka.ru/article/n/promyshlennoe-videonablyudenie>

- 19.К вопросу о системах видеонаблюдения// [Электронный ресурс]. URL:
<https://cyberleninka.ru/article/n/k-voprosu-o-sistemah-videonablyudeniya>
- 20.Верификация систем видеонаблюдения// [Электронный ресурс]. URL:
<https://cyberleninka.ru/article/n/verifikatsiya-sistem-videonablyudeniya>

Приложение А. Листинг программы
(рекомендуемое)

```

#include <iostream>
#include <cmath>
#include "fmincon.hpp"

const double t12 = 5;
const double t14 = 5;
const double t23 = 5;
const double t24 = 5;
const double t31 = 5;
const double t34 = 5;
const double t45 = 5;
const double t51 = 5;
const double L = 0.2;
const double T = 30;
const int kt = 10;
const int np = 5;

double p1(double t, const std::vector<double>& Y) {
    return L * Y[2] + L * Y[4] - (L + L) * Y[0];
}

double p2(double t, const std::vector<double>& Y) {
    return L * Y[0] - (L + L) * Y[1];
}

double p3(double t, const std::vector<double>& Y) {
    return L * Y[1] - (L + L) * Y[2];
}

double p4(double t, const std::vector<double>& Y) {
    return L * Y[0] + L * Y[1] + L * Y[2] - L * Y[3];
}

double p5(double t, const std::vector<double>& Y) {
    return L * Y[3] - L * Y[4];
}

std::vector<double> div(const std::vector<double>& Y) {
    std::vector<double> F(5);
    F[0] = p1(0, Y);
    F[1] = p2(0, Y);
    F[2] = p3(0, Y);
    F[3] = p4(0, Y);
    F[4] = p5(0, Y);
    return F;
}

std::vector<double> ode45(const std::vector<double>& Y0) {
    const double h = 0.01;
    std::vector<double> tspan;
    for (double t = 0; t <= T; t += h) {
        tspan.push_back(t);
    }
}

```

```

std::vector<double> Y = Y0;
for (int i = 0; i < tspan.size(); ++i) {
    double t = tspan[i];

    std::vector<double> F = div(Y);

    Y[0] += h * F[0];
    Y[1] += h * F[1];
    Y[2] += h * F[2];
    Y[3] += h * F[3];
    Y[4] += h * F[4];
}

return Y;
}

double objective(const std::vector<double>& x) {
    std::vector<double> Y0(5);
    Y0[0] = x[0];
    Y0[1] = x[1];
    Y0[2] = x[2];
    Y0[3] = x[3];
    Y0[4] = x[4];

    std::vector<double> Y = ode45(Y0);

    double result = Y[0] - Y[1] + Y[2] - Y[3] + Y[4];
    return result;
}

double constraints(const std::vector<double>& x) {
    std::vector<double> Y0(5);
    Y0[0] = x[0];
    Y0[1] = x[1];
    Y0[2] = x[2];
    Y0[3] = x[3];
    Y0[4] = x[4];

    std::vector<double> Y = ode45(Y0);

    double Nk = Y.size() / 5;

    double result = Nk - kt;
    return result;
}

int main() {
    using namespace std;

    // Начальное приближение
    vector<double> x0 = {1, 1, 1, 1, 1};

```

```

// Пределы для переменных
vector<double> lb = {0, 0, 0, 0, 0};
vector<double> ub = {2, 2, 2, 2, 2};

// Создание объекта оптимизации
OptimLib::fmincon optimizer(objective, constraints);

// Установка начального приближения и границ переменных
optimizer.setStartPoint(x0);
optimizer.setBounds(lb, ub);

// Запуск оптимизации
vector<double> result = optimizer.optimize();

// Вывод результатов
cout << "Optimization Result:" << endl;
for (int i = 0; i < result.size(); ++i) {
    cout << "x[" << i << "] = " << result[i] << endl;
}

return 0;
}

#ifndef FMINCON_HPP
#define FMINCON_HPP

#include <vector>

namespace OptimLib {

    class fmincon {
    public:
        // Конструктор
        fmincon(double (*objective)(const std::vector<double>&),
                double (*constraints)(const
std::vector<double>&));

        // Установка начального приближения и границ переменных
        void setStartPoint(const std::vector<double>&
startPoint);
        void setBounds(const std::vector<double>& lb, const
std::vector<double>& ub);

        // Запуск оптимизации
        std::vector<double> optimize();

    private:
        double (*objective_)(const std::vector<double>&); //
Целевая функция
        double (*constraints_)(const std::vector<double>&); //
Функция ограничений

```

```

        std::vector<double> startPoint_;
Начальное приближение
        std::vector<double> lb_;
// Нижние границы переменных
        std::vector<double> ub_;
// Верхние границы переменных

        // Решение дифференциального уравнения
        std::vector<double> ode45(const std::vector<double>&
Y0);

        // Вычисление вектора производных
        std::vector<double> div(const std::vector<double>& Y);

        // Метод наискорейшего спуска
        std::vector<double> gradientDescent();

        // Проверка удовлетворения ограничений
        bool satisfyConstraints(const std::vector<double>& x);

        // Функция вычисления градиента целевой функции
        std::vector<double> computeGradient(const
std::vector<double>& x);

        // Методы для работы с ограничениями
        std::vector<double> computeConstraints(const
std::vector<double>& x);
        std::vector<double> computeConstraintsGradient(const
std::vector<double>& x);

    };

} // namespace OptimLib

#endif // FMINCON_HPP

#include "mainwindow.h"
#include "ui_mainwindow.h"

MainWindow::MainWindow(QWidget *parent)
    : QMainWindow(parent)
    , ui(new Ui::MainWindow)
{
    ui->setupUi(this);

    h = 0.1;
    xBegin = -3;
    xEnd = 3 + h;

    ui->widget->xAxis->setRange(0, 35);
    ui->widget->yAxis->setRange(0, 1);

    ui->widget_2->xAxis->setRange(0, 35);

```

```

ui->widget_2->yAxis->setRange(0, 1);

ui->widget_3->xAxis->setRange(0, 35);
ui->widget_3->yAxis->setRange(0, 1);

N = (xEnd - xBegin) / h + 2;

std::vector<double> init_dist{ 0.2, 0.0, 0.0, 0.6, 0.0 };
double time = 30.0;
std::vector<std::vector<double>> transition_matrix{ {0.0,
0.3, 0.0, 0.9, 0.0},
                                                    {0.0,
0.0, 0.3, 0.3, 0.0},
                                                    {0.1,
0.0, 0.7, 0.6, 0.0},
                                                    {0.0,
0.2, 0.1, 0.6, 0.4},
                                                    {0.7,
0.8, 0.0, 0.0, 0.0} };

int n = init_dist.size();
std::vector<std::vector<double>> P(time + 1,
std::vector<double>(n, 0));
P[0] = init_dist;
for (int i = 1; i <= time; i++) {
    x.push_back(i);
    for (int j = 0; j < n; j++) {
        double prob = 0;
        for (int k = 0; k < n; k++) {
            prob += P[i - 1][k] * transition_matrix[k][j];
        }
        P[i][j] = prob;
        y.push_back(P[i][j]);
    }
}

ui->widget->addGraph();
ui->widget->graph(0)->addData(x, y);
ui->widget->replot();

const double PI = 3.14159265358979323846;
const double kappa = 2.5; // параметр распределения Мизеса

// Функция вычисления матрицы распределения на основе Мизеса
double c = 1.0 / (2 * PI * sinh(kappa));
for (int i = 0; i < n; i++) {
    for (int j = 0; j < n; j++) {
        double x = cos(i - j);
        y1.push_back(c * exp(kappa * x));
    }
}

ui->widget_2->addGraph();
ui->widget_2->graph(0)->addData(x, y1);

```

```

    ui->widget_2->replot();

    // Задаем начальные условия
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            if (i == 0 || i == n - 1 || j == 0 || j == n - 1) {
                transition_matrix[i][j] = sin(PI * i * h) *
sin(PI * j * h);
            }
            else {
                transition_matrix[i][j] = 0.0;
            }
        }
    }

    // Метод конечных разностей
    for (int k = 0; k < 10000; k++) {
        for (int i = 1; i < n - 1; i++) {
            for (int j = 1; j < n - 1; j++) {
                transition_matrix[i][j] = 0.25 *
                (transition_matrix[i - 1][j] + transition_matrix[i + 1][j] +
                transition_matrix[i][j - 1] + transition_matrix[i][j + 1]);
                y2.push_back(transition_matrix[i][j]);
            }
        }
    }
    ui->widget_3->addGraph();
    ui->widget_3->graph(0)->addData(x, y2);
    ui->widget_3->replot();
}

MainWindow::~MainWindow()
{
    delete ui;
}

#ifndef QCUSTOMPLOT_H
#define QCUSTOMPLOT_H

#include <QtCore/qglobal.h>

// some Qt version/configuration dependent macros to include or
exclude certain code paths:
#ifdef QCUSTOMPLOT_USE_OPENGL
#   if QT_VERSION < QT_VERSION_CHECK(5, 0, 0)
#       define QCP_OPENGL_PBUFFER
#   else
#       define QCP_OPENGL_FBO
#   endif
#   if QT_VERSION >= QT_VERSION_CHECK(5, 3, 0)
#       define QCP_OPENGL_OFFSCREENSURFACE
#   endif

```

```

#endif

#if QT_VERSION >= QT_VERSION_CHECK(5, 4, 0)
#   define QCP_DEVICEPIXELRATIO_SUPPORTED
#   if QT_VERSION >= QT_VERSION_CHECK(5, 6, 0)
#       define QCP_DEVICEPIXELRATIO_FLOAT
#   endif
#endif

#include <QtCore/QObject>
#include <QtCore/QPointer>
#include <QtCore/QSharedPointer>
#include <QtCore/Qtimer>
#include <QtGui/QPainter>
#include <QtGui/QPainterPath>
#include <QtGui/QPaintEvent>
#include <QtGui/QMouseEvent>
#include <QtGui/QWheelEvent>
#include <QtGui/QPixmap>
#include <QtCore/QVector>
#include <QtCore/QString>
#include <QtCore/QDateTime>
#include <QtCore/QMultiMap>
#include <QtCore/QFlags>
#include <QtCore/QDebug>
#include <QtCore/QStack>
#include <QtCore/QCache>
#include <QtCore/QMargins>
#include <qmath.h>
#include <limits>
#include <algorithm>
#ifdef QCP_OPENGL_FBO
#   include <QtGui/QOpenGLContext>
#   if QT_VERSION < QT_VERSION_CHECK(6, 0, 0)
#       include <QtGui/QOpenGLFramebufferObject>
#   else
#       include <QOpenGLFramebufferObject>
#       include <QOpenGLPaintDevice>
#   endif
#   ifdef QCP_OPENGL_OFFSCREENSURFACE
#       include <QtGui/QOffscreenSurface>
#   else
#       include <QtGui/QWindow>
#   endif
#endif
#ifdef QCP_OPENGL_PBUFFER
#   include <QtOpenGL/QGLPixelBuffer>
#endif
#if QT_VERSION < QT_VERSION_CHECK(5, 0, 0)
#   include <qnumeric.h>
#   include <QtGui/QWidget>
#   include <QtGui/QPrinter>
#   include <QtGui/QPrintEngine>

```

```

#else
# include <QtNumeric>
# include <QtWidgets/QWidget>
# include <QtPrintSupport/QtPrintSupport>
#endif
#if QT_VERSION >= QT_VERSION_CHECK(4, 8, 0)
# include <QtCore/QElapsedTimer>
#endif
# if QT_VERSION >= QT_VERSION_CHECK(5, 2, 0)
# include <QtCore/QTimeZone>
#endif

class QCPPainter;
class QCustomPlot;
class QCPLayerable;
class QCPLayoutElement;
class QCPLayout;
class QCPAxis;
class QCPAxisRect;
class QCPAxisPainterPrivate;
class QCPAbstractPlottable;
class QCPGraph;
class QCPAbstractItem;
class QCPPlottableInterface1D;
class QCPLegend;
class QCPItemPosition;
class QCPLayer;
class QCPAbstractLegendItem;
class QCPSelectionRect;
class QCPColorMap;
class QCPColorScale;
class QCPBars;
class QCPPolarAxisRadial;
class QCPPolarAxisAngular;
class QCPPolarGrid;
class QCPPolarGraph;

/* including file 'src/global.h' */
/* modified 2022-11-06T12:45:57, size 18102 */

#define QCUSTOMPLOT_VERSION_STR "2.1.1"
#define QCUSTOMPLOT_VERSION 0x020101

// decl definitions for shared library compilation/usage:
#if defined(QT_STATIC_BUILD)
# define QCP_LIB_DECL
#elif defined(QCUSTOMPLOT_COMPILE_LIBRARY)
# define QCP_LIB_DECL Q_DECL_EXPORT
#elif defined(QCUSTOMPLOT_USE_LIBRARY)
# define QCP_LIB_DECL Q_DECL_IMPORT
#else
# define QCP_LIB_DECL
#endif

```

```

// define empty macro for Q_DECL_OVERRIDE if it doesn't exist
(Qt < 5)
#ifndef Q_DECL_OVERRIDE
# define Q_DECL_OVERRIDE
#endif

/*!
    The QCP Namespace contains general enums, QFlags and functions
    used throughout the QCustomPlot
    library.

    It provides QMetaObject-based reflection of its enums and
    flags via \a QCP::staticMetaObject.
*/

// Qt version < 6.2.0: to get metatypes Q_GADGET/Q_ENUMS/Q_FLAGS
in namespace we have to make it look like a class during moc-run
#if QT_VERSION >= 0x060200 // don't use QT_VERSION_CHECK here,
some moc versions don't understand it
namespace QCP {
    Q_NAMESPACE // this is how to add the staticMetaObject to
namespaces in newer Qt versions
#else // Qt version older than 6.2.0
# ifndef Q_MOC_RUN
namespace QCP {
# else // not in moc run
class QCP {
    Q_GADGET
    Q_ENUMS(ExportPen)
    Q_ENUMS(ResolutionUnit)
    Q_ENUMS(SignDomain)
    Q_ENUMS(MarginSide)
    Q_ENUMS(AntialiasedElement)
    Q_ENUMS(PlottingHint)
    Q_ENUMS(Interaction)
    Q_ENUMS(SelectionRectMode)
    Q_ENUMS(SelectionType)

    Q_FLAGS(AntialiasedElements)
    Q_FLAGS(PlottingHints)
    Q_FLAGS(MarginSides)
    Q_FLAGS(Interactions)
public:
# endif
#endif

/*!
    Defines the different units in which the image resolution can
    be specified in the export
    functions.

```

```

    \see QCustomPlot::savePng, QCustomPlot::saveJpg,
    QCustomPlot::saveBmp, QCustomPlot::saveRastered
    */
enum ResolutionUnit { ruDotsPerMeter          ///< Resolution is
given in dots per meter (dpm)
                    ,ruDotsPerCentimeter ///< Resolution is
given in dots per centimeter (dpcm)
                    ,ruDotsPerInch          ///< Resolution is
given in dots per inch (DPI/PPI)
                    };

    /*!
    Defines how cosmetic pens (pens with numerical width 0) are
    handled during export.

    \see QCustomPlot::savePdf
    */
enum ExportPen { epNoCosmetic          ///< Cosmetic pens are
converted to pens with pixel width 1 when exporting
                ,epAllowCosmetic ///< Cosmetic pens are
exported normally (e.g. in PDF exports, cosmetic pens always
appear as 1 pixel on screen, independent of viewer zoom level)
                };

    /*!
    Represents negative and positive sign domain, e.g. for passing
    to \ref
    QCPAbstractPlottable::getKeyRange and \ref
    QCPAbstractPlottable::getValueRange.

    This is primarily needed when working with logarithmic axis
    scales, since only one of the sign
    domains can be visible at a time.
    */
enum SignDomain { sdNegative          ///< The negative sign domain,
i.e. numbers smaller than zero
                ,sdBoth          ///< Both sign domains, including
zero, i.e. all numbers
                ,sdPositive ///< The positive sign domain,
i.e. numbers greater than zero
                };

    /*!
    Defines the sides of a rectangular entity to which margins can
    be applied.

    \see QCPLayoutElement::setAutoMargins,
    QCPAxisRect::setAutoMargins
    */
enum MarginSide { msLeft          = 0x01 ///< <tt>0x01</tt> left
margin
                ,msRight         = 0x02 ///< <tt>0x02</tt> right
margin

```

```

        ,msTop      = 0x04 ///< <tt>0x04</tt> top
margin
        ,msBottom   = 0x08 ///< <tt>0x08</tt> bottom
margin
        ,msAll      = 0xFF ///< <tt>0xFF</tt> all
margins
        ,msNone     = 0x00 ///< <tt>0x00</tt> no margin
    };
Q_DECLARE_FLAGS(MarginSides, MarginSide)

/ *!
    Defines what objects of a plot can be forcibly drawn
    antialiased/not antialiased. If an object is
    neither forcibly drawn antialiased nor forcibly drawn not
    antialiased, it is up to the respective
    element how it is drawn. Typically it provides a \a
    setAntialiased function for this.

    \c AntialiasedElements is a flag of or-combined elements of
    this enum type.

    \see QCustomPlot::setAntialiasedElements,
    QCustomPlot::setNotAntialiasedElements
*/
enum AntialiasedElement { aeAxes          = 0x0001 ///<
<tt>0x0001</tt> Axis base line and tick marks
                        ,aeGrid            = 0x0002 ///<
<tt>0x0002</tt> Grid lines
                        ,aeSubGrid         = 0x0004 ///<
<tt>0x0004</tt> Sub grid lines
                        ,aeLegend          = 0x0008 ///<
<tt>0x0008</tt> Legend box
                        ,aeLegendItems     = 0x0010 ///<
<tt>0x0010</tt> Legend items
                        ,aePlottables      = 0x0020 ///<
<tt>0x0020</tt> Main lines of plottables
                        ,aeItems           = 0x0040 ///<
<tt>0x0040</tt> Main lines of items
                        ,aeScatters        = 0x0080 ///<
<tt>0x0080</tt> Scatter symbols of plottables (excluding scatter
symbols of type ssPixmap)
                        ,aeFills           = 0x0100 ///<
<tt>0x0100</tt> Borders of fills (e.g. under or between graphs)
                        ,aeZeroLine        = 0x0200 ///<
<tt>0x0200</tt> Zero-lines, see \ref QCPGrid::setZeroLinePen
                        ,aeOther           = 0x8000 ///<
<tt>0x8000</tt> Other elements that don't fit into any of the
existing categories
                        ,aeAll              = 0xFFFF ///<
<tt>0xFFFF</tt> All elements
                        ,aeNone            = 0x0000 ///<
<tt>0x0000</tt> No elements
    };

```

```

Q_DECLARE_FLAGS(AntialiasedElements, AntialiasedElement)

/*!
    Defines plotting hints that control various aspects of the
    quality and speed of plotting.

    \see QCustomPlot::setPlottingHints
*/
enum PlottingHint { phNone          = 0x000 ///<
<tt>0x000</tt> No hints are set
                    ,phFastPolylines = 0x001 ///<
<tt>0x001</tt> Graph/Curve lines are drawn with a faster method.
This reduces the quality especially of the line segment
                    ///<
joins, thus is most effective for pen sizes larger than 1. It is
only used for solid line pens.
                    ,phImmediateRefresh = 0x002 ///<
<tt>0x002</tt> causes an immediate repaint() instead of a soft
update() when QCustomPlot::replot() is called with parameter
\ref QCustomPlot::rpRefreshHint.
                    ///<
This is set by default to prevent the plot from freezing on fast
consecutive replots (e.g. user drags ranges with mouse).
                    ,phCacheLabels      = 0x004 ///<
<tt>0x004</tt> axis (tick) labels will be cached as pixmaps,
increasing replot performance.
};
Q_DECLARE_FLAGS(PlottingHints, PlottingHint)

/*!
    Defines the mouse interactions possible with QCustomPlot.

    \c Interactions is a flag of or-combined elements of this enum
    type.

    \see QCustomPlot::setInteractions
*/
enum Interaction { iNone          = 0x000 ///<
<tt>0x000</tt> None of the interactions are possible
                    ,iRangeDrag    = 0x001 ///<
<tt>0x001</tt> Axis ranges are draggable (see \ref
QCPAxisRect::setRangeDrag, \ref QCPAxisRect::setRangeDragAxes)
                    ,iRangeZoom    = 0x002 ///<
<tt>0x002</tt> Axis ranges are zoomable with the mouse wheel
(see \ref QCPAxisRect::setRangeZoom, \ref
QCPAxisRect::setRangeZoomAxes)
                    ,iMultiSelect  = 0x004 ///<
<tt>0x004</tt> The user can select multiple objects by holding
the modifier set by \ref QCustomPlot::setMultiSelectModifier
while clicking
                    ,iSelectPlottables = 0x008 ///<
<tt>0x008</tt> Plottables are selectable (e.g. graphs, curves,
bars,... see QCPAbstractPlottable)

```

```

        ,iSelectAxes          = 0x010 ///  

<tt>0x010</tt> Axes are selectable (or parts of them, see  

QCPAxis::setSelectableParts)  

        ,iSelectLegend       = 0x020 ///  

<tt>0x020</tt> Legends are selectable (or their child items, see  

QCPLegend::setSelectableParts)  

        ,iSelectItems        = 0x040 ///  

<tt>0x040</tt> Items are selectable (Rectangles, Arrows,  

Textitems, etc. see \ref QCPAbstractItem)  

        ,iSelectOther        = 0x080 ///  

<tt>0x080</tt> All other objects are selectable (e.g. your own  

derived layerables, other layout elements,...)  

        ,iSelectPlottablesBeyondAxisRect = 0x100 ///  

<tt>0x100</tt> When performing plottable selection/hit tests,  

this flag extends the sensitive area beyond the axis rect  

    };  

Q_DECLARE_FLAGS(Interactions, Interaction)

```

```

//Histogram-----
-----
/*QBarSet *halfMarkProc = new QBarSet("Расчёт через
переходный полумарковский процесс");
QBarSet *markProc = new QBarSet("Расчёт через финальные
вероятности вложений цели Маркова");

double y1Max = *std::max_element(y1.begin(), y1.end());
double y2Max = *std::max_element(y2.begin(), y2.end());
double y3Max = *std::max_element(y3.begin(), y3.end());
double y4Max = *std::max_element(y4.begin(), y4.end());
double y5Max = *std::max_element(y5.begin(), y5.end());

double y1_mMax = *std::max_element(y1_m.begin(),
y1_m.end());
double y2_mMax = *std::max_element(y2_m.begin(),
y2_m.end());
double y3_mMax = *std::max_element(y3_m.begin(),
y3_m.end());
double y4_mMax = *std::max_element(y4_m.begin(),
y4_m.end());
double y5_mMax = *std::max_element(y5_m.begin(),
y5_m.end());

*halfMarkProc << y1Max << y2Max << y3Max << y4Max << y5Max;
*markProc << y1_mMax << y2_mMax << y3_mMax << y4_mMax <<
y5_mMax;

QBarSeries *series = new QBarSeries();
series->append(halfMarkProc);
series->append(markProc);

QChart *chart = new QChart();

```

```

    chart->addSeries(series);
    chart->setTitle("Стационарные полумарковские вероятности состояний");
    chart->setAnimationOptions(QChart::SeriesAnimations);

    QStringList categories;
    categories << "p1(t)" << "p2(t)" << "p3(t)" << "p4(t)" <<
    "p5(t)";
    QBarCategoryAxis *axisX = new QBarCategoryAxis();
    axisX->append(categories);
    chart->addAxis(axisX, Qt::AlignBottom);
    series->attachAxis(axisX);

    QValueAxis *axisY = new QValueAxis();
    axisY->setRange(0,1);
    chart->addAxis(axisY, Qt::AlignLeft);
    series->attachAxis(axisY);

    chart->legend()->setVisible(true);
    chart->legend()->setAlignment(Qt::AlignBottom);

    QChartView *chartView = new QChartView(chart);
    chartView->setRenderHint(QPainter::Antialiasing);

    setCentralWidget(chartView);
    resize(800, 400);

    QList<QBarSet *> sets = series->barSets();
    float currentHue = 0.0;
    for(int i = 0; i < sets.size(); ++i)
    {
        QColor col = QColor::fromHslF(currentHue, 0.7, 0.5);
        currentHue += 0.618033988749895f;
        currentHue = std::fmod(currentHue, 1.0f);
        sets[i]->setColor(col);
    }*/

}

/*double MainWindow::interpolate(std::vector<double> x,
std::vector<double> y, double xi) {
    int n = x.size();

    for (int i = 0; i < n - 1; i++) {
        if (xi >= x[i] && xi <= x[i + 1]) {
            double slope = (y[i + 1] - y[i]) / (x[i + 1] -
x[i]);
            double intercept = y[i] - slope * x[i];
            return slope * xi + intercept;
        }
    }
}*/

MainWindow::~MainWindow()

```

```

{
    delete ui;
}

#ifndef MAINWINDOW_H
#define MAINWINDOW_H

#include <QMainWindow>

QT_BEGIN_NAMESPACE
namespace Ui { class MainWindow; }
QT_END_NAMESPACE

class MainWindow : public QMainWindow
{
    Q_OBJECT

public:
    MainWindow(QWidget *parent = nullptr);
    ~MainWindow();

    //double interpolate(std::vector<double> x,
    std::vector<double> y, double xi);

private:
    Ui::MainWindow *ui;
};
#endif // MAINWINDOW_H

```