**Система мониторинга динамики транспортных потоков на базе метода виртуальных детекторов - «ViDeS» / System for monitoring the dynamics of traffic flows based on the method of virtual detectors - «ViDeS» (Virtual Detectors System)**

Система предназначена для мониторинга характеристик транспортных потоков на выделенных участках магистрали по видеоданным. С помощью графического интерфейса пользователь задает характеристики виртуальных детекторов и размещает их на видео, после чего начинается обработка видеопотока. В результате работы система осуществляет детектирование транспортных средств, проходящих через детекторы, фильтрует их и записывает в файл с расширением .csv. Обрабатывая данные из файла, происходит получения числовых характеристик плотности, интенсивности и средней скорости транспортных потоков на заданных участках магистрали за заданный временной интервал. Имеется возможность визуализации полученных характеристик в виде графиков.

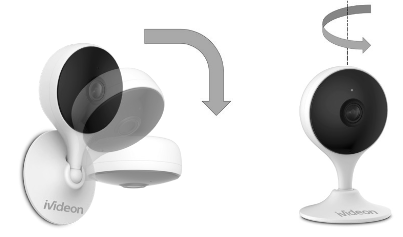
# 1. Метод виртуальных детекторов

Метод виртуальных детекторов позволяет собирать данные практически с любого ракурса съемки благодаря гибкости в настройке размера, форме и расположению детекторов. Рассмотрим видеопоток с фиксированной панорамой и периодичностью *∆t*. На экране монитора есть виртуальные датчики. Мы называем их контрольным доменом (КД). КД – прямоугольная область на экране для регистрации и накопления цветовых нарушений: *xij*, где *i = 1, ..., m*, где *m* – высота каждого изображения с видеокамеры в пикселях: *j = 1, ..., n*, где *n* – ширина каждого изображения с видеокамеры в пикселях. Мы будем использовать координаты в градациях серого *xij* ∈ [0, 255]. На каждом кадре детектор вычисляет средний цвет в пределах заданной области, затем средние значения цвета могут быть использованы программой для определения наличия транспортного средства в кадре.

# 2. Описание датасета

Для проведения качественного мониторинга транспортных потоков к видеоданным предъявляются определенные требования. Зачастую камеры, осуществляющие трансляцию, обладают низким разрешением, небольшим углом обзора объектива и неудачным размещением, не позволяющим использовать видеоданные с них.

В качестве входных данных используется видеопоток с камеры *iVideonCute 2* (Рис. 1).



*Рис. 1.* Способы размещения камеры iVideonCute 2.

Камера установлена в лаборатории кафедры высшей математики МАДИ (Рис. 2) и транслирует движение автотранспорта по Ленинградскому шоссе в профиль.



*Рис. 2.* Схема размещения камеры на карте.

Она размещена на высоте ~ 18 м под углом ~ 60°, что позволяет осуществлять захват ~ 300 м магистрали (Рис. 3). Для вычисления плотности потока требуется участок большой протяженности, чему полностью удовлетворяет длина захваченного участка.

Изображение выглядит как внешний

Автоматически созданное описание

*Рис. 3.* Захват видеоданных с камеры – 16:00 06.07.2022.

Характеристики камеры и видеоданных представлены в Табл. 2.1.

Таблица 2.1

Технические характеристики камеры

| **Модель iVideonCute 2** | |
| --- | --- |
| Сенсор | 1/2,7, 2 Мп |
| Чувствительность | 0,00249лк |
| Угол обзора объектива | 2,8 мм горизонтальный: 112°  2,8 мм вертикальный: 58°  2,8 мм диагональный: 131° |
| **Сжатие** | |
| Видеосжатие | H/264/MJPEG |
| Битрейт видео | 116 Мб/с |
| **Изображение** | |
| Основной поток | 1920х1080, 25 к/с |

Расположение, угол размещения и разрешение камеры позволяют осуществлять захват 20 полос движения. Из-за проблемы перекрытия транспортных средств друг другом для применения метода виртуальных детекторов наиболее пригодны 10 ближайших из них. Для обзора всего шоссе требуется размещение камеры с противоположной стороны.

# 3. Развертывание программного комплекса

В качестве программного средства использовалась разработанная на кафедре «Высшая математика» система мониторинга динамики транспортных потоков на базе метода виртуальных детекторов - «ViDeS» / System for monitoring the dynamics of traffic flows based on the method of virtual detectors - «ViDeS» (Virtual Detectors System).

## 3.1. Используемый программный комплекс﻿

Python – это интерпретируемый объектно-ориентированный язык программирования высокого уровня с динамической семантикой. Его высокоуровневые встроенные структуры данных в сочетании с динамической типизацией и динамической привязкой делают его очень привлекательным для быстрой разработки приложений, а также для использования в качестве языка сценариев или клея для соединения существующих компонентов вместе. Простой, легкий в освоении синтаксис Python подчеркивает удобочитаемость и, следовательно, снижает затраты на обслуживание программы. Python поддерживает модули и пакеты, что способствует модульности программ и повторному использованию кода. Интерпретатор Python и обширная стандартная библиотека доступны в исходном или двоичном виде бесплатно для всех основных платформ и могут свободно распространяться.

Python часто обеспечивает повышенною производительность. Поскольку этап компиляции отсутствует, цикл редактирования-тестирования-отладки выполняется быстро. Отладка программ на Python проста: ошибка или неверный ввод никогда не вызовут сбоя сегментации. Вместо этого, когда интерпретатор обнаруживает ошибку, он вызывает исключение. Когда программа не улавливает исключение, интерпретатор выводит трассировку стека. Отладчик исходного уровня позволяет проверять локальные и глобальные переменные, вычислять произвольные выражения, устанавливать точки останова, выполнять пошаговый просмотр кода по строке за раз и так далее. Отладчик написан на самом Python, что свидетельствует о способности Python к самоанализу. С другой стороны, часто самый быстрый способ отладки программы – это добавить несколько инструкций печати в исходный код: быстрый цикл редактирования-тестирования-отладки делает этот простой подход очень эффективным.

В качестве среды разработки использовалась IDE PyCharm.

PyCharm – это кроссплатформенная IDE, обеспечивающая единообразную работу в операционных системах Windows, macOS и Linux.

PyCharm доступен в двух редакциях: Professional и Community. Версия Community – это проект с открытым исходным кодом, бесплатный, но с меньшим количеством функций. Профессиональная версия является коммерческой и предоставляет выдающийся набор инструментов и функций.

## 3.2. Применяемые библиотеки

Ниже представлено частичное описание используемых библиотек:

* Matplotlib – это обширная библиотека для создания статических, анимированных и интерактивных визуализаций на Python.
* NumPy обеспечивает: 1. мощный объект N-мерного массива; 2. сложные (вещательные) функции; 3. инструменты для интеграции кода C/C++ и Fortran; 4. полезная линейная алгебра, преобразование Фурье и возможности случайных чисел и многое другое.

Помимо очевидного научного применения, NumPy также можно использовать в качестве эффективного многомерного контейнера общих данных. Могут быть определены произвольные типы данных. Это позволяет NumPy легко и быстро интегрироваться с самыми разными базами данных.

* OpenCV-Python – это библиотека привязок Python, предназначенная для решения задач компьютерного зрения.
* Pandas – это пакет Python, который предоставляет быстрые, гибкие и выразительные структуры данных, предназначенные для того, чтобы сделать работу с «реляционными» или «помеченными» данными простой и интуитивно понятной. Он призван стать фундаментальным строительным блоком высокого уровня для практического анализа данных в реальном мире в Python. Кроме того, у него есть более широкая цель – стать самым мощным и гибким инструментом анализа/манипулирования данными с открытым исходным кодом, доступным на любом языке. Он уже на пути к этой цели.
* Pip – это установщик пакетов для Python. Вы можете использовать pip для установки пакетов из индекса пакетов Python и других индексов.

## 3.4. Настройка системы ViDeS

В качестве первого этапа идет настройка параметров расстановки детекторов и настройка их чувствительности. В системе предусмотрен следующий набор параметров для настройки пользователем перед запуском.

* **drawing** = True/False - Включить/Отключить визуализацию обработки
* **create\_median\_frame** = True/False - True - Создать медианный кадр перед началом обработки / False - взять в качестве медианного кадра изображение median\_frame.png
* **set\_detectors\_by\_hand** = True/False - True - Установка детекторов вручную / False - взять координаты детекторов из файла Coordinates.csv
* **metric** = 1 - Выбранная метрика 1-4
* **detector\_heigth** = 30 - Высота детектора в пикселях
* **detector\_width** = 60 - Ширина детектора в пикселях
* **activation\_avg\_colour\_delta** = 1.5 - Значение разницы метрики в % при котором будет проходить активация детектора
* **frames\_unite** = 10 - Количество кадров, на которые смотрим вперед от текущего для их объединения в случае повторения активаций.

## 3.5. Первоначальный запуск системы ViDeS

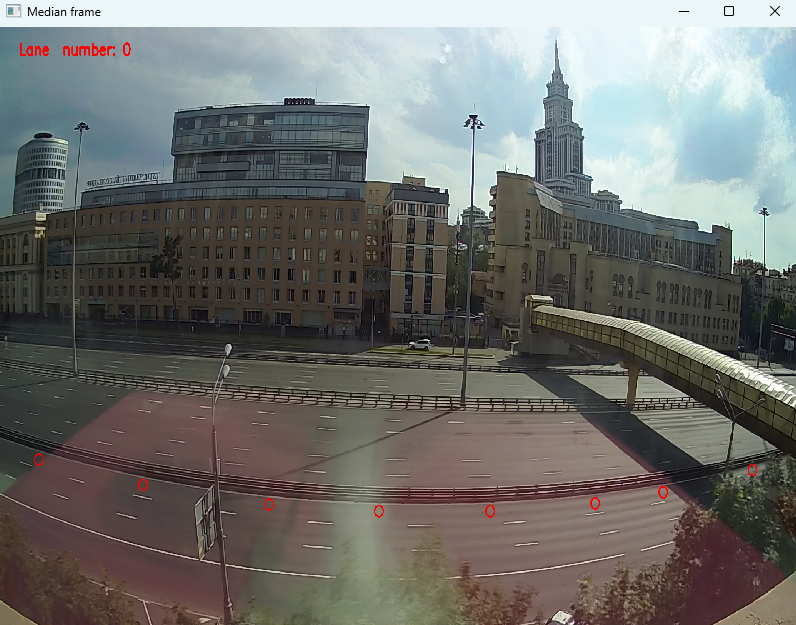
Перед первоначальным запуском нужно поместить видеофайлы для обработки в папку Videos и установить рекомендуемые параметры:

* **drawing** = True
* **create\_median\_frame** = True
* **set\_detectors\_by\_hand** = True
* **metric** = 1 - Выбранная метрика 1-4

В этом случае на основе первого видеофайла в папке Videos будет сгенерирован медианный кадр и сохранен в виде изображения median\_frame.png. В случае если медианный кадр получается неудовлетворительным в папку Videos можно поместить более качественный видеофайл, сгенерировать медианный кадр для него и использовать его в дальнейшем.



Затем на медианном кадре требуется расставить точки вдоль полосы, на основе которых будет проинтерполирована функция, вдоль которой будут расставляться детекторы заданного размера. Точки расставляются кликами мыши в любом порядке по ходу движения.



Для сохранения добавленных точек и перехода к расстановке точек следующей полосы требуется нажать на клавиатуре клавишу “w” на английской раскладке. В случае успеха в консоли выведется сообщение “lane added” после чего можно приступать к расстановке точек следующей полосы.



Таким образом требуется расставить точки для всех полос.



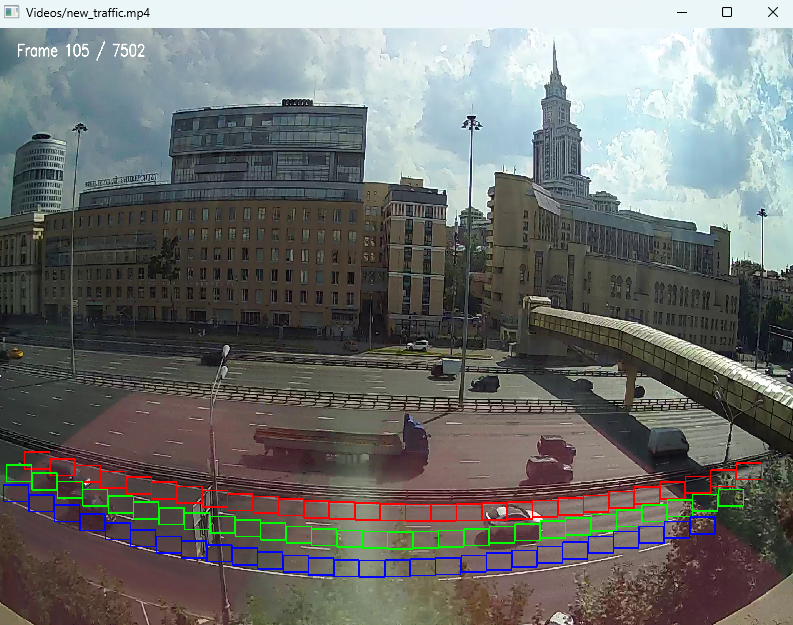
Затем для начала обработки следует зажать клавишу “e” на английской раскладке пока не закроется текущее окно и не начнется обработка. После этих действий для каждой полосы будут сгенерированы детекторы их координаты будут сохранены в файл Coordinates.csv. В дальнейшем при установке параметров

* **create\_median\_frame** = False
* **set\_detectors\_by\_hand** = False

обработка видеофайлов будет начинаться сразу.

## 3.6. Обработка видео системой ViDeS

Обработка видео начнется либо после ручной расстановки детекторов, либо после взятия их координат их файла Coordinates.csv при запуске с соответствующими параметрами. Для каждого видеофайла будет генерироваться свое окно с именем, обрабатываемого файла, и числом обработанных / общим числом кадров в верхнем левом углу. После обработки окно уничтожается. Остановить обработку с сохранением файлов можно нажатием клавиши “q” на английской раскладке.



После завершения обработки заполняются следующие файлы:

* MetricValues.csv – значения выбранной метрики для каждого детектора для каждой полосы для каждого кадра;
* RawDetections.csv – неотфильтрованные значения активации для каждого детектора для каждой полосы для каждого кадра. Определяются как 1 - если разница между метрикой для текущего и последующего кадра больше, чем параметр **activation\_avg\_colour\_delta;**
* FilteredDetections.csv – отфильтрованные значения активаций полученные для параметра **frames\_unite**;

## 3.7. Описание метрик

В качестве метрик для каждого детектора используются следующие значения – цвет пикселя с координатами ij и N – количество пикселей, располагающихся внутри детектора.

Представлены следующие метрики

1. Средний цвет области детектора -
2. Среднее значение из корня суммы квадратов цветов пикселей области детектора -
3. Среднее значение модуля разницы между цветом пикселей области детектора на текущем кадре и предыдущем -

Модуль разницы функций для текущего и предыдущего кадров -

1. Сумма пикселей области детектора на основе функции с заданным пороговым значением -

# 4. Задание

В качестве задания требуется провести мониторинг транспортных потоков трех полос магистрали для предоставленного датасета с помощью метода виртуальных детекторов. Провести предварительный подбор оптимальных параметров и чувствительности детекторов для каждого часового временного интервала.

В качестве первого этапа требуется посчитать количество транспортных средств на каждой полосе за заданный интервал времени вручную.

Затем требуется подобрать оптимальные параметры размера и чувствительности детекторов и построить графики по данным из файла MetricValues.csv для заданной метрики так, чтобы количество и время появления транспортных средств на видео максимально совпадало с реальными данными.

После этого требуется обработать весь массив данных для подобранных оптимальных параметров.

В отчете отразить процесс ручного подсчета и подбора оптимальных параметров с указанием точности для каждого эксперимента.