**Общее руководство пользователя системы «РоадАР Аналитика - падение средней скорости транспортного потока»**

**Содержание**

[**Введение 3**](#_je8m6wcagh1x)

[**1 Назначение и условия применения системы “РоадАР Аналитика - падение средней скорости транспортного потока” 4**](#_3dy6vkm)

[1.1 Назначение системы 4](#_1t3h5sf)

[1.2 Базовый функционал 4](#_4d34og8)

[1.3 Функциональные характеристики 4](#_czhijeca4yrq)

[1.4 Программное обеспечение рабочего места 4](#_2s8eyo1)

[1.5 Виды пользователей 5](#_17dp8vu)

[1.6 Описание системы 5](#_3rdcrjn)

[**2 Инструкция по установке системы 8**](#_8hnbzf3klg5a)

[2.1 Установка Docker 8](#_b98xe46k97cc)

[2.2 Скачивание и сбор частей архив, загрузка образа и его запуск 9](#_z83rduyufh6j)

[**3 Контактная информация производителя программного продукта 11**](#_ecomzkltjoke)

[3.1 Юридическая информация 11](#_3rdcrjn)

[3.2 Контактная информация службы технической поддержки 11](#_2tow4ctptd8)

#

# Введение

**“РоадАР Аналитика - падение средней скорости транспортного потока”** – система, разработанная компанией ООО “РоадАР” для распознавания и фиксации падения средней скорости транспортного потока на полосе - аналитика генерирует событие, если на полосе зафиксировано падение средней скорости более, чем на Х процентов.

**Система “РоадАР Аналитика - падение средней скорости транспортного потока”** поддерживает механизмы взаимодействия со сторонними информационными системами. Посредством вызова API поставляемой библиотеки. А также текущая версия системы предполагает выбор и реализацию конкретного механизма взаимодействия с внешними системами по согласованию с заказчиками.

# 1 Назначение и условия применения системы “РоадАР Аналитика - падение средней скорости транспортного потока”

## 1.1 Назначение системы

Обработка изображений для определения и фиксации падения средней скорости транспортного потока на полосе - аналитика генерирует событие, если на полосе зафиксировано падение средней скорости более, чем на Х процентов.

## 1.2 Базовый функционал

* обработка изображений;
* определение и фиксация падения средней скорости транспортного потока на полосе - аналитика генерирует событие, если на полосе зафиксировано падение средней скорости более, чем на Х процентов;
* информационный обмен с внешними системами.

## 1.3 Функциональные характеристики

Текущая версия системыпозволяет решать задачи по выявлению падения средней скорости транспортного потока на полосе - аналитика генерирует событие, если на полосе зафиксировано падение средней скорости более, чем на Х процентов.

## 1.4 Программное обеспечение рабочего места

Библиотека может быть встроена в программное обеспечение, установленное на рабочем месте пользователя. В этом случае клиентская часть **системы “РоадАР Аналитика - падение средней скорости транспортного потока”** может использоваться на любом рабочем месте, имеющем подключение к сети Internet (или сети передачи данных предприятия). Разрешающая способность видеосистемы и монитора – не ниже 1280х1024. Рекомендуется широкоформатный монитор.

| Вид ПО | Программный продукт |
| --- | --- |
| ОС (приведены варианты) | Windows 10 Linux (Ubuntu, Debian, Альт, ROSA, UBLinux, ICLinux). |

*Таблица 1. Системные требования*

Данные требования могут меняться в зависимости от особенностей программного обеспечения, которое использует библиотеку системы **“РоадАР Аналитика - падение средней скорости транспортного потока”**.

## 1.5 Виды пользователей

Пользователем системы является пользователь программного обеспечения, в которое встраивается библиотека **системы “РоадАР Аналитика - падение средней скорости транспортного потока”**. Описание доступных возможностей API выполняется для пользователя, обладающего максимально возможными правами по доступу к программе.

## 1.6 Описание системы

API Системы написано на C++.

***~ src/core/modules/analytic\_specific\_params.hpp:***

…

namespace an::core {

/\*!

 \* \brief Общая структура для хранения оющих парметров для дорожных аналитик

 \*/

 struct RoadAnalyticParams {

 RoadAnalyticParams() = default;

 /\*\*

 \* \brief Конструктор с параметрами

 \* \param [in] grnd указатель на класс для проекции объектов на дорогу

 \* \param [in] roadLanes множество полос, поданных на вход аналитике

 \* выпаших грузов

 \*/

 RoadAnalyticParams(std::shared\_ptr<GroundCalibration> &grnd,

 std::vector<RoadLane> lanes)

 : ground(grnd), roadLanes(std::move(lanes)) {

 spdlog::info("Road lanes count for road analytic: {}", roadLanes.size());

 }

 /// Conversion API -> Core

 explicit RoadAnalyticParams(const an::RoadAnalyticParams &params) {

 std::transform(params.roadLanes.begin(),

 params.roadLanes.end(),

 std::back\_inserter(roadLanes),

 [](const an::RoadLane &rl) { return RoadLane(rl); });

 }

 bool operator==(const RoadAnalyticParams &other) const {

 bool eqGround = ground == other.ground

 || (ground && other.ground && \*ground == \*other.ground);

 return eqGround && roadLanes == other.roadLanes;

 }

 /// указатель для проецирования объектов на дорогу (через калибровочную

 /// матрицу)

 std::shared\_ptr<GroundCalibration> ground;

 /// множество полос дороги

 std::vector<RoadLane> roadLanes;

 };

/\*!

 \* \brief Структура для хранения параметров дорожной аналитики "падение

 \* средней транспортного потока"

 \*/

 struct SpeedDecreaseAnalyticParams : RoadAnalyticParams {

 /\*\*

 \* \brief Конструктор с параметрами

 \* \param [in] roadParams общие параметры для дорожной аналитики

 \* \param [in] minCarCount минимальное количество ТС, при котором

 \* аналитика генерирует событие \param [in] dropPercent процент, на

 \* который должна упасть скорость ТС, чтобы аналитика отправила событие

 \* \param [in] minSpeedIgnore минимальная скорость (км/ч), ниже которой

 \* скорости ТС не включаются в подсчет средней \param [in] maxSpeedIgnore

 \* максимальная скорость (км/ч), ниже которой скорости ТС не включаются в

 \* подсчет средней \param [in] intervalCheck интервал (сек) времени для

 \* проверки падения скорости

 \*/

 SpeedDecreaseAnalyticParams(RoadAnalyticParams roadParams,

 int minCarCount,

 float dropPercent,

 int minSpeedIgnore,

 int maxSpeedIgnore,

 int intervalCheck)

 : RoadAnalyticParams(std::move(roadParams)),

 minCarCount(minCarCount),

 dropPercent(dropPercent),

 minSpeedIgnore(minSpeedIgnore),

 maxSpeedIgnore(maxSpeedIgnore),

 intervalCheck(intervalCheck) {}

 /// Conversion API -> Core

 explicit SpeedDecreaseAnalyticParams(

 const an::SpeedDecreaseAnalyticParams &params)

 : RoadAnalyticParams(params),

 minCarCount(params.minCarCount),

 dropPercent(params.dropPercent),

 minSpeedIgnore(params.minSpeedIgnore),

 maxSpeedIgnore(params.maxSpeedIgnore),

 intervalCheck(params.intervalCheck) {}

 /// минимальное число ТС, которое должно проехать по полосе за последние

 /// intervalCheck секунд

 int minCarCount;

 /// процент, на который должна упасть скорость ТС, чтобы аналитика

 /// отправила событие

 float dropPercent;

 /// минимальная скорость (км/ч), ниже которой скорости ТС не включаются в

 /// подсчет средней

 int minSpeedIgnore;

 /// максимальная скорость (км/ч), ниже которой скорости ТС не включаются в

 /// подсчет средней

 int maxSpeedIgnore;

 /// интервал (сек) времени для проверки падения скорости

 int intervalCheck;

 bool operator==(const SpeedDecreaseAnalyticParams &other) const {

 if (!RoadAnalyticParams::operator==(other))

 return false;

 return minCarCount == other.minCarCount

 && dropPercent == other.dropPercent

 && minSpeedIgnore == other.minSpeedIgnore

 && maxSpeedIgnore == other.maxSpeedIgnore

 && intervalCheck == other.intervalCheck;

 }

 };

}

***~ src/core/modules/ traffic\_average\_speed\_decrease/traffic\_average\_speed\_decrease\_analytic.hpp:***

#pragma once

#include "modules/analytic.hpp"

#include "modules/analytic\_specific\_params.hpp"

#include "modules/tracker/vehicle/track\_analyzer/track\_analyzer.hpp"

namespace an { namespace core {

 struct LaneCarImage {

 an::core::MatCore img;

 cv::Rect rect;

 uint64\_t trackId = 0;

 };

 /\*\*

 \* \brief count car speeds, calculates average speed and send events when car

 \* goes slower

 \*

 \* Store the average speeds history for each lane, find the longest decreasing

 \* subsequence that ends at the last element of the history, if we detect a

 \* sequence with a size more than 3 then we detect that the speed decrease

 \* started and as long as this subsequence keeps getting new elements then we

 \* the speed decrease is still there.

 \*

 \* if any of the following reasons occur the even will end:

 \*

 \* 1. No more speed decrease

 \*

 \* 2. The number of the cars is less than the minimum number of cars specified

 \*

 \* In addition, the parameters min speed and max speed are used to clamp the

 \* value of the current speed to be in the range [min, max] (if it was smaller

 \* it will become min and if it was larger it will be considered as max)

 \*/

 class TrafficAverageSpeedDecreaseAnalytic : public Analytic {

 public:

 /\*\*

 \* \param analyticId

 \*

 \* Unique identifier of analytic

 \* Usually generated by UUID generator

 \*

 \* \param analyticEventQueue

 \*

 \* Uncomming queue from previous layer of pipeline

 \*

 \* \param needSavePreview

 \*

 \* Argument enables preview image

 \*

 \* \param roadParams

 \*

 \* Set of parameters such as lane geomerty

 \*/

 explicit TrafficAverageSpeedDecreaseAnalytic(

 const std::string &analyticId,

 const AnalyticEventQueue &analyticEventQueue,

 bool needSavePreview,

 const SpeedDecreaseAnalyticParams &roadParams);

 /\*\*

 \* Entrypoint to processing

 \*

 \* \param frame

 \*

 \* pointer to frame container taht stores useful information

 \*/

 void processFrame(const pFrame &frame) override;

 private:

 std::mutex mutex\_;

 std::unique\_ptr<TrackAnalyzer> trackAnalyzer\_;

 std::unordered\_map<std::string, std::string> laneIdToStartedEventId\_;

 int dropSpeedGenerateCount\_;

 float dropSpeedGeneratePercent\_;

 int dropSpeedIntervalCheck\_;

 cv::Range dropSpeedAllowRange\_;

 an::AnalyticEvents produceEvents(const an::core::pFrame &frame);

 static void drawSpeedDecreaseForDemo(

 const an::core::pFrame &frame,

 const std::vector<RoadLaneStatus> &lanes,

 const std::map<std::string, bool> &laneHasSpeedDropped,

 const std::vector<Track> &carTracks);

 static void drawPolyAndLegendInfo(

 const pFrame &frame,

 const std::vector<RoadLaneStatus> &lanes,

 const std::map<std::string, bool> &laneHasSpeedDropped);

 static void drawTracks(const pFrame &frame,

 const std::vector<RoadLaneStatus> &lanes,

 const std::vector<Track> &carTracks);

 std::shared\_ptr<an::AnalyticEvent> generateAndPrintEventStartedDetails(

 const pFrame &frame,

 const RoadLaneStatus &lane,

 const DropSpeedDetails &details);

 std::shared\_ptr<an::AnalyticEvent> generateAndPrintEventFinishedDetails(

 const pFrame &frame,

 const RoadLaneStatus &lane,

 const DropSpeedDetails &details);

 void showLanesAverageSpeed(const pFrame &frame);

 };

}} // namespace an::core

***~ src/core/modules/ traffic\_average\_speed\_decrease/traffic\_average\_speed\_decrease\_analytic\_controller.hpp:***

#pragma once

#include "modules/analytic\_controller.hpp"

namespace an { namespace core {

 class TrafficAverageSpeedDecreaseAnalyticController

 : public AnalyticController {

 public:

 TrafficAverageSpeedDecreaseAnalyticController(

 const FrameQueue &inputQueue,

 const FrameQueue &outputQueue,

 const AnalyticEventQueue &eventQueue);

 ~TrafficAverageSpeedDecreaseAnalyticController() noexcept override;

 AnalyticType getControllerAnalyticType() const override;

 protected:

 std::unique\_ptr<Analytic> makeSpecificAnalytic(

 const AnalyticSpecificParams &params) override;

 };

}} // namespace an::core

#

# 2 [Инструкция по установке системы](#_26in1rg)

## 2.1 Установка Docker

В инструкции подразумевается, что пользователь использует ОС Linux, Ubuntu 20.04 (для других дистрибутивов, инструкции могут отличаться)

Обязательным предусловием для установки серверной платформы MDT является установка следующих пакетов:

* ***docker 18.06.1-ce+***;

На Ubuntu их можно установить следующими командами:

| $ sudo apt-get update$ sudo apt-get install \ apt-transport-https \ ca-certificates \ curl \ gnupg-agent \ software-properties-common$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -$ sudo add-apt-repository \ "deb [arch=amd64] https://download.docker.com/linux/ubuntu \ $(lsb\_release -cs) \ stable"$ sudo apt-get update$ sudo apt-get install docker-ce docker-ce-cli containerd.io |
| --- |

Детали для установки Докера и добавления вашего пользователя в группу Докер можно найти по ссылкам:

1. <https://docs.docker.com/compose/install>
2. [https://docs.docker.com/install/linux/docker-ce/ubuntu](https://docs.docker.com/install/linux/docker-ce/ubuntu/).

## 2.2 [Скачивание и сбор частей архив, загрузка образа и его запуск](https://docs.google.com/document/d/1D2HKTmWfVFK2WDGv-CmaTSiSUEOrdlzCtZN6pbzHwXc/edit#heading=h.z83rduyufh6j)

1. Необходимо установить архиватор. В терминале системы Linux команда:

sudo apt install unzip

1. Скачайте папку Документы РоадАР Аналитики <https://www.roadar.info/file-share/5ad2fb14-dd07-427d-afb5-a0feb1f99cbe>Наименование файла: \_\_\_\_\_\_\_\_\_\_\_

Для загрузки файла необходимо авторизоваться. Логин и пароль передаются при покупке системы.

Загруженный файл представляет собой архив с компонентами системы. Пароль от него так же передаётся при покупке системы.

 

Рисунок 1. Скачивание папки с демо.

1. С помощью терминала зайдите в папку Документы РоадАР Аналитика - падение средней скорости транспортного потока. В текущей рабочей директории вы должны увидеть папку demo\_image\_parts и архив rename.zip



1. Разархивируйте архив rename.zip (пароль не нужен).

| unzip rename.zip |
| --- |

1. Выполните команду (она уберет .zip с конца названия файла):

 python3 rename.py

1. Собираем один архив из его частей:

|  cat demo\_image\_parts/analytics\_demo.part-\* > analytics\_image.zip |
| --- |

1. Далее разархивируем полученный файл (потребуется ввести пароль):

|  unzip analytics\_demo\_img.zip |
| --- |

1. Затем загружаем полученный докер-образ в список образов докера

|  docker load -i analytic\_demo |
| --- |

1. Далее запускаем образ:

|  docker run --entrypoint=/app/entrypoint.sh analytics:rospotent |
| --- |



Рисунок 2. Пример вывода в консоль удачного запуска (дождитесь вывода статуса).



Рисунок 3. Пример вывода текущего статуса (обновляется каждые 2 минуты).

#

# 3 Контактная информация производителя программного продукта

## 3.1 Юридическая информация

* **Название компании:** ООО «РоадАР».
* **ИНН 1615013172**
* **ОГРН 1161690183665**
* **Юр. адрес:** 420500, г Иннополис, ул Университетская, д 7, офис 332

## 3.2 Контактная информация службы технической поддержки

* **Сайт:** roadar.info
* **Email:** info@roadar.info
* **Тел.:** +7-903-307-16-75

Фактический адрес размещения инфраструктуры разработки:

420500, г. Иннополис, ул. Университетская, дом 7, офис 715

Фактический адрес размещения разработчиков:

420500, г. Иннополис, ул. Университетская, дом 7, офис 715

Фактический адрес размещения службы поддержки:

420500, г. Иннополис, ул. Университетская, дом 7, офис 715