МИНОБРНАУКИ РОССИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
"Национальный исследовательский ядерный университет МИФИ"
НИЯУ МИФИ

Факультет управления и экономики высоких технологий

Кафедра стратегического планирования и методологии управления

УДК 681.58

«ДОПУЩЕНА К ЗАЩИТЕ»

Зам. зав.
кафедрой _____(Королёв А.С.)

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА БАКАЛАВРА

Разработка базы логических программ интеллектуального видеонаблюдения за действиями персонала АЭС

Руководители ВКР:	Доцент, к.т.н.	_____	(Королёв А.С.)
	К.ф.-м.н.	_____	(Морозов А.А.)
Исполнитель:	студентка гр. У08-82	_____	(Булычева А.А.)

Москва – 2016

МИНОБРНАУКИ РОССИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
"Национальный исследовательский ядерный университет МИФИ"

НИЯУ МИФИ

Факультет управления и экономики высоких технологий

Кафедра стратегического планирования и методологии управления

УТВЕРЖДАЮ	СОГЛАСОВАНО
Декан факультета «У» _____ А.В. Путилов	Зам. зав. кафедрой _____ А.С. Королёв
«____» _____201__г.	«____» _____201__г.

ЗАДАНИЕ
на выполнение выпускной квалификационной работы

Студент: Булычева А.А.
Шифр:
Направление: Системный анализ и управление
Группа: У08-82

1. Тема выпускной квалификационной работы

Разработка базы логических программ интеллектуального видеонаблюдения за действиями персонала АЭС

2. Техническое задание на выполнение выпускной квалификационной работы

1. Исследовать предметную область и выработать перечень задач интеллектуального видеонаблюдения за действиями персонала АЭС.
2. Провести экспериментальное исследование методов низкоуровневого анализа видеоизображений в Акторном Прологе, в том числе, метода вычитания фона и метода оценки скорости движения объектов.
3. Подготовить экспериментальные видеоклипы для исследования методов низкоуровневого анализа видео в Акторном Прологе, в том числе, примеры поведения «ходьба», «бег», «человек заснул на рабочем месте», «человек покинул рабочее место», а также примеры аномального поведения сотрудников АЭС.

4. Составить перечень типовых алгоритмов логического анализа видео, полезных для интеллектуального видеонаблюдения за действиями персонала АЭС.

3. Планируемые результаты выполнения выпускной квалификационной работы

1. Экспериментальные видеоклипы для исследования алгоритмов интеллектуального видеонаблюдения.
2. Примеры алгоритмов логического анализа видео.

4. Этапы выполнения выпускной квалификационной работы

№ этапа	Содержание этапа работы	Результаты выполнения этапа	Срок выполнения
1	Исследование предметной области и выработка перечня задач интеллектуального видеонаблюдения за действиями персонала АЭС	Перечень задач интеллектуального видеонаблюдения за действиями персонала АЭС	03.06.2016
2	Подготовка экспериментальных видеоклипов для исследования методов низкоуровневого анализа видео в Акторном Прологе	Набор экспериментальных видеоклипов для исследования алгоритмов интеллектуального видеонаблюдения	09.06.2016
3	Экспериментальное исследование методов низкоуровневого анализа видеоизображений в Акторном Прологе, в том числе, метода вычитания фона и метода оценки скорости движения объектов	Результаты статистической оценки качества работы алгоритмов низкоуровневого анализа изображений Акторного Пролога	17.06.2016
4	Выработка перечня типовых алгоритмов логического анализа видео, полезных для интеллектуального видеонаблюдения за действиями персонала АЭС	Перечень типовых алгоритмов логического анализа видео с примерами логических программ	27.06.2016

5. Перечень разрабатываемых документов и графических материалов

1. Экспериментальные видеоклипы.
2. Примеры алгоритмов логического анализа видео.

6. Руководители и консультанты выпускной квалификационной работы

Функциональные обязанности	**Должность в НИЯУ МИФИ**	**Ф.И.О.**	**Подпись**
Руководитель выпускной работы	Заместитель заведующего кафедрой № 82	Королёв Антон Сергеевич	
Руководитель выпускной работы	К.ф.-м.н., с.н.с. ИРЭ им. В.А. Котельникова РАН	Морозов Алексей Александрович	

7. Мониторинг выполнения выпускной квалификационной работы

Этап, №	Дата представления результатов руководителю и консультантам	Отметка руководителя и консультантов о выполнении работы
1		
2		
3		
4		

Задание выдал
Руководитель _____

«___»_____201__г.

Задание принял к исполнению

Студент _____

«___»_____201__г.

Список сокращений

АЭС – Атомная электростанция

ВКР – Выпускная квалификационная работа

ИВ – Интеллектуальное видеонаблюдение

ИРТ – Исследовательский реактор типовой

МФ – Медианная фильтрация

Реферат

Пояснительная записка представлена на 79 страницах, состоит из 3 частей и включает в себя 22 рисунка, 3 таблицы, 9 источников, 3 приложения.

Тема: Разработка базы логических программ интеллектуального видеонаблюдения за действиями персонала АЭС.

Объектом исследования и разработки являются алгоритмы интеллектуального видеонаблюдения, реализованные средствами объектно-ориентированного логического языка Акторный Пролог и распознающие аномальное поведение персонала АЭС.

Цель работы – повышение уровня безопасности за счёт уменьшения влияния человеческого фактора в процессе контроля выполнения техники безопасности сотрудниками АЭС.

В процессе работы была исследована предметная область, выработан перечень задач интеллектуального видеонаблюдения за действиями персонала АЭС, рассмотрены существующие системы интеллектуального видеонаблюдения, целью которых является повышение уровня безопасности на АЭС. Также было проведено экспериментальное исследование методов низкоуровневого анализа видеоизображений в Акторном Прологе, в том числе, метода вычитания фона и метода оценки скорости движения объектов, для последнего был проведён анализ зависимости точности оценки от различных параметров. Были подготовлены экспериментальные видеоклипы для исследования методов низкоуровневого анализа видео в Акторном Прологе, а также видеоклипы, иллюстрирующие аномальное поведение сотрудников АЭС, снятые на основе выработанного списка ситуаций аномального поведения. Как итог, был составлен перечень типовых алгоритмов логического анализа видео, полезных для интеллектуального видеонаблюдения за действиями персонала АЭС.

При тестировании разработанных программ, распознающих аномальное поведение сотрудников АЭС, не было выявлено недостатков, препятствующих корректному видеоанализу.

Оглавление

Введение

Интеллектуальное видеонаблюдение и распознавание человеческой деятельности является быстроразвивающейся областью исследований, которая находит своё применение во многих сферах, с том числе и в обеспечении безопасности людей. Атомные электростанции являются объектами с потенциально высоким уровнем угрозы безопасности. Поэтому внедрение систем интеллектуального видеонаблюдения на АЭС, позволяющих распознавать аномальное поведение и опасные ситуации и подающих сигнал тревоги оператору, является рациональным и целесообразным способом предупреждения опасной обстановки.

Логическое программирование является перспективным подходом для анализа ситуаций на видеоизображении. Идея такого подхода состоит в том, чтобы использовать логические правила для описания и анализа последовательности видеоизображений. Общая схема такого подхода заключается в использовании двух уровней обработки видеоизображений – анализа низкого уровня, реализуемого, как правило, на обычных императивных языках программирования, и анализа высокого уровня, реализуемого на логическом языке.

В данной работе используется объектно-ориентированный логический язык программирования Акторный Пролог для видеоанализа и Java для трансляции программ. Акторный Пролог воплощает новый подход к объединению логического и объектно-ориентированного программирования, обладающий следующими достоинствами:

• В основе подхода лежит использование классической логики (логики предикатов первого порядка).

• Центральной идеей и сущностью подхода является обнаружение и устранение логических противоречий, возникающих в процессе взаимодействия объектов.

• Разработанный подход позволил математически корректным образом ввести в логический язык разрушающее присваивание и параллельные процессы.

Таким образом, объектно-ориентированные средства Акторного Пролога позволяют разделять программу на параллельные взаимодействующие процессы, которые осуществляют различные стадии обработки видеоизображений, в то время как Java обеспечивает необходимую надёжность, мобильность и открытость программного обеспечения интеллектуального видеонаблюдения и позволяет использовать стандартные библиотеки для низкоуровневой обработки видеоизображений. В общем случае, программа состоит из нескольких параллельных процессов. Например, один процесс может считывать последовательность кадров видеоизображения, что позволяет имитировать ввод видео в реальном времени, и осуществляет низкоуровневую обработку. Второй процесс при этом может анализировать информацию, которую дал на выходе первый процесс, и выдавать результаты анализа на экран.

Цель данной работы – повышение уровня безопасности за счёт уменьшения влияния человеческого фактора в процессе контроля выполнения техники безопасности сотрудниками АЭС.

Новизна исследования, проведённого в данной работе, заключается в том, что видеоизображение анализируется автоматически в реальном времени с помощью объектно-ориентированного логического языка. В настоящее время на АЭС используются системы видеонаблюдения, осуществляющие простой мониторинг и архивную запись происходящих событий. Разработанная база логических программ интеллектуального видеонаблюдения позволит не только записывать видео, но и оповещать оператора о тревожных и опасных ситуациях. Хотя в настоящее время ведутся исследования по созданию таких систем, автором настоящей работы не найдено информации о том, что в разработанных программах реализованы

оба уровня обработки (в найденных работах осуществлён лишь низкоуровневый анализ), в то время как в данной работе реализована и высокоуровневая обработка. Более того, использование объектно-ориентированного логического языка программирования является совершенно новым подходом для решения данной задачи.

Актуальность исследований, проведённых в работе, обусловлена следующими факторами и обстоятельствами:

- Постоянно возрастающее использование систем видеонаблюдения в различных сферах деятельности.
- Необходимость предотвращения опасных ситуаций, связанных с нарушением техники безопасности и должностных инструкций на АЭС.
- Перспективность создания систем интеллектуального видеонаблюдения для обеспечения безопасности и для оптимизации процессов жизненного цикла предприятия.

1. Изучение предметной области и постановка задачи интеллектуального видеонаблюдения

1.1. Существующие методы и системы интеллектуального видеонаблюдения для повышения уровня безопасности на АЭС

Надёжность при использовании первых систем видеонаблюдения в высшей степени зависела от человека-оператора во время контроля и мониторинга различных процессов. Такие пассивные системы наблюдения являлись неэффективными. Поэтому исследования, которые проводятся в последнее время по разработке систем видеонаблюдения, фокусируют своё внимание на интеллектуальных техниках, осуществляющих автоматическую обработку видеоизображений и обеспечивающих широкую область наблюдения. Такие системы интеллектуального видеонаблюдения (ИВ) позволяют снизить стоимость обслуживания и зависимость от человеческого фактора.

Обнаружение объектов, трекинг (слежение за объектами), а также интерпретация и понимание действий объектов в областях интереса являются активно развивающимися областями исследований на протяжении последнего десятилетия. ИВ предполагает автоматический визуальный контроль процессов, включающий анализ и интерпретацию поведения объектов, обнаружение и слежение за объектами для понимания событий, происходящих в видеосцене. Основная задача ИВ заключается в объяснении видеосцены и возможности контролировать множество процессов.

В исследованиях по обнаружению движения в ИВ активно изучается повышение точности вычитания фона, так как вычитание фона является первым шагом в обработке видеоизображений, и от аккуратности этих вычислений зависит корректность последующего анализа. В частности, многие исследования посвящены вычитанию фона в сложных условиях, таких как изменение погодных условий или движущийся фон. В настоящее время существует три основных подхода, реализующих вычитание фона:

вероятностный, статистический и подход, использующий шаблоны [1]. В вероятностном подходе используются методы из теории вероятности, такие как смесь Гауссовых моделей и оценка плотности вероятности. Во втором подходе для вычитания фона используются такие характеристики, как медиана, среднее значение и дисперсия фона. Третий подход фокусируется на методах нахождения определённого шаблона фона. Такие методы нацелены на извлечение параметров фона, полученных из последовательности видеокадров, и нахождение в них определённых закономерностей.

В исследованиях задачи слежения за объектами (трекинг объектов) можно выделить три основных подхода: слежение за точечными особенностями сцены, методы сопровождения компонент и методы сопровождения контура [1]. В первом подходе используется метод статистической фильтрации для оценки состояния интересующего объекта. Фильтр Калмана и фильтр частиц являются наиболее популярными методами фильтрации. В подходе сопровождения компонент (под компонентой понимается форма объекта) используются различные оценочные методы для определения принадлежности соответствующих областей интересующему объекту. Наиболее известными методами в данном подходе являются методы сдвига среднего и фильтр частиц. Подход сопровождения контура позволяет прогнозировать положение контура на следующем кадре. В данном подходе выделяют два метода: первый состоит в использовании моделей пространства состояний, второй – в минимизации функции энергии контура с использованием таких техник, как градиентный спуск. В настоящее время в исследования, посвящённых слежению за объектом, изучается решение таких проблем, как окклюзия (частичное или полное перекрытие интересующего объекта), слежение за объектом в условиях изменяющегося освещения.

Для интерпретации и понимания действий человека проводятся исследования, решающие проблемы моделирования и оценки

местонахождения частей тела человека [1]. Данные исследования связаны между собой и позволяют автоматически распознавать активность человека. В исследованиях, посвящённых моделированию человеческого тела, обычно используются 2D-контуры и объёмные 3D-фигуры для представления всего человеческого тела или какой-либо отдельной его части. Локализация части тела выполняется методом слежения за ней. В исследованиях по оценке местонахождения части тела выделяют подход «сверху вниз» и подход «снизу вверх». Первый нацелен на реконструирование человеческого тела по входному изображению, второй отслеживает части тела, собирая их в модель человеческого тела. Широко используется комбинированный подход, позволяющий компенсировать недостатки каждого метода.

Интеллектуальное видеонаблюдение анализирует видеоизображения для обнаружения аномальных или необычных активностей. Конечная цель системы ИВ – автоматическая генерация тревоги, чтобы помочь оператору онлайн или контролю событий офлайн. В Институте по Атомной энергии в Португалии (IEN, CNEN) исследовалась задача повышения безопасности работников во время эксплуатации и технического обслуживания АЭС при помощи ИВ [2]. В данном исследовании камера была установлена в помещении с повышенной радиационной опасностью. Человек, находящийся в этой комнате, обнаруживался и отслеживался в течение некоторого времени. Далее, анализируя полученный трек и карту позволенных доз радиации в этой комнате, система могла оценить дозу, полученную человеком, пока он находился в комнате. Данное исследование является частью масштабного НИОКР, посвящённого виртуальному моделированию ядерных установок, разрабатываемых в настоящее время в IEN. Данная работа сконцентрирована на стадии сегментации (отделении переднего плана от заднего) на заранее определённом промежутке времени. Осуществление стадий обнаружения и трекинга планируется реализовать в будущем.

Примеры обработки видеоизображений в рамках проекта представлены на рисунке 1.1.

Рисунок 1.1. Примеры обработки видеоизображений в рамках проекта, осуществляемого в Институте по Атомной энергии (Португалия).

Аналогичные задачи повышения уровня безопасности на АЭС решаются в Институте плазменных исследований (Индия), где ведётся разработка системы ИВ для АЭС, конечными целями которой являются предотвращение несанкционированного доступа к ядерным материалам и установкам и своевременное обнаружение аномального поведения людей с последующей генерацией сигнала тревоги [3]. Данная система в конечном итоге предусматривает анализ по четырём основным ступеням: обнаружение движущихся объектов, классификация обнаруженных объектов, трекинг и распознавание аномальной активности на основе проведённого анализа. В данной работе речь идёт о стадии обнаружения движущегося объекта. Алгоритм обнаружения движущегося человека включает в себя следующие этапы: преобразование видеоклипа в последовательность кадров, использование фильтра Гаусса для сглаживания и подавления шумов, вычитание переднего плана, выделение блобов, трекинг движущихся объектов. В настоящее время реализовано вычитание переднего плана с использованием фильтра Гаусса для подавления шумов (результаты представлены на рисунке 1.2). Исследование продолжается, и результаты последующей обработки и анализа будут опубликованы в будущем.

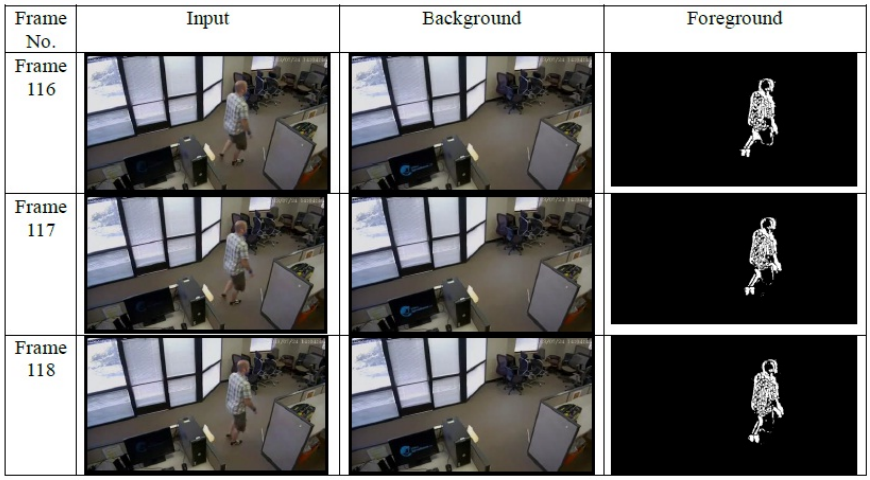

Рисунок 1.2. Обнаружение движущегося объекта в исследовании, проведённом в Институте плазменных исследований (Индия).

Кроме того, для повышения уровня надёжности АЭС, используются так называемые роботы контроля. Эти роботы должны двигаться по территории АЭС согласно заданному маршруту, наблюдая за статическими и динамическими характеристиками окружающей среды. Например, старший научный сотрудник Национального института современной промышленной технологии и науки (AIST, Япония) разработал компактную визуальную стереосистему, оборудованную воронкообразным прибором для сбора видеоинформации с больших областей, установленную на мобильном роботе [4]. Данная система изображена на рисунке 1.3. Разработка такой системы позволила обеспечить безопасное передвижение мобильного робота и высокую разрешающую способность видеоизображений для более точного видеонаблюдения.

Рисунок 1.3. Компактная визуальная стереосистема, разработанная в Национальном институте современной промышленной технологии и науки (AIST, Япония).

Таким образом, в исследованиях, посвящённых разработке систем ИВ для повышения уровня безопасности на АЭС, в настоящее время осуществлена только часть низкоуровневого анализа видеоизображений (вычитание фона и обнаружение объектов). В настоящей выпускной квалификационной работе в контексте решения задачи повышения уровня безопасности на АЭС помимо низкоуровневого анализа будет также рассмотрена обработка высокого уровня. Реализация такой обработки видеоизображений позволит распознавать аномальное поведение персонала АЭС в конкретных ситуациях, которые смоделированы на основе опыта эксплуатации и правил техники безопасности, действующих на исследовательском реакторе ИРТ МИФИ.

1.2. Классификация разновидностей аномального поведения персонала АЭС

Наиболее важным этапом в создании базы логических программ интеллектуального видеонаблюдения является выявление и классификация типов аномального поведения персонала АЭС, которые целесообразно распознавать в автоматическом режиме. Классификация разновидностей аномального поведения персонала необходима также для составления сценариев видеопримеров для экспериментального исследования методов интеллектуального видеонаблюдения. В качестве примера объекта с

повышенными требованиями к радиационной безопасности был выбран исследовательский реактор, расположенный на территории НИЯУ МИФИ. Реактор ИРТ (исследовательский реактор типовой) Атомного центра МИФИ (ИРТ МИФИ), сооружённый по типовому проекту ТП-3304М, относится к универсальным исследовательским ядерным реакторам средней мощности. Главный инженер Портнов Александр Алексеевич, начальник службы по эксплуатации реактора Яковлев Леонид Иванович, начальник службы по обеспечению и контролю радиационной безопасности Савкин Владимир Алексеевич и инженер первой категории Карцев Константин Павлович провели экскурсию и рассказали о правилах поведения на реакторе ИРТ. Кроме того, были получены регламентирующие документы и инструкции по технике безопасности в электронном виде.

После изучения и анализа полученной документации, был выделен ряд ситуаций, когда поведение сотрудника АЭС можно рассматривать в качестве аномального. Подробное описание ситуаций приведено ниже.

1. В помещениях комплекса реактора запрещено оставлять дозиметр вблизи источников ионизирующих излучений и в любом другом месте, кроме стенда обмена дозиметров [5]. Помимо этого, в стерильных помещениях запрещено класть на пол различные предметы (сумки, ящики и т. п.), если они не находятся в специальном стерильном и герметичном чехле. Нарушение данных правил может лечь в основу ситуации, которую можно условно назвать «Оставленный предмет».
2. Пультовая комната относится к зоне строгого режима, кроме того, персоналу запрещается оставлять данное помещение пустым или без присмотра во время работы реактора. Следовательно, можно рассмотреть две ситуации: а) Сотрудник покинул пультовую в рабочее время – ситуация «Пустая пультовая комната»;

б) Сотрудник уснул или потерял сознание на рабочем месте в пультовой – ситуация «Пультовая комната без присмотра».

3. Существует правило, обязывающее во всех помещениях комплекса реактора ИРТ всех сотрудников носить спецодежду, включающую белый халат, шапочку и тапочки [5]. Очевидно, что такая ситуация как «Сотрудник одет не в белый халат» свидетельствует о нарушении данного правила и может быть включена в базу логических программ, распознающих аномальное поведение персонала.
4. В инструкции по радиационной безопасности указано, что при работе с источниками излучений должна предусматриваться такая мера, как ограничение длительности пребывания людей вблизи источников ионизирующих излучений [5]. Отсюда вытекает, что ситуация «Сотрудник превысил время пребывания вблизи источника ионизирующего излучения» иллюстрирует нарушение вышеописанного правила.
5. При работе с частичной разгерметизацией насосной комнаты первого контура (данное правило действует во время проведения радиационно-опасных работ) должно быть обеспечено присутствие в помещении только персонала, непосредственно занятого ремонтными операциями [5]. Как правило, всегда известно, каким количеством человек ограничивается команда, проводящая ремонтные операции. Соответственно, ситуация «Количество сотрудников в насосной первого контура при проведении радиационно-опасных работ превышено» может рассматриваться в качестве аномального поведения сотрудников и может быть включена в базу логических программ.
6. Персонал обязан проходить предварительный медосмотр при поступлении на работу и периодические ежегодные медицинские осмотры, включающие консультации невропатолога и психолога [5].

Несмотря на это, нельзя исключать такую ситуацию как «Драка между сотрудниками», которая сама по себе является ярким примером аномального поведения людей. Вдобавок к этому, такая ситуация как «Сотрудник потерял сознание и упал» (имеется в виду, что сотрудник упал, не встаёт и не двигается в течение продолжительного времени) также является наглядным примером аномального поведения, и поэтому тоже может быть включена в базу логических программ, распознающих аномальное поведение людей.

В связи с вышеизложенным, в настоящей работе решено разработать логические программы, распознающие аномальное поведение людей для следующих ситуаций:

- «Сотрудник уснул или потерял сознание на рабочем месте в пультовой».
- «Сотрудник покинул пультовую комнату в рабочее время».
- «Оставленный предмет».
- «Сотрудник потерял сознание и упал».
- «Драка между сотрудниками».

2. Экспериментальное исследование методов низкоуровневого анализа видеоизображений

2.1. Подготовка видеопримеров и их низкоуровневая обработка

Для изучения метода интеллектуального видеонаблюдения аномального поведения людей, основанного на использовании объектно-ориентированного логического языка программирования Акторный Пролог, было снято несколько видеоклипов. Видеоклипы снимались на камеру модели SONY HANDYCAM HDR-XR 350 7.1 MPx, с частотой кадров равной 25, разрешение видео составляло 1280x720 px. Первоначально эксперименты проводились на компьютере модели ASUS с процессором Intel Core i3-2350M CPU 2.3 GHz и операционной системой Windows 10. В дальнейшем эксперименты проводились на более мощном компьютере модели MacPro с процессором Intel Xeon CPU E5645 @2.40 GHz 2.53 GHz (2 процессора) 64-разрядная ОС, установленная память 18.0 ГБ.

По итогам проведенных съёмок получено несколько серий видеоклипов:

- Видеоклипы на улице (первая серия).
- Видеоклипы в закрытом помещении (в работу не пошли).
- Видеоклипы в спортзале, где пол был без покрытия (третья серия).
- Видеоклипы в спортзале, где пол застелен зелёными матами (четвёртая серия).
- Видеоклипы в спортзале, где пол застелен чёрными матами (пятая серия).

Прежде чем приступить к рассмотрению результатов проведённых экспериментов, необходимо описать методы низкоуровневой обработки, реализованные в используемом методе интеллектуального видеонаблюдения.

2.1.1. Описание принципов низкоуровневого анализа

Все алгоритмы низкоуровневого анализа реализованы на языке Java во встроенном классе Акторного Пролога *'ImageSubtractor'* [6]. Ниже описаны основные этапы низкоуровневого анализа.

- Считывание фона. Класс *'ImageSubtractor'* получает последовательность видеоизображений и вычисляет математическое ожидание и дисперсию значений каждого пикселя на видеокадрах. Если разность между величиной яркости пикселя и вычисленным математическим ожиданием отличается на значение, которое больше заданного порога (порог задаётся количеством среднеквадратичных отклонений), то эти пиксели рассматриваются как элементы переднего плана. Значения разности и математического ожидания могут изменяться и становиться более точными во время выполнения программы или на основе определённого количества первых кадров. Для уменьшения цифрового шума видеоизображения обрабатываются фильтром Гаусса.
- Распознавание блобов. На этой стадии используется алгоритм, который создает прямоугольные блобы. Идея состоит в том, что пиксели, являющиеся передним планом, заключаются в рамки (прямоугольники). Все соприкасающиеся между собой рамки в свою очередь также обводятся в новые, большие по размеру рамки. Этот процесс продолжается до тех пор, пока все пиксели объектов на переднем плане не будут обведены в непересекающиеся рамки. Вычисленные таким образом прямоугольники и объявляются блобами.
- Создание треков. Треки – это траектории движения блобов. Если прямоугольные блобы пересекаются в соседних кадрах, то считается, что это один и тот же объект, который просто переместился. Если есть несколько пересекающихся блобов, то за продолжение трека берётся тот блоб, площадь пересечения с которым больше. Также принято, что блоб, для которого не найден пересекающийся блоб, будет ещё находиться в памяти некоторое

определённое количество времени. Если в течение этого времени не будет найдено продолжение трека, то трек будет считаться законченным. Все законченные треки некоторое время хранятся в памяти, а затем стираются. Кроме того, треки, длина которых меньше определённого значения, рассматриваются как ошибочные и сразу отбрасываются. Во время построения треков сохраняются точки их пересечения, необходимые для построения графов связей между блобами, а также для определения скорости блобов (для определения скорости блобов выбираются такие области треков, у которых нет пересечений с другими треками).

- Определение скорости блобов. Для определения скорости блоба на основе обратной матрицы проективных преобразований вычисляются физические координаты четырёх углов прямоугольного блоба [7]. Числовое дифференцирование координат даёт грубую оценку скорости для каждого угла блоба. Предполагается, что хотя бы одна точка (один угол блоба) находится на уровне пола. Тогда для других углов блоба оценки скорости могут оказаться завышенными, так как камера находится выше уровня пола, и верхняя часть человека визуально соответствует более далёким точкам. Основываясь на этом предположении, в качестве нижней границы скорости человека берётся наименьшая скоростная оценка из четырёх. Для повышения стабильности работы алгоритма оценки скорости последовательность значений координат углов прямоугольного блоба, последовательность промежуточных оценок скорости (отдельно по горизонтальной и вертикальной осям), а также последовательность окончательных значений скорости подвергаются медианной фильтрации.

- Построение связных графов движения прямоугольных блобов. Каждый граф включает в себя все треки, которые имеют точки пересечения в некоторые моменты времени. При необходимости алгоритм позволяет исключать треки неподвижных и медленно движущихся объектов.

2.1.2. Анализ видео, снятого на открытом пространстве

Для изучения и проверки корректности методов низкоуровневого анализа видеоизображений был снят видеоклип на улице (серия видеоклипов 1). Видео снималось при ярком солнечном свете на фоне здания. В кадр попадали движущиеся тени от деревьев и окружающих предметов, что повлияло на качество низкоуровневого анализа. Примечательной особенностью данного видео является тот факт, что камера несколько раз дёргалась из-за порывов ветра. Пример кадра из видеоклипа представлен на рисунке 2.1.

Рисунок 2.1. Пример кадра из видео серии 1.

Для того чтобы проанализировать видеоклип в логической программе, видеоклип необходимо представить в виде последовательности кадров. Представление видеоролика в виде последовательности кадров осуществлялось с помощью программы Free Video to GPG Converter, с помощью которой из видео формата MP4 были получены кадры формата JPEG. Логическая программа, реализующая метод, загружает последовательность кадров и воспроизводит видео в режиме реального времени.

При анализе данного видеоклипа с помощью программы, реализованной на логическом языке программирования Акторный Пролог, проводилось несколько экспериментов по проверке метода вычитания фона. Рассмотрим подробнее суть экспериментов и их результаты.

1) В первую очередь изучался предикат *get_background_image*, возвращающий результаты усреднения кадров. Использовался режим отбрасывания цвета видеокадров перед усреднением. В итоге получается неподвижная (чёрно-белая) картинка (см. рисунок 2.2). В связи с тем, что в некоторый момент времени камера дёрнулась, алгоритм вычитания фона сработал неправильно, поэтому картинка получилась смазанной (рисунок 2.2б).

а)  б) 

Рисунок 2.2. Применение предиката get_background_image.

2) Затем рассматривался предикат *get_sigma_image*, показывающий распределение дисперсии (а точнее, стандартного отклонения) яркости пикселей по экрану. В местах, где происходит значительное изменение яркости, значение дисперсии большое, и, следовательно, на изображении появляются белые полосы и пятна, а там, где яркость меняется незначительно – видны тёмные пятна. В результате того, что камера дёрнулась, распределение дисперсии по экрану сдвинулось, вследствие чего картинка раздвоилась (рисунок 2.3б).

а)  б) 

Рисунок 2.3. Применение предиката get_sigma_image.

3) Также исследовался предикат *get_foreground_image*, позволяющий выделить объекты переднего плана. Согласно описанию предиката, статические объекты он распознаёт как задний план. Таким образом, если изображение неподвижно, в результате получается картинка однотонного цвета (в нашем случае, серого). Однако, как видно из рисунка 2.4а, алгоритм распознаёт статичные объекты как передний план (какие-то элементы здания и тени в данном случае), хотя предполагается, что он должен воспринимать их как объекты заднего плана. Причиной этого является то, что в момент, когда штатив с камерой дёрнулся, алгоритм вычитания фона выделил практически все присутствующие в кадре объекты в качестве переднего плана, что, очевидно, является ошибкой (см. рисунок 2.4б).

а) 

б) 

Рисунок 2.4. Применение предиката get_foreground_image.

4) Ради эксперимента была проверена работа предиката *get_blobs*, который возвращает найденные программой блобы. Результаты работы алгоритма выявления блобов приведены на рисунке 2.5. На рисунке видно, что алгоритм даёт сбой и выделяет в качестве блобов элементы здания, деревьев и теней.

Рисунок 2.5. Результаты распознавания блобов.

Таким образом, можно сделать вывод, что на данном этапе разработки метода выявления аномального поведения людей не предусмотрена работа с движущимися тенями. Изученные методы работают неправильно для первой серии видео, что обусловлено движением теней в кадре и неустойчивым положением штатива с камерой.

Программный код прилагается к настоящей дипломной работе на электронном носителе (диск CD-ROM).

2.1.3. Анализ видео, снятого в закрытом помещении

Во избежание проблем, выявленных при съёмках на улице, были проведены съёмки третьей серии видеоклипов (далее видео будет называться «видео серии 3»). Третья серия снималась в спортзале, где отсутствовали движущиеся тени и посторонние предметы. Штатив с камерой были хорошо закреплены, статичны и установлены на втором этаже (балконе) помещения. Пример кадров из видео представлен на рисунке 2.6.

Рисунок 2.6. Пример кадров из видео серии 3.

Для данных видеоклипов проверялся алгоритм вычитания фона, а также изучалась ещё одна стадия обработки низкого уровня – вычисление скорости движущегося объекта. С этой целью были сняты два видеоклипа, на первом из которых человек проходит спокойным шагом, а на втором – пробегает.

Для определения скорости блоба необходимо предварительно вычислить и записать в программу обратную матрицу проективных преобразований. Она необходима для сопоставления дистанций в реальном мире (сколько человек пробежал в метрах) с дистанциями на экране (сколько человек пробежал в пикселях). Для того чтобы вычислить матрицу, необходимо разметить помещение, то есть выбрать не менее четырёх точек на полу снимаемой сцены, и отметить их точные физические координаты (начало и система координат выбираются произвольно). Далее необходимо отметить точные логические (в пикселях) координаты этих же самых точек. На видео серии 3 было отмечено 10 точек (они выделялись жёлтыми квадратиками на полу, рисунок 2.6), первая из которых является началом координат. Матрица вычислялась с помощью программы, написанной на языке Matlab. Результаты вычисления матрицы представлены на рисунке 2.7.

```
Projective transformation matrix:
 -0.2799   0.2119  -0.0005
 -0.6202   0.0187  -0.0000
646.1375 149.5157   0.6290
Inverse matrix of projective transformation:
   0.1057   -1.6064    0.0001
   3.0386    0.9820    0.0023
-830.8996 1416.8348    1.0000
```

Рисунок 2.7. Вычисление матрицы с помощью программы на языке Matlab.

Для проверки правильности выбора координат и расчёта обратной матрицы проективных преобразований, обратная матрица была применена к графическому изображению «видео серии 3». В нашем случае, в качестве результата осуществления этой операции ожидалось получение вида сверху на пол с искажёнными стенами. Как видно из рисунка 2.8, координаты выбраны верно, и, соответственно, проекция отобразила правильный вид сверху на площадку. Круг отобразился в виде эллипса из-за сжатия изображения, стены исказились.

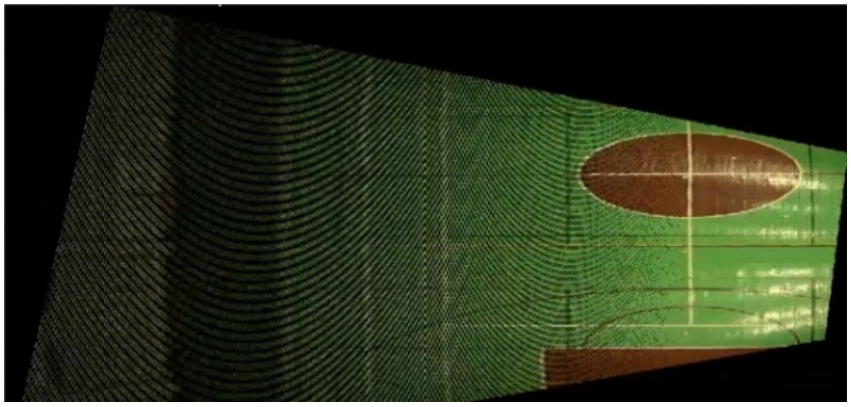

Рисунок 2.8. Результат применения обратной матрицы проективных преобразований для видео серии 3.

Чтобы программа смогла обнаружить пробегающего человека в качестве объекта переднего плана, следует проводить усреднение фона по определённому количеству первых кадров. Обычно фон усредняется непрерывно, то есть по всем кадрам. Усреднение происходит вычислением среднего значения каждого пикселя на всех кадрах. В случае обнаружения

бегущего человека, в результате непрерывного усреднения изображение человека будет по ошибке усреднено и частично сольётся с фоном. В связи с этим в Акторном Прологе реализован метод, позволяющий усреднять фон по заданному количеству первых кадров, после чего усреднение прекращается.

С целью усреднения фона по заданному количеству первых кадров используется атрибут класса *'ImageSubtractor' maximal_training_interval*. В видео серии 3 усреднение производилось по первым 100 кадрам. Так как камера оставалась статична, и в помещении не было движения теней от посторонних предметов и т. п., алгоритм усреднения фона сработал корректно: алгоритм выделил идущего и пробегающего человека в качестве объектов переднего плана, на переднем плане присутствовало минимальное количество посторонних пятен. Пятна появляются вследствие цифровых шумов (то есть изменения случайным образом яркости пикселей на видеоизображении), которые свойственны камере во время съёмок. Результат представлен на рисунке 2.9.

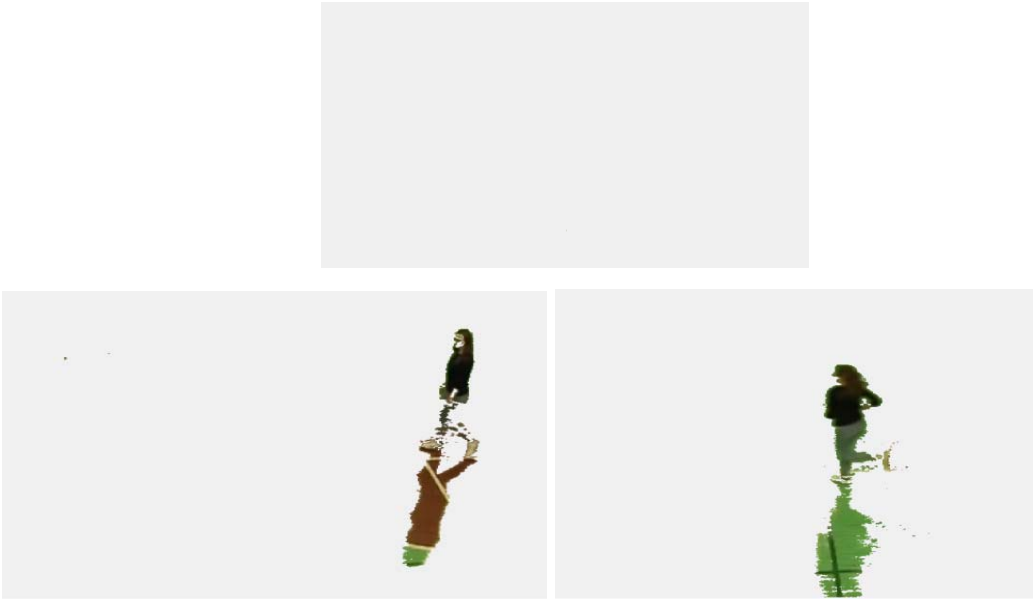

Рисунок 2.9. Усреднение фона.

На данном этапе была выявлена проблема отражения человека в полу на видеоизображении (см. рисунок 2.9). Это связано с лакированным покрытием пола и хорошим освещением. В связи с этим, предикат

draw_blobs (данный предикат рисует блобы) нарисовал блобы в два раза больше, чем следует (так как произошёл видеозахват самого человека и его отражения), что и видно на рисунке 2.10.

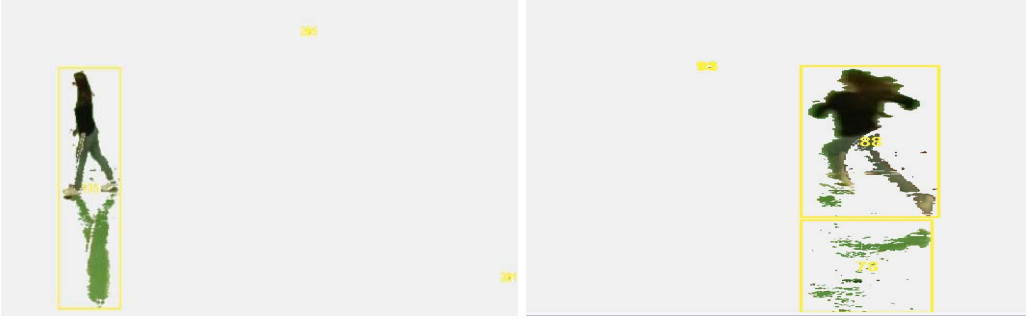

Рисунок 2.10. Выделение блобов.

В связи с тем, что выделение блобов сработало неверно, алгоритм, строящий треки и вычисляющий скорость, тоже сработал некорректно.

Программа, в которой реализован алгоритм вычисления скорости, выводит на экран в правом верхнем углу легенду, в которой разными цветами выделены различные диапазоны скорости, и далее рисует трек цветом, соответствующим определённой скорости в легенде. Результаты построения трека и вычисления скорости представлены на рисунке 2.11.

а)

б)

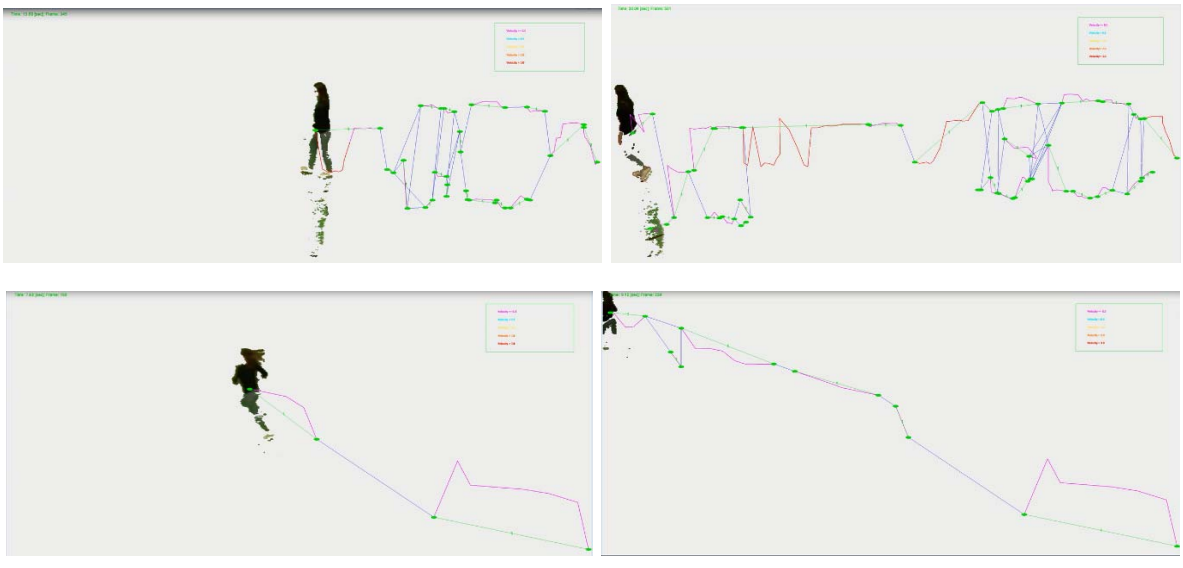

Рисунок 2.11. Построение трека и вычисление скорости.

Из рисунка 2.11 видно постоянное изменение формы блоба, потому что алгоритм выделения блобов то включает отражение человека в состав блоба, то нет. В некоторых случаях алгоритм ошибочно включает в состав блоба отражение человека, но не его самого. В связи с этим трек был построен в форме «пилы», что более заметно на видеоклипе, где человек идёт спокойным шагом (рисунок 2.11а). На видеоклипе с пробегающим человеком (рисунок 2.11б) трек выглядит правдоподобнее из-за меньшего отражения. Так как трек был построен неверно, скорость тоже была вычислена некорректно.

Программный код, реализующий данную обработку, прилагается к настоящей дипломной работе на электронном носителе (диск CD-ROM).

С целью нивелировать перечисленные проблемы, для съёмок последующих серий видео были созданы условия, снижающие риски некорректной работы алгоритмов. В связи с этим пол был покрыт матами, предотвращающими появление отражения. Примеры кадров представлены на рисунке 2.12.

Рисунок 2.12. Кадры из видеоклипов серий 4 и 5.

В серии видеоклипов 4 пол был покрыт зелёными матами. Затем, в серии видеоклипов 5, было принято решение использовать чёрные маты, чтобы уменьшить влияние теней на работу алгоритма вычитания фона.

В серии видеоклипов 5 также была вычислена обратная матрица проективных преобразований для определения скорости (координаты точек и вычисленные матрицы для всех версий видео представлены в приложении Б).

Результаты действия обратной матрицы проективных преобразований на видео представлены на рисунке 2.13.

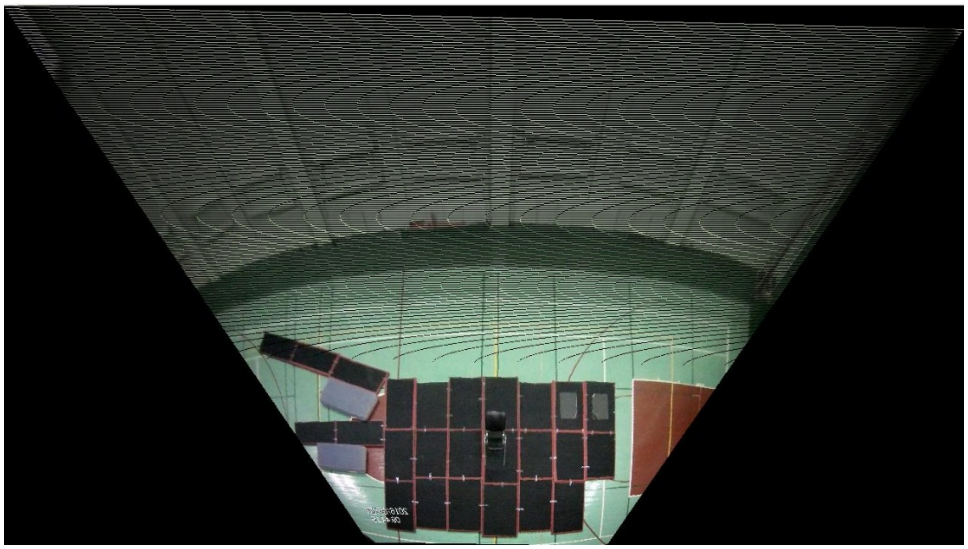

Рисунок 2.13. Действие обратной матрицы проективных преобразований на видео серии 5.

Для того чтобы понять, при каких условиях программа выдаёт скорость, наиболее приближенную к реальной, снималось несколько видеороликов:

- Во-первых, снимались отдельные видеоклипы для определения скорости пробегающего и проходящего спокойным шагом человека. Кроме того, были сняты видеоклипы, на которых человек половину пути проходит спокойным шагом, оставшуюся – пробегает.
- Во-вторых, на снятых видеороликах человек проходит и пробегает сначала параллельно, потом по диагонали относительно плоскости объектива камеры.
- В-третьих, пол был застелен сначала зелёными матами, потом – чёрными.

Также в логической программе были заданы различные параметры для обработки видеоизображений:

- Скорости в логической программе вычислялись для кадров с высоким разрешением (1280x720 px, разрешение камеры) и для кадров с низким разрешением (640x480 px, разрешение было уменьшено при помощи программы «Киностудия Windows 2012»).
- В логической программе скорость вычислялась с использованием медианной фильтрации, а также без неё. Медианная фильтрация является эффективным способом подавления шумов в сигналах, поэтому использование такой фильтрации для сглаживания последовательности значений скорости движения объекта также представляет определённый интерес. В связи с этим использовались атрибуты класса *'ImageSubtractor'* *apply_median_filtering_to_velocity='yes'*, *velocity_median_filter_halfwidth=3*, включающие медианную фильтрацию значений скорости с шириной окна 7 кадров.
- При обработке видеоизображений с низким разрешением усреднение фона происходило непрерывно, что позволило наибольшим образом сгладить цифровые шумы, в то время как при обработке кадров с высоким разрешением усреднение фона производилось по первым 100 кадрам (атрибут *minimal_training_interval*), так как для кадров с высоким разрешением непрерывное усреднение фона усредняло и самого человека, если он останавливался, что мешало правильно построить трек.
- Изменялся параметр *minimal_blob_size*, задающий минимальный размер блоба. Если этот параметр слишком маленький, на экране появляются ошибочные блобы, если слишком большой – люди перестают захватываться. По этой причине для обработки видеоизображений с низким разрешением данный параметр был равен 1000, для видеоизображений с высоким разрешением – 5000.

- Изменялся порог выделения переднего плана *background_standard_deviation_factor*. Данный параметр измеряется в стандартных отклонениях (сигмах) значений яркости пикселей. Если порог слишком низкий, на переднем плане остаются шумы, особенно от теней и отражений. Если слишком высокий – люди не попадают на передний план. Ввиду этого для обработки видеоизображений с низким разрешением данный параметр был равен 1,7, для видеоизображений с высоким разрешением – 1,2.
- Минимальная длина трека *minimal_track_duration* была равна 2. На некоторых видеороликах человек пробегал настолько быстро, что трек состоял всего из нескольких точек, и для того, чтобы программа воспринимала этот трек и не теряла видеозахват, задавалась минимальная длина трека, равная 2.

Результаты вычисления скорости логической программой были занесены в таблицу и далее сравнивались по различным параметрам, а также с реальной скоростью. Скорость, с которой человек на самом деле пробегал, равна V= 3,8 м/с, с которой шёл спокойным шагом – V= 1,1 м/с. Анализ проводился по следующим параметрам:

- Медианная фильтрация скорости. Анализировалось, когда логическая программа точнее вычисляет скорость – с использованием медианной фильтрации или без неё.
- Направление относительно плоскости объектива камеры. Проверялось, насколько точнее будет вычисляться скорость в зависимости от того, в каком направлении относительно плоскости объектива камеры проходит или пробегает человек (параллельно или по диагонали).
- Разрешение видеоизображений. Исследовалось, для какого разрешения кадров (низкого или высокого) скорость вычислялась наиболее точно.

Были получены следующие результаты (последняя строка в каждой приведённой таблице является средним значением по столбцу):

- По параметру «медианная фильтрация скорости». В зависимости от того, была ли включена медианная фильтрация скорости, были получены результаты, отображённые в таблице 2.1.

Таблица 2.1 – Результаты вычисления скорости, проанализированные по параметру «Медианная фильтрация». Результаты представлены в относительных единицах «отношение вычисленной скорости к реальной».

Бег		**Шаг**	
МФ	**Без МФ**	**МФ**	**Без МФ**
1,45	1,18	1,27	1,36
0,89	0,92	0,55	0,55
0,58	0,5	0,45	0,55
0,37	0,37	0,73	0,82
0,89	0,97	0,55	0,95
0,76	0,68	0,18	0,55
0,24	0,29	0,18	0,45
0,27	0,34	0,18	0,36
0,68	0,66	0,51	0,69

Очевидно, что максимально близким к реальной скорости считается значение, наиболее приближённое к единице (эти значения выделены зелёным цветом). Таким образом, можно сделать вывод, что как в случае пробегающего человека, так и в случае проходящего спокойным шагом человека, наиболее точно скорость была вычислена в программе, не использующей медианную фильтрацию скорости. В качестве объяснения этому факту можно предложить гипотезу, что одновременно с удалением выбросов (ошибочных значений большой величины) медианный фильтр

удаляет также и экстремумы в последовательности значений скорости. Экстремумы возникают естественным образом, вследствие неравномерного характера движения человека, и удаление этих величин может приводить к заниженным оценкам средней скорости движения человека. Сравнение средних значений по столбцам показывает, что исследуемый параметр «Медианная фильтрация», в среднем, практически не влияет на результаты работы алгоритма оценки скорости движения людей.

- По параметру «Направление относительно плоскости объектива камеры». В зависимости от того, в каком направлении относительно плоскости объектива камеры проходил или пробегал человек, логической программой была вычислена скорость, результаты представлены в таблице 2.2.

Таблица 2.2 – Результаты вычисления скорости, проанализированные по параметру «Направление относительно плоскости объектива камеры». Результаты представлены в относительных единицах «отношение вычисленной скорости к реальной».

Бег		**Шаг**	
Параллельно	**По диагонали**	**Параллельно**	**По диагонали**
0,37	1,45	1,27	0,73
0,37	0,89	0,55	0,82
0,27	0,58	0,45	0,18
0,34	1,18	1,36	0,36
	0,92	0,55	
	0,5	0,55	
	0,89	0,55	
	0,97	0,95	
	0,76	0,18	
	0,68	0,18	
	0,24	0,45	
	0,29	0,55	
0,34	0,78	0,63	0,52

Как видно из таблицы, наиболее точные оценки скорости соответствуют диагональному направлению относительно плоскости объектива камеры в случае пробегающего человека и параллельному направлению в случае проходящего спокойным шагом человека. Такие неоднозначные результаты говорят о том, что данный фактор не имеет однозначного влияния на точность оценок скорости перемещения и разброс значений оценок, по всей видимости, носит случайный характер. Этот вывод подтверждается сравнением средних значений по столбцам.

- По параметру «Разрешение видеоизображений». В зависимости от разрешения видеоизображений были получены значения скоростей, представленные в таблице 2.3.

Таблица 2.3 – Результаты вычисления скорости, проанализированные по параметру «Разрешение видеоизображений». Результаты представлены в относительных единицах «отношение вычисленной скорости к реальной».

Бег		**Шаг**	
Высокое разрешение	**Низкое разрешение**	**Высокое разрешение**	**Низкое разрешение**
1,45	0,89	1,27	0,55
1,18	0,97	1,36	0,95
0,89	0,76	0,55	0,18
0,92	0,68	0,55	0,55
0,37	0,27	0,45	0,18
0,37	0,34	0,55	0,45
0,58	0,24	0,73	0,18
0,5	0,29	0,82	0,36
0,78	0,56	0,78	0,42

Как видно из таблицы, максимально приближенные к реальным значения скоростей получились для видеоизображений с низким разрешением, как для бегущего человека, так и для человека, идущего спокойным шагом. Это означает, что при обработке изображений большого разрешения производительности использованного компьютера всё-таки недостаточно для построения аккуратных треков движения, и это приводит к искажённым оценкам скорости. Интересно, что сравнение средних значений по столбцам, наоборот, показывает, что, в среднем, для видео с высоким разрешением оценки скорости оказались более правдоподобными. Такие

противоречивые результаты можно объяснить тем, что на точность оценки скорости влияют два противоположно направленных фактора. С одной стороны, алгоритм оценки скорости имеет тенденцию систематически занижать значения скорости, с другой, при высоком разрешении изображений компьютер не успевает обработать все поступающие кадры, что приводит к завышению оценок скорости движения. В результате случайного взаимодействия этих противоположных факторов, оценки скорости могут как ухудшаться, так и улучшаться.

Стоит отметить тот факт, что в видеороликах, где человек половину пути проходит спокойным шагом, а оставшуюся – пробегает, независимо от того, в каком направлении относительно плоскости объектива камеры двигался человек, вычисленные логической программой значения скоростей получились наиболее далёкими от реальных. Это связано с тем, что логическая программа не успевала корректно вычислить скорость, так как к тому моменту, когда человек разбегался и достигал максимальной скорости, он выходил из кадра.

Для пробегающего человека наиболее точная оценка скорости равняется V= 3,7 м/с (соответствует значению 0,97 в относительных единицах), для идущего спокойным шагом – V= 1,05 м/с (соответствует значению 0,95 в относительных единицах). Приблизительные значения скорости в обоих случаях были вычислены логической программой без использования медианной фильтрации и для видеоизображений с низким разрешением. Однако для идущего спокойным шагом человека наиболее точная оценка скорости была получена при условии, что человек проходит параллельно относительно плоскости объектива камеры, а для бегущего человека – по диагонали относительно плоскости объектива камеры. Вместе с тем, усреднённые значения скоростей, вычисленные по каждому параметру (см. последнюю строку в таблицах 1.1, 1.2, 1.3), оказались приблизительно одинаковыми. Это может говорить о том, что проведённого в настоящей

работе анализа недостаточно для заключения более определённых и достоверных выводов. С этой целью требуется более глубокий анализ, учитывающий другие факторы и параметры, влияющие на алгоритм оценки скорости, а также их взаимное влияние, однако это выходит за рамки данной работы.

Помимо вышеописанного анализа также была составлена общая таблица, в которой содержатся все вычисленные логической программой значения скоростей (таблица представлена в приложении А настоящей работы), и вычислено среднеквадратичное отклонение (сигма) для полученных значений. Среднеквадратичное отклонение характеризует меру рассеяния данных. В случае пробегающего человека сигма б=1,37, в случае идущего спокойным шагом – б=0,39. Также были вычислены средние значения скоростей для бегущего и проходящего человека. Для бегущего человека данная величина равна 2,55 м/с, для идущего спокойным шагом – 0,67 м/с. Данные значения средней скорости отличаются от реальной скорости не больше, чем на значение сигмы, из чего можно сделать вывод, что метод оценки скорости, реализованный в классе *'ImageSubtractor'* Акторного Пролога обеспечивает приемлемую точность оценки скорости передвижения людей.

3. База логических программ, распознающих аномальное поведение персонала АЭС

3.1. Описание высокоуровневого анализа

Общая концепция ИВ на основе объектно-ориентированного логического программирования заключается в использовании двух уровней обработки видеоизображений:

- Обработка низкого уровня заключается в вычитании фона, распознавании объектов в кадре, построении траекторий объектов, оценке скорости и т. д. Низкоуровневая обработка видеоизображений реализуется с помощью стандартных методов видеоанализа, использующих языки более низкого уровня (в данном случае, Java).
- Обработка высокого уровня берёт в качестве входных данных результаты низкоуровневой обработки и использует логические правила для распознавания тревожных ситуаций (примерами таких ситуаций могут служить падение человека или оставленный предмет).

Подход к ИВ на основе объектно-ориентированного логического программирования использует следующие принципы анализа и описания поведения объектов при высокоуровневой обработке видеоизображений:

- Во-первых, язык Пролог является декларативным, что позволяет сформулировать задачу распознавания аномального поведения в терминах анализа графов движения блобов. Однако, ввиду неоднозначности результатов низкоуровневого анализа и расплывчатости самого сценария аномального поведения, возникает необходимость введения элементов нечёткого логического вывода.
- Простейшие элементы нечёткого логического вывода могут быть введены в язык Пролог с помощью стандартных арифметических

предикатов языка. Для распознавания аномального поведения используются предикаты, учитывающие одновременно несколько характеристик блоба (например, для распознавания бегущего человека следует принимать во внимание такие характеристики блоба, как средняя скорость и длина пройденного пути). Объединение нескольких характеристик выполняется с помощью очень простой нечёткой метрики, описанной с помощью арифметических функций. С точки зрения декларативной семантики языка, процедура распознавания бегущего человека будет стандартной формулой логики предикатов первого порядка [6].

3.2. Описание логических программ, распознающих аномальное поведение персонала АЭС

В разделе 2.1 были выделены следующие ситуации, иллюстрирующие возможное аномальное поведение персонала АЭС:

1) «Сотрудник покинул пультовую комнату в рабочее время».
2) «Сотрудник уснул или потерял сознание на рабочем месте в пультовой».
3) «Сотрудник потерял сознание и упал».
4) «Оставленный предмет».
5) «Драка между сотрудниками».

Рассмотрим более подробно принципы высокоуровневого анализа видеоизображений на примере логических программ, распознающих аномальное поведение в перечисленных выше ситуациях.

Ситуация «Сотрудник покинул пультовую комнату в рабочее время»

В терминах логического языка, ситуация «Сотрудник покинул пультовую комнату в рабочее время» излагается следующим образом: «Человек находится в комнате некоторое время, после чего выходит и пропадает из зоны видеонаблюдения». В логической программе данную

ситуацию достаточно просто описать: если граф движения человека в какой-то момент времени пропал (а это может свидетельствовать о том, что человек вышел из зоны видеонаблюдения), то можно считать, что сотрудник покинул помещение. Данную проверку в логической программе осуществляет предикат *check_graphs*:

```
check_graphs([]):-!,
        report_target_event.
check_graphs(_).
report_target_event1:-
        graphic_window ? set_text_alignment('CENTER','CENTER'),
        graphic_window ? set_font({size:48}),
        graphic_window ? set_pen({color:'Red'}),
        graphic_window ? draw_text(
                0.5,0.2,
                "Внимание! Пультовая комната "
                "осталась без присмотра!").
```

Правило говорит о том, что если в кадре не найдено ни одного графа, то можно считать, что в комнате нет людей, и следует подать сигнал тревоги (предикат *report_target_event* выдаёт текстовое сообщение о том, что пультовая комната осталась без присмотра). Результат работы программы представлен на рисунке 3.1.

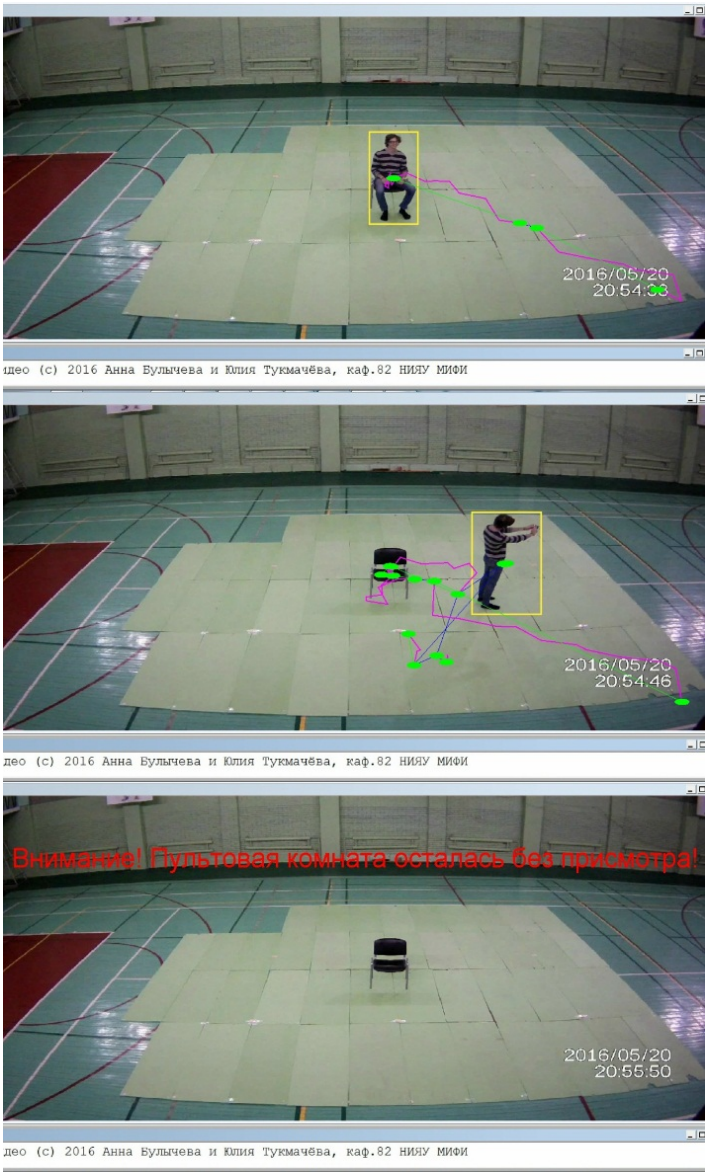

Рисунок 3.1. Программа ИВ распознала ситуацию «Сотрудник покинул пультовую комнату в рабочее время».

Программный код, реализующий данную обработку, прилагается к настоящей дипломной работе на электронном носителе (диск CD-ROM).

Ситуация «Сотрудник уснул на рабочем месте в пультовой»

Самым простым способом распознать уснувшего человека является применение того же алгоритма, что и в случае человека, покинувшего помещение. Это объясняется тем, что при непрерывном усреднении фона человек, который на протяжении продолжительного времени не шевелится, будет усреднён и сольётся с фоном, вследствие чего граф его движения

«растворится» и мы получим «пустую комнату». Результат такой обработки видео приведён на рисунке 3.2.

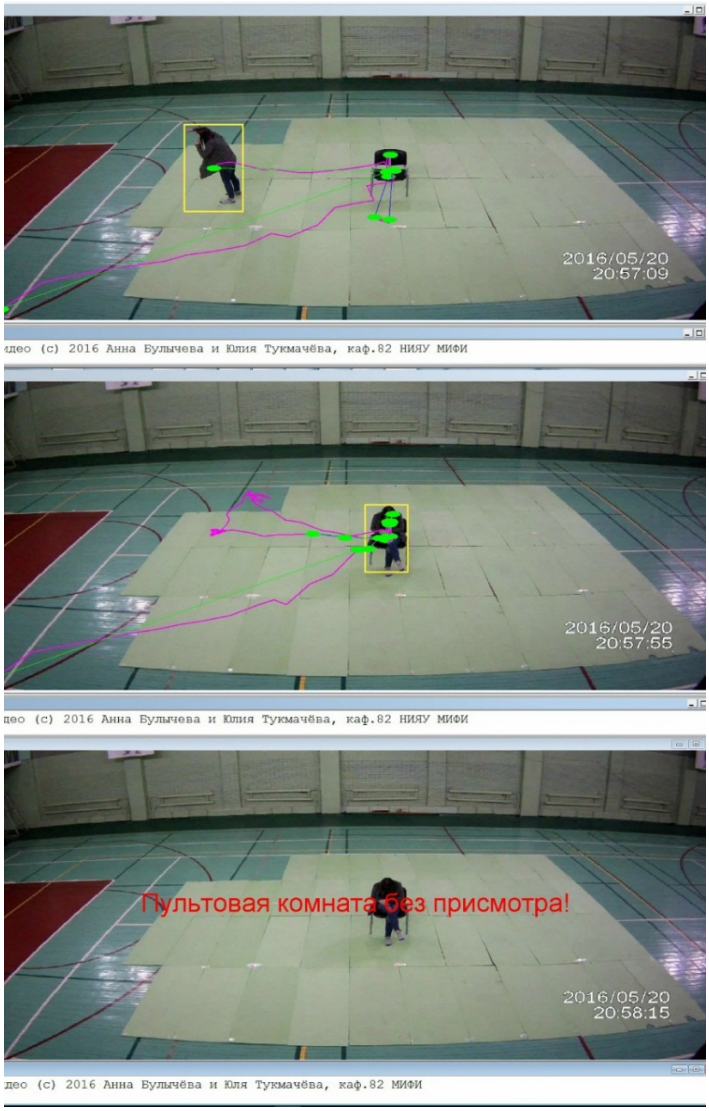

Рисунок 3.2. Программа ИВ распознала ситуацию «Сотрудник уснул на рабочем месте в пультовой».

Однако данное решение задачи распознавания уснувшего человека является не совсем корректным, так как человек может, например, периодически шевелить разными частями тела во сне. Такие движения, хотя и небольшие, приведут к тому, что программа не будет усреднять человека вместе с фоном, а соответственно, и не будет распознавать «пустую комнату». Поэтому правильнее сравнивать координаты блоба в разные моменты времени. Если координаты будут отличаться друг от друга меньше,

чем на заданный порог в течение определённого промежутка времени, значит, можно сделать вывод о том, что человек уснул.

Программный код, реализующий данную обработку, прилагается к настоящей дипломной работе на электронном носителе (диск CD-ROM).

Ситуация «Сотрудник потерял сознание и упал»

В первую очередь необходимо распознать падение человека, только после этого можно сделать вывод о том, что человек потерял сознание, если он лежит и не двигается в течение некоторого промежутка времени.

Предикат *check_graphs* используется с целью распознавания падения человека.

```
check_graphs([Graph|Rest],CurrentFrame):-!,
        check_graph(Graph,CurrentFrame),
        check_graphs(Rest,CurrentFrame).
check_graphs(_,_).
check_graph([Edge|_],CurrentFrame):-
        Edge == {coordinates:TrackOfBlob|_},
        there_is_a_lying_person(TrackOfBlob,CurrentFrame),!,
        report_target_event.
check_graph([_|Rest],CurrentFrame):-!,
        check_graph(Rest,CurrentFrame).
check_graph(_,_).
there_is_a_lying_person([Blob],CurrentFrame):-
        Blob == {frame:CurrentFrame,width:W,height:H|_},
        Ratio== W / H,
        Ratio > 1.5,!.
there_is_a_lying_person([_|Rest],CurrentFrame):-
        there_is_a_lying_person(Rest,CurrentFrame).
```

```
report_target_event:-
        graphic_window ? set_text_alignment('CENTER','CENTER'),
        graphic_window ? set_font({size:48,weight:'WEIGHT_BOLD'}),
        graphic_window ? set_pen({color:'Red'}),
        graphic_window ? draw_text(
                0.5,0.25,
                "Сотрудник АЭС упал!").
```

Предикат *check_graph* рекурсивно обрабатывает все дуги, входящие в состав графа. Аналогично, в каждой дуге рекурсивно обрабатываются все блобы. Далее, для каждого блоба проверяется такая характеристика, как отношение ширины прямоугольного блоба к высоте. Если это отношение больше значения, равного 1,5 (то есть, если блоб был вертикальным прямоугольником, после чего приобрёл форму горизонтального прямоугольника), логическая программа делает вывод о том, что сотрудник упал, и выдаёт текстовое сообщение. Результат выполнения программы представлен на рисунке 3.3.

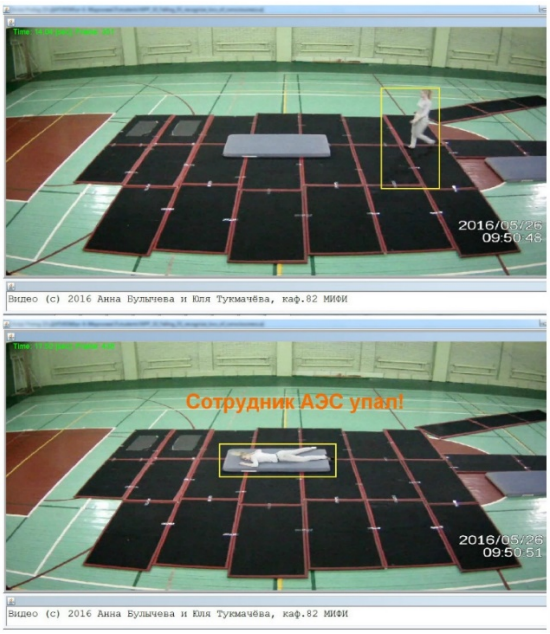

Рисунок 3.3. Программа ИВ распознала ситуацию «Сотрудник упал».

Программный код, реализующий данную обработку, прилагается к настоящей дипломной работе на электронном носителе (диск CD-ROM).

Для заключения о том, что человек упал и потерял сознание, логической программе достаточно установить, что сотрудник некоторое время лежал неподвижно. В данном примере было выбрано время, равное трём секундам. Программный код, распознающий падение человека и потерю сознания, приведён ниже.

```
check_graphs([Graph|Rest],CFN):-
        check_graph(Graph,CFN),!,
        check_graphs(Rest,CFN).
check_graphs(_,_).
check_graph([Edge|_],CurrentFrame):-
        Edge == {coordinates:TrackOfBlob|_},
        there_is_a_lying_person(
                TrackOfBlob,'no',0,
                FirstFrame,
                LastFrame),
        LastFrame == CurrentFrame,!,
        Duration== (LastFrame - FirstFrame) / sampling_rate,
        report_target_event(Duration).
check_graph([_|Rest],CFN):-!,
        check_graph(Rest,CFN).
check_graph(_,_).
there_is_a_lying_person([Blob],'no',_,LF,LF):-
        this_is_a_lying_person(Blob,LF),!.
there_is_a_lying_person([Blob],'yes',FF,FF,LF):-
        this_is_a_lying_person(Blob,LF),!.
```

```
there_is_a_lying_person([Blob|Rest],'no',_,FF2,LF):-
        this_is_a_lying_person(Blob,FF1),!,
        there_is_a_lying_person(Rest,'yes',FF1,FF2,LF).
there_is_a_lying_person([Blob|Rest],'yes',FF1,FF2,LF):-
        this_is_a_lying_person(Blob,_),!,
        there_is_a_lying_person(Rest,'yes',FF1,FF2,LF).
there_is_a_lying_person([_|Rest],IsLying,FF1,FF2,LF):-
        there_is_a_lying_person(Rest,IsLying,FF1,FF2,LF).
this_is_a_lying_person(Blob,N):-
        Blob == {width:W,height:H,frame:N|_},
        Ratio== W / H,
        Ratio > 1.5.
report_target_event(Duration):-
        Duration > 3.0,!,
        graphic_window ? set_text_alignment('CENTER','CENTER'),
        graphic_window ? set_font({size:48,weight:'WEIGHT_BOLD'}),
        graphic_window ? set_pen({color:'Red'}),
        graphic_window ? draw_text(
                0.5,0.25,
                "Сотрудник АЭС потерял сознание!!!").
report_target_event(_):-
        graphic_window ? set_text_alignment('CENTER','CENTER'),
        graphic_window ? set_font({size:48,weight:'WEIGHT_BOLD'}),
        graphic_window ? set_pen({color:'Orange'}),
        graphic_window ? draw_text(
                0.5,0.25,
                "Сотрудник АЭС упал!").
```

Программа работает следующим образом. Так же как и в первом случае, рекурсивно обрабатываются все графы движения блобов, найденные

программой. В каждом графе рекурсивно обрабатываются все дуги, входящие в его состав. В каждой дуге рекурсивно обрабатывается список найденных блобов. При выполнении правила *check_graph* вызывается правило *there_is_a_lying_person*, в котором в качестве аргументов передаются: список блобов *TrackOfBlob*, принадлежащих дуге; переменная *IsLying*, имеющая одно из двух значений (*'yes'* или *'no'*); номер первого кадра, на котором человек уже лежит, в качестве входного аргумента; номер первого кадра, на котором человек уже лежит, в качестве выходного аргумента; номер последнего кадра, на котором человек лежит. В теле правила *there_is_a_lying_person* вызывается предикат *this_is_a_lying_person*, регистрирующий факт падения человека (при помощи оценки отношения ширины прямоугольного блоба к высоте). Программа содержит пять правил с названием *there_is_a_lying_person* и начинает выполнение предиката с пятого правила, так как в момент времени, когда человек появляется в кадре, это единственное правило, которое успешно пройдёт унификацию аргументов в заголовке, и в этот момент ещё не установлено, что человек упал. Это правило будет выполняться до тех пор, пока программа не зарегистрирует падение. Как только это произошло, программа исполнит третье правило, где переменная *IsLying* примет значение *'yes'*, и перейдёт к выполнению четвёртого правила, которое продолжит разматывать рекурсивно список найденных блобов и проверять, лежит ли всё ещё человек. Первое и второе правило выполняются в том случае, когда в списке найденных блобов остался всего один элемент. В тот момент, когда программа распознала падение человека, она выдаёт текстовое сообщение о том, что сотрудник упал. Если человек в течение трёх секунд продолжает лежать (*Duration== (LastFrame - FirstFrame) / sampling_rate* в правиле *check_graph* и условие *Duration > 3.0* в правиле *report_target_event(Duration)*), программа выдаёт сообщение о том, что сотрудник потерял сознание. Результат работы программы приведён на рисунке 3.4.

Рисунок 3.4. Программа ИВ распознала ситуацию «Сотрудник упал и потерял сознание».

Программный код, реализующий данную обработку, прилагается к настоящей дипломной работе на электронном носителе (диск CD-ROM), а также представлен в приложении В настоящей работы.

Ситуация «Оставленный предмет»

Задача распознавания оставленного предмета сводится к использованию информации о скорости и стандартизированной площади блоба. Стандартизированная площадь используется с целью оценки размера

объекта в кадре (так как один и тот же объект может сильно отличаться по размерам в пикселях в зависимости от того, как далеко он находится от камеры) и равно отношению площади прямоугольного блоба к квадрату характеристической длины [8]. Характеристическая длина является одним из свойств блоба и имеет разное значение на разных участках сцены (если объект расположен близко к камере, она имеет большое значение, если далеко – маленькое). Зная значение характеристической длины, можно оценить размер объекта: если значение маленькое – объект находится далеко от камеры, если большое – близко. Таким образом, стандартизированная площадь блоба является величиной, которая характеризует его размер и практически не зависит от удалённости объекта от камеры.

Поиск оставленного предмета осуществляется следующим образом:

```
check_graphs([Graph|Rest],CurrentFrame):-!,
        check_graph(Graph,CurrentFrame),
        check_graphs(Rest,CurrentFrame).
check_graphs(_,_).
check_graph([Edge|_],CurrentFrame):-
        Edge == {coordinates:TrackOfBlob,mean_velocity:V|_},
        there_is_abandoned_object(TrackOfBlob,CurrentFrame,V),!,
        report_target_event.
check_graph([_|Rest],CurrentFrame):-!,
        check_graph(Rest,CurrentFrame).
check_graph(_,_).
there_is_abandoned_object([Blob],CurrentFrame,V):-
        Blob == {       frame:CurrentFrame,
                        width:W,
                        height:H,
                        characteristic_length:CL|_},
```

```
        RectangleArea== W * H,
        StandardizedArea== RectangleArea / (CL*CL),
        M1== 1 - ?fuzzy_metrics(StandardizedArea,0.5,0.2),
        M2== 1 - ?fuzzy_metrics(V,0.05,0.02),
        M1 * M2 >= 0.25,!.
there_is_abandoned_object([_|Rest],CurrentFrame,V):-
        there_is_abandoned_object(Rest,CurrentFrame,V).
fuzzy_metrics(X,T,H) = 1.0 :-
        X >= T + H,!.
fuzzy_metrics(X,T,H) = 0.0 :-
        X <= T - H,!.
fuzzy_metrics(X,T,H) = V :-
        V== (X-T+H) * (1 / (2*H)).
report_target_event:-
        graphic_window ? set_text_alignment('CENTER','CENTER'),
        graphic_window ? set_font({size:48,weight:'WEIGHT_BOLD'}),
        graphic_window ? set_pen({color:'Red'}),
        graphic_window ? draw_text(
                0.5,0.25,
                "Нарушена стерильность помещения!").
```

При рекурсивной обработке графов движения блобов, проверяется каждая дуга. В каждой дуге проверяется список найденных блобов, где осуществляется оценка стандартизированной площади и скорости блоба при помощи функции *fuzzy_metrics.* Данная функция проверяет, что какая-то величина больше заданного порога при заданной ширине порогового значения. Если величина меньше заданного порога, функция возвращает 0, если больше – 1, в ином случае – значение от 0 до 1:

```
fuzzy_metrics(X,T,H) = 1.0 :-
        X >= T + H,!.
```

```
fuzzy_metrics(X,T,H) = 0.0 :-
        X <= T - H,!.
fuzzy_metrics(X,T,H) = V :-
        V== (X-T+H) * (1 / (2*H)).
```

В случае обнаружения оставленного предмета требуется оценить, меньше ли стандартизированная площадь и скорость блоба, чем заданный порог (так как нужно понять, что это предмет (сумка), а не человек, и что предполагаемый оставленный предмет не двигается). Поэтому при вычислении метрик *M1* и *M2* вычисляется разность между единицей и значением, полученным при использовании функции *fuzzy_metrics* (так как она сравнивает, больше ли величина, чем порог). Если выполняется условие *M1 * M2 >= 0.25*, логическая программа делает вывод о том, что предмет был оставлен и выдаёт текстовое сообщение о том, что была нарушена стерильность помещения. Результат выполнения программы представлен на рисунке 3.5.

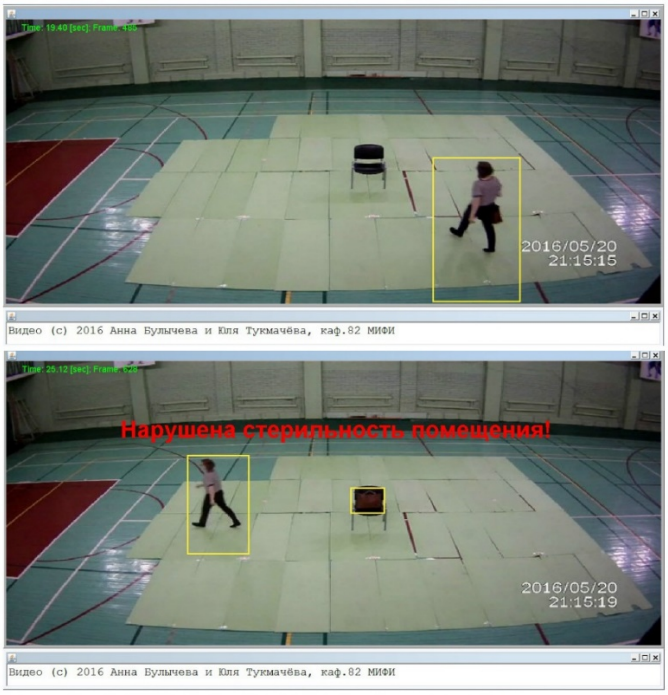

Рисунок 3.5. Программа распознала ситуацию «Оставленный предмет».

Программный код, реализующий данную обработку, прилагается к настоящей дипломной работе на электронном носителе (диск CD-ROM).

Ситуация «Драка между сотрудниками»

Если описывать логическим языком ситуацию «Драка между сотрудниками», она представляется следующим образом: «Двое (или несколько) человек встретились где-то в пределах зоны видеонаблюдения. После этого группа разделяется и, по крайней мере, один из людей убегает».

В логической программе движение блобов представлено в виде связных графов, анализ которых реализуется в классе *'ImageAnalyzer'*. Предикат *is_a_kind_of_a_running_away* описывает искомый граф движения блоба:

```
is_a_kind_of_a_running_away([E2|_],G,E1,E2,E3):-
        E2 == {inputs:O,outputs:B|_},
        B == [_,_|_],
        contains_a_running_person(B,G,E3),
        is_a_meeting(O,G,E2,E1),!.
is_a_kind_of_a_running_away([_|R],G,E1,E2,E3):-
        is_a_kind_of_a_running_away(R,G,E1,E2,E3).
--
contains_a_running_person([N|_],G,P):-
        get_edge(N,G,E),
        is_a_running_person(E,G,P),!.
contains_a_running_person([_|R],G,P):-
        contains_a_running_person(R,G,P).
--
is_a_meeting(O,_,E,E):-
        O == [_,_|_],!.
```

```
is_a_meeting([N1|_],G,_,E2):-
        get_edge(N1,G,E1),
        E1 == {inputs:O|_},
        is_a_meeting(O,G,E1,E2).
--
get_edge(1,[Edge|_],Edge):-!.
get_edge(N,[_|Rest],Edge):-
        N > 0,
        get_edge(N-1,Rest,Edge).
```

Искомый граф описывает ситуацию убегающего человека в контексте ситуации «Драка», если он содержит дугу *E2*, которая:

1) Имеет ребро *E3* в качестве наследника, соответствующее убегающему человеку (проверку на соответствие реализует предикат *contains_a_running_person*).
2) Имеет ребро *E1* в качестве предшественника, соответствующее встрече (проверку на соответствие реализует предикат *is_a_meeting*).

Стоит обратить внимание, что ребро *E2* должно иметь, по крайней мере, два прямых наследника (*B == [_,_|_]*), как минимум один из которых и есть убегающий человек (предикат *is_a_running_person*). Таким образом, у искомой дуги *E2*, соответствующей драке, должны быть предшественники, соответствующие тому, что два (или несколько) человека встретились, и наследники, соответствующие тому, что хотя бы один из людей убежал после инцидента.

Ребро графа соответствует убегающему человеку, если средняя скорость и длина пути блоба отвечают данному нечёткому определению:

```
is_a_running_person(E,_,E):-
        E == {mean_velocity:V,frame1:T1,frame2:T2|_},
        M1== ?fuzzy_metrics(V,2.5,1.0),
```

```
        D== (T2 - T1) / sampling_rate,
        M2== ?fuzzy_metrics(D,0.4,0.2),
        M1 * M2 >= 0.5,!.
is_a_running_person(E,G,P):-
        E == {outputs:B|_},
        contains_a_running_person(B,G,P).
```

Вспомогательная функция для вычисления нечётких метрик приведена ниже (первым аргументом является величина, которая будет оценена, вторым – заданный порог скорости и длины пути соответственно, третьим – ширина порога):

```
fuzzy_metrics(X,T,H) = 1.0 :-
        X >= T + H,!.
fuzzy_metrics(X,T,H) = 0.0 :-
        X <= T - H,!.
fuzzy_metrics(X,T,H) = V :-
        V== (X-T+H) * (1 / (2*H)).
```

Таким образом, программа ИВ сравнивает с пороговыми значениями такие параметры, как средняя скорость и длина пути блоба, для того чтобы сделать вывод о том, что человек бежит. Более того, для того чтобы сделать вывод о том, что был инцидент «Драка», программе нужно найти в графе такие дуги-предшественники, которые будут соответствовать тому, что два человека встретились, прежде чем как минимум один из них убежал. Результат работы программы представлен на рисунке 3.6.

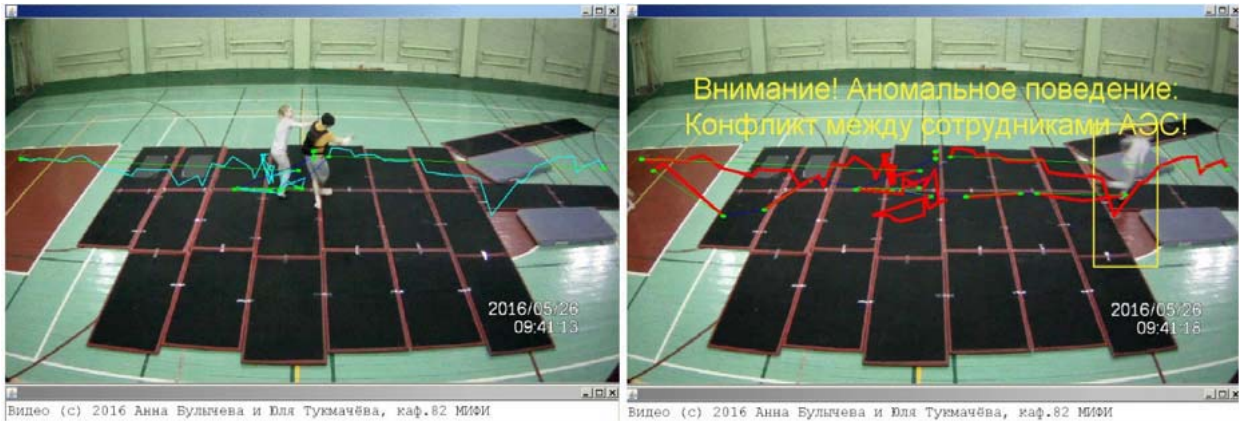

Рисунок 3.6. Программа ИВ распознала ситуацию «Драка между сотрудниками».

Программный код, реализующий данную обработку, прилагается к настоящей дипломной работе на электронном носителе (диск CD-ROM).

3.3 Тестирование разработанных программ ИВ

Экспериментальная проверка программ интеллектуального видеонаблюдения является нетривиальной задачей, потому что распознавание аномального поведения включает некоторую иерархию взаимосвязанных подзадач распознавания [8], а именно:

1) Аккуратное выделение движущихся объектов.
2) Достаточно точное вычисление координат и оценка скорости движения объектов в кадре.
3) Поиск подграфов, соответствующих искомым схемам аномального поведения, в графе траекторий движения объектов.

Простое тестирование алгоритмов распознавания, а именно, подсчёт правильно и неправильно распознанных тестовых образцов, является совершенно недостаточным и, более того, может ввести в заблуждение относительно качества работы программы, потому что не даёт представления о том, как небольшие изменения условий видеонаблюдения, ухудшающие количественные характеристики работы методов низкого уровня, могут повлиять на качество решения задач более высокого уровня [8]. Поэтому для

полноценного тестирования программ интеллектуального видеонаблюдения необходимо:

1) Создание специальных тестовых видеоклипов, отражающих реальные условия видеонаблюдения, включая возможные ракурсы съёмки, возможные изменения освещения и фона, состав наблюдаемых объектов и прочее (подобное тестирование подробно описано в разделе 2.1.3 для анализа вычисленной скорости движущегося человека).
2) Всестороннее тестирование отдельных этапов обработки видеоданных с пониманием того, какие параметры одного этапа обработки существенно влияют на реализацию последующих этапов.
3) Совместное тестирование всех этапов анализа, включая проверку производительности системы анализа при различных уровнях нагрузки.

В данной работе представляет интерес, прежде всего, экспериментальная проверка и тестирование работы новых методов и средств распознавания аномального поведения, разработанных в рамках ВКР. Для проверки стабильности работы программ проверялось, сколько раз программа даёт сбой при непрерывной работе. Причинами сбоев в работе программ, анализирующих видеоизображения в реальном времени, могут быть всевозможные факторы, влияющие на производительность компьютера, а именно, замедление работы компьютера из-за фоновых программ Windows, фрагментация памяти, неравномерная работа сборщика мусора Java и др. При тестировании выяснилось, что программы, описанные в данной работе, ни одного раза из двадцати не дали сбой. Таким образом, проведённое тестирование не выявило ошибок и недочётов, которые могли бы мешать корректному выполнению программ.

Заключение

В настоящей ВКР изучен подход к разработке программ ИВ, основанный на использовании объектно-ориентированного логического языка Акторный Пролог, и разработана база логических программ, распознающих аномальное поведение сотрудников АЭС, представленная пятью ситуациями. Также были исследованы существующие методы и техники ИВ, целью которых является повышение уровня безопасности на АЭС. Для изучения подхода к разработке систем ИВ на основе объектно-ориентированного логического языка Акторный Пролог были сняты видеоклипы, при помощи которых проводилось тестирование методов низкоуровневого анализа, а именно, методов вычитания фона и оценки скорости движущегося объекта. Из проведённого анализа и полученных данных можно сделать следующие выводы:

1) Для корректного вычитания фона, в видеоизображениях не должно присутствовать движение теней от посторонних предметов, более того, камера со штативом должны быть хорошо закреплены и статичны. Для построения аккуратного трека движущегося человека, напольное покрытие в помещениях не должно быть отражающим.
2) Было проведено сравнение вычисленных логической программой скоростей в зависимости от различных параметров (разрешение видеоизображений, направление движения относительно плоскости объектива камеры, использование медианной фильтрации скорости). Согласно полученным данным, максимально близкие к реальным значениям оценки скорости были получены для видеоизображений с высоким разрешением и без использования медианной фильтрации скорости. Однако разброс полученных значений получился большим, за счёт чего средние значения по каждому параметру получились приблизительно одинаковые. Следовательно, чтобы

сделать более достоверные и аккуратные выводы по оценке алгоритма вычисления скорости, нужен более глубокий и детальный анализ, учитывающий связь различных параметров, что выходит за рамки настоящей ВКР.

С целью создания базы логических программ, распознающих аномальное поведение сотрудников АЭС, был составлен перечень ситуаций на основе правил техники безопасности и регламентирующих документов реактора ИРТ МИФИ. Далее, были написаны логические программы, распознающие аномальное поведение, описанное в перечисленных ситуациях. При тестировании программ не было выявлено ошибок и недочётов, которые могли бы мешать корректному выполнению программ.

Список литературы

1. Kim I.S., Choi H.S., Yi K.M., Choi J.Y., Kong S.G. Intelligent Visual Surveillance – A Survey // International Journal of Control, Automation, and Systems. – 2010. – Vol. 8, No. 5. – pp. 926-939.

2. Jorge C.A.F., Seixas J.M., Silva E.A.B., Mól A.C.A., Cota R.E., Ramos B.L. People detection in nuclear plants by video processing for safety purpose // International Nuclear Atlantic Conference (INAC 2011). – Belo Horizonte, MG, Brazil: 2011. – Vol. 42. – pp. 15-30.

3. Singh G.K., Shukla V., Patil S., Shah P. Automatic detection of abnormal event using smart video surveillance system in a nuclear power plant // 55th Annual Meeting of the Institute of Nuclear Materials Management – Atlanta, USA: Institute for Nuclear Materials and Management, 2014. – Vol. 1. – pp. 3139-3146.

4. Kita N. Visual Attention Control for Nuclear Power Plant Inspection // Proceedings of 15th International Conference on Pattern Recognition. – 2000. – Vol. 4. – pp. 118-123.

5. Инструкция по радиационной технике безопасности в комплексе ИРТ МИФИ. № 609Р.07.14-И-06. – Москва: НИЯУ МИФИ, 2014.

6. Morozov A.A. Development of a Method for Intelligent Video Monitoring of Abnormal Behavior of People Based on Parallel Object-Oriented Logic Programming // Pattern Recognition and Image Analysis. – 2015. – Vol. 25, No. 3. – pp. 481-492.

7. Morozov A.A., Vaish A., Polupanov A.F., Antciperov V.E., Lychkov I.I., Alfimtsev A.N., Deviatkov V.V. Development of concurrent object-oriented logic programming platform for the intelligent monitoring of anomalous human activities // Biomedical Engineering Systems and Technologies. 7th International Joint Conference, BIOSTEC 2014, Angers, France, March 3-6, 2014, Revised Selected Papers / Ed. by Guy Plantier and Tanja Schultz and Ana

Fred and Hugo Gamboa. – CCIS 511. – Springer International Publishing, 2015. – pp. 82-97.

8. Morozov A.A., Polupanov A.F. Development of the logic programming approach to the intelligent monitoring of anomalous human behaviour // 9th Open German-Russian Workshop on Pattern Recognition and Image Understanding (OGRW 2014), Proceedings / Ed. by D. Paulus, C. Fuchs, D. Droege. – Koblenz: University of Koblenz-Landau, 2015. – No. 5. – pp. 82-85.

9. Морозов А.А., Сушкова О.С. Анализ видеоизображений в реальном времени средствами языка Акторный Пролог // Международная конференция и молодёжная школа «Информационные технологии и нанотехнологии» (ИТНТ-2016), 17-19 мая 2016 года, Самара, Россия. – Самара: СГАУ & ИСОИ, 2016. – с. 350-356.

Приложение А. Результаты тестирования алгоритма оценки скорости движения людей

Таблица А.1. – Таблица вычисленных логической программой значений скоростей для анализа видео, снятого в помещении. Значения в таблице представлены в «м/с».

Высокое разрешение								**Низкое разрешение**							
Бег				**Шаг**				**Бег**				**Шаг**			
Параллельно		**По диагонали**		**Параллельно**		**По диагонали**		**Параллельно**		**По диагонали**		**Параллельно**		**По диагонали**	
МФ	**Без МФ**	**МФ**	**Без МФ**	**МФ**	**Без МФ**	**МФ**	**Без МФ**	**МФ**	**Без МФ**	**МФ**	**Без МФ**	**МФ**	**Без МФ**	**МФ**	**Без МФ**
		5,5	4,5	1,4	1,5					3,4	3,7	0,6	1,05		
		3,4	3,5	0,6	0,6					2,9	2,6	0,2	0,6		
1,4	1,4	2,2	1,9	0,5	0,6	0,8	0,9	1,03	1,3	0,9	1,13	0,2	0,5	0,2	0,4

Приложение Б. Координаты реперных точек.

Серия 3. Высокое разрешение.

Координаты (единицы измерения: метры):

(0, 7.91); (0, 5.27); (0, 2.93); (2.88, 7.815); (2.88, 4.37); (2.88, 1.27).

Координаты (единицы измерения: пиксели):

(251,264); (512,256); (742,248); (166,463); (599,445); (986,431).

Обратная матрица:

[[0.0001, -0.0016, 0.0001],

[0.0030, 0.0010, 0.0023],

[-0.8309, 1.4168, 1.0000]].

Серия 4. Высокое разрешение (1280x720 px).

Координаты (единицы измерения: метры):

(4.43, 0); (0, 0); (4.43, 2.385); (2.515, 2.41); (0, 2.38).

Координаты (единицы измерения: пиксели):

(503, 356); (954, 353); (465, 501); (719, 499); (1039, 489).

Обратная матрица:

[[0.0328, 0.0103, 0.0003],

[0.0430, -0.0669, 0.0022],

[-10.0751, 5.9671, 1.0000]].

Серия 4. Низкое разрешение (640x480 px).

Координаты (единицы измерения: метры):

(4.43, 0); (0, 0); (4.43, 2.385); (2.515, 2.41); (0, 2.38).

Координаты (единицы измерения: пиксели):

(252, 237); (478, 236); (233, 333); (359, 333); (520, 328).

Обратная матрица:

[[-0.0567, 0.0005, -0.0003],

[0.0257, 0.0921, 0.0083],

[21.0432, -21.9583, 1.0000]].

Серия 5. Высокое разрешение (1280x720 px).

Координаты (единицы измерения: метры):

(0, 0); (2.99, 0); (6.36, 0); (0, 2.10); (2.99, 2.10); (6.36, 2.10); (0, 4.10); (2.99, 4.10); (6.36, 4.10).

Координаты (единицы измерения: пиксели):

(904, 270); (628, 265); (320, 266); (956, 364); (627, 360); (260, 356); (1023, 496); (623, 502); (178, 485).

Обратная матрица:

[[-0.0231, -0.0005, -0.0000],
[0.0121, 0.0555, 0.0042],
[17.6322, -14.4389, 1.0000]].

Серия 5. Низкое разрешение (640x480 px).

Координаты (единицы измерения: метры):

(0, 0); (2.99, 0); (6.36, 0); (0, 2.10); (2.99, 2.10); (6.36, 2.10); (0, 4.10); (2.99, 4.10); (6.36, 4.10).

Координаты (единицы измерения: пиксели):

(452, 180); (314, 177); (161, 177); (478, 243); (314, 240); (130, 238); (512, 331); (312, 335); (90, 332).

Обратная матрица:

[[-0.0446, -0.0007, -0.0001],
[0.0177, 0.0780, 0.0060],
[16.9544, -13.5901, 1.0000]].

Приложение В. Пример логической программы ИВ

Файл с расширением «.a»:

```
import .. from "morozov/Java2D";
import .. from "morozov/Vision";
class 'Main' (specialized 'Alpha'):
constant:
inverse_transformation_matrix        = [
                        [ -0.0231,   -0.0005, -0.0000],
                        [  0.0121,    0.0555,  0.0042],
                        [17.6322, -14.4389,  1.0000]];
sampling_rate       = 25.0;
stage_one           = (('ImagePreprocessor',
                            sampling_rate,
                            low_level_analyzer,
                            stage_two));
stage_two           = (('ImageAnalyzer',
                            low_level_analyzer,
                            sampling_rate));
internal:
low_level_analyzer        = ('ImageSubtractor',
                            extract_blobs= 'yes',
                            track_blobs= 'yes',
                            use_grayscale_colors= 'yes',
                            minimal_training_interval= 50,
                            maximal_training_interval= 50,
                            minimal_blob_size= 5000,
                            apply_gaussian_filtering_to_background= 'yes',
```

```
                                background_gaussian_filter_radius= 1,
                                background_standard_deviation_factor= 1.2,
                                apply_rank_filtering_to_background= 'yes',
                                background_rank_filter_threshold= 5,
                                minimal_track_duration= 5,
                                inverse_transformation_matrix,
                                sampling_rate,
                                apply_median_filtering_to_velocity= 'no');
[
]
class 'ImagePreprocessor' (specialized 'Timer'):
constant:
        sampling_rate;
        low_level_analyzer;
        stage_two;
internal:
        subtractor   = ('SynchronizedImageSubtractor',
                                image_subtractor= low_level_analyzer);
        text         = ('Text');
        image        = ('BufferedImage');
        state        = ('ProgramState');
[
goal:-!,
        Time0== ?milliseconds(),
        state ? set_beginning_time(Time0),
        set_period(1/sampling_rate,0),
        activate.
tick:-
        T2== ?milliseconds(),
```

```
        state ? get_beginning_time(T1),
        Delta== (T2 - T1) / 1000.0 * sampling_rate,
        N== ?convert_to_integer(?round(Delta)),!,
        load_figure(N,T2).
load_figure(N2,_):-
        state ? get_current_frame(N1),
        N1 == N2,!.
load_figure(N,_):-
        state ? set_current_frame(N),
        ImageToBeLoaded==
                "D:/ДИПЛОМ/от А. Морозова/3/students/data/Falling/" +
                "Falling_" +
                text?format("%04d",N) + ".jpg",
        image ? does_exist(ImageToBeLoaded),!,
        image ? load(ImageToBeLoaded),
        subtractor ? subtract(N,image),
        stage_two [<<] draw_scene().
load_figure(_,T2):-
        state ? set_beginning_time(T2),
        subtractor ? reset_results.
]
class 'ImageAnalyzer' (specialized 'Alpha'):
constant:
        sampling_rate;
        low_level_analyzer;
internal:
        subtractor   = ('SynchronizedImageSubtractor',
                                image_subtractor= low_level_analyzer);
internal:
```

```
        graphic_window   = ('Canvas2D',
                           background_color= 'SystemControl',
                           y= 0,
                           height= 22);
        prompt_window    = ('Report',
                           font_size= 24,
                           y= 22,
                           height= 3);
        text             = ('Text');
        image            = ('BufferedImage');
        timer            = ('Timer');
[
goal:-!,
        timer ? set_priority('minimal'),
        prompt_window ? write(
                "Видео (c) 2016 ",
                "Анна Булычева и Юля Тукмачёва, "
                "каф.82 МИФИ"),
        graphic_window ? show.
draw_scene():-
        subtractor ? commit,
        subtractor ? get_recent_frame_number(CFN),
        graphic_window ? suspend_redrawing,
        graphic_window ? clear,
        subtractor ? get_recent_image(image),
        graphic_window ? draw_image(image,0,0,1,1),
        subtractor ? get_connected_graphs(Graphs),
        image ? get_size_in_pixels(IW,IH),
        check_graphs(Graphs,CFN),
```

```
        subtractor ? get_blobs(Blobs),
        draw_blobs(IW,IH,Blobs),
        graphic_window ? set_brush('Green'),
        graphic_window ? set_font({size:18}),
        graphic_window ? set_text_alignment('LEFT','CENTER'),
        graphic_window ? draw_text(
                15 / IW, 15 / IH,
                text?format(
                        "Time: %3.2f [sec]; Frame: %s",
                        CFN/sampling_rate,CFN)),
        graphic_window ? draw_now.
draw_blobs(IW,IH,[Blob|Rest]):-!,
        draw_blob(IW,IH,Blob),
        draw_blobs(IW,IH,Rest).
draw_blobs(_,_,_).
draw_blob(IW,IH,Blob):-
        Blob == {
                x:X0,y:Y0,
                width:W1,height:H1|_},
        graphic_window ? set_brush('off'),
        graphic_window ? set_pen({color:'Yellow',lineWidth:3}),
        graphic_window ? set_font({size:21,weight:'WEIGHT_BOLD'}),
        graphic_window ? set_text_alignment('CENTER','CENTER'),
        X2== (X0 - W1 / 2) / IW,
        Y2== (Y0 - H1 / 2) / IH,
        W2== W1 / IW,
        H2== H1 / IH,
        graphic_window ? draw_rectangle(X2,Y2,W2,H2),
        fail.
```

```
draw_blob(_,_,_).
check_graphs([Graph|Rest],CFN):-
        check_graph(Graph,CFN),!,
        check_graphs(Rest,CFN).
check_graphs(_,_).
check_graph([Edge|_],CurrentFrame):-
        Edge == {coordinates:TrackOfBlob|_},
        there_is_a_lying_person(
                TrackOfBlob,'no',0,
                FirstFrame,
                LastFrame),
        LastFrame == CurrentFrame,!,
        Duration== (LastFrame - FirstFrame) / sampling_rate,
        report_target_event(Duration).
check_graph([_|Rest],CFN):-!,
        check_graph(Rest,CFN).
check_graph(_,_).
there_is_a_lying_person([Blob],'no',_,LF,LF):-
        this_is_a_lying_person(Blob,LF),!.
there_is_a_lying_person([Blob],'yes',FF,FF,LF):-
        this_is_a_lying_person(Blob,LF),!.
there_is_a_lying_person([Blob|Rest],'no',_,FF2,LF):-
        this_is_a_lying_person(Blob,FF1),!,
        there_is_a_lying_person(Rest,'yes',FF1,FF2,LF).
there_is_a_lying_person([Blob|Rest],'yes',FF1,FF2,LF):-
        this_is_a_lying_person(Blob,_),!,
        there_is_a_lying_person(Rest,'yes',FF1,FF2,LF).
there_is_a_lying_person([_|Rest],IsLying,FF1,FF2,LF):-
        there_is_a_lying_person(Rest,IsLying,FF1,FF2,LF).
```

```
this_is_a_lying_person(Blob,N):-
        Blob == {width:W,height:H,frame:N|_},
        Ratio== W / H,
        Ratio > 1.5.
report_target_event(Duration):-
        Duration > 3.0,!,
        graphic_window ? set_text_alignment('CENTER','CENTER'),
        graphic_window ? set_font({size:48,weight:'WEIGHT_BOLD'}),
        graphic_window ? set_pen({color:'Red'}),
        graphic_window ? draw_text(
                0.5,0.25,
                "Сотрудник АЭС потерял сознание!!!").
report_target_event(_):-
        graphic_window ? set_text_alignment('CENTER','CENTER'),
        graphic_window ? set_font({size:48,weight:'WEIGHT_BOLD'}),
        graphic_window ? set_pen({color:'Orange'}),
        graphic_window ? draw_text(
                0.5,0.25,
                "Сотрудник АЭС упал!").
]
-----------------------------------------------------------------
class 'ProgramState' (specialized 'Database'):
[
CLAUSES:
get_beginning_time(T):-
        Item== ?match(beginning_time(_)),
        Item == beginning_time(T),!.
set_beginning_time(T):-
        retract_all(beginning_time(_)),
```

```
        append(beginning_time(T)).
    get_current_frame(N):-
        Item== ?match(current_frame(_)),
        Item == current_frame(N),!.
    set_current_frame(N):-
        retract_all(current_frame(_)),
        append(current_frame(N)).
    ]
```

Файл с расширением «.def»:

```
    pragma: INTERMEDIATE_SOURCE_CODE = JDK;
    pragma: USE_COMPILER_INSTEAD_OF_PROVER = ON;
    import .. from "morozov/Java2D";
    import .. from "morozov/Vision";
    DOMAINS:
    PersonIsLying                = 'no'; 'yes'.
    interface 'Main' (specialized 'Alpha'):
    constant:
        inverse_transformation_matrix        : TransformationMatrix;
        sampling_rate                        : REAL;
        stage_one            : 'ImagePreprocessor';
        stage_two            : 'ImageAnalyzer';
    internal:
        low_level_analyzer        : 'ImageSubtractor';
    [
    ]
    interface 'ImagePreprocessor' (specialized 'Timer'):
    constant:
        sampling_rate                : REAL;
```

```
        low_level_analyzer        : 'ImageSubtractor';
        stage_two                 : 'ImageAnalyzer';
internal:
        subtractor          : 'ImageSubtractor';
        text                : 'Text';
        image               : 'BufferedImage';
        state               : 'ProgramState';
[
PREDICATES:
imperative:
load_figure(INTEGER,INTEGER)                - (i,i);
]
interface 'ImageAnalyzer' (specialized 'Alpha'):
constant:
        sampling_rate             : REAL;
        low_level_analyzer        : 'ImageSubtractor';
internal:
        subtractor          : 'SynchronizedImageSubtractor';
        graphic_window      : 'Canvas2D';
        prompt_window       : 'Report';
        text                : 'Text';
        image               : 'BufferedImage';
        timer               : 'Timer';
[
PREDICATES:
imperative:
draw_scene;
draw_blobs(PointX,PointY,BlobList)          - (i,i,i);
draw_blob(PointX,PointY,Blob)               - (i,i,i);
```

```
check_graphs(GraphList,FrameNumber)                    - (i,i);
check_graph(ConnectedGraph,FrameNumber)                - (i,i);
determ:
there_is_a_lying_person(
        TrackOfBlob,
        PersonIsLying,
        FrameNumber,
        FrameNumber,
        FrameNumber)              - (i,i,i,o,o);
this_is_a_lying_person
        (BlobCoordinates,FrameNumber)      - (i,o);
imperative:
report_target_event(REAL)                          - (i);
PREDICATES:
imperative:
'+'(PointX,PointX) = PointX                        - (i,i);
'+'(PointY,PointY) = PointY                        - (i,i);
'-'(PointX,PointX) = PointX                        - (i,i);
'-'(PointY,PointY) = PointY                        - (i,i);
'/'(PointX,PointX) = PointX                        - (i,i);
'/'(PointY,PointY) = PointY                        - (i,i);
]
interface 'ProgramState' (specialized 'Database'):
[
DOMAINS:
Target                  = beginning_time(INTEGER);
                          current_frame(INTEGER).
PREDICATES:
determ:
```

```
get_beginning_time(INTEGER)        - (o);
get_current_frame(INTEGER)         - (o);
imperative:
set_beginning_time(INTEGER)        - (i);
set_current_frame(INTEGER)         - (i);
]
```