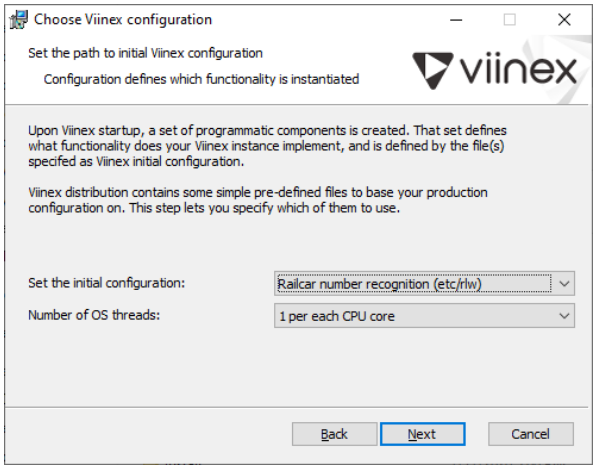
Распознавание номеров грузовых железнодорожных вагонов в Viinex 3.0

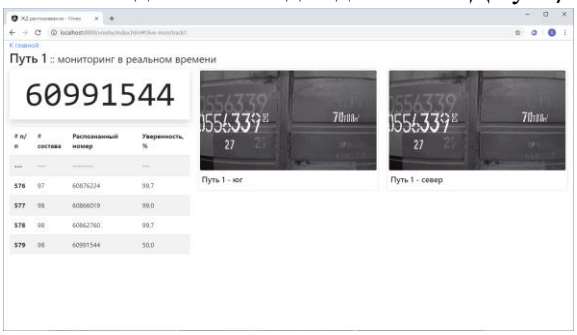
Первая установка и запуск демонстрационного ролика

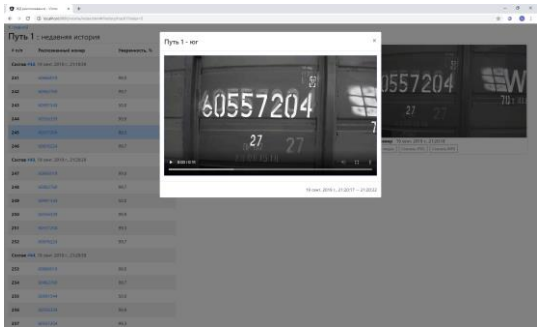
Для первой установки и запуска системы с демо-роликом предлагается выполнить следующие шаги:

- Скачать дистрибутив по ссылке ftp://ftp.viinex.com/viinex-3.0/Viinex-3.0.0.403.msi
- Установить дистрибутив. При установке выбрать в выпадающем меню выбора начальной конфигурации предпоследний пункт (Railcar number recognition, папка etc/rlw). (Это рекомендуется делать только при ознакомительной установке. Для разработки и реальной эксплуатации Viinex 3.0 рекомендуется выбирать опцию «Конфигурация в папке etc/conf.d»).



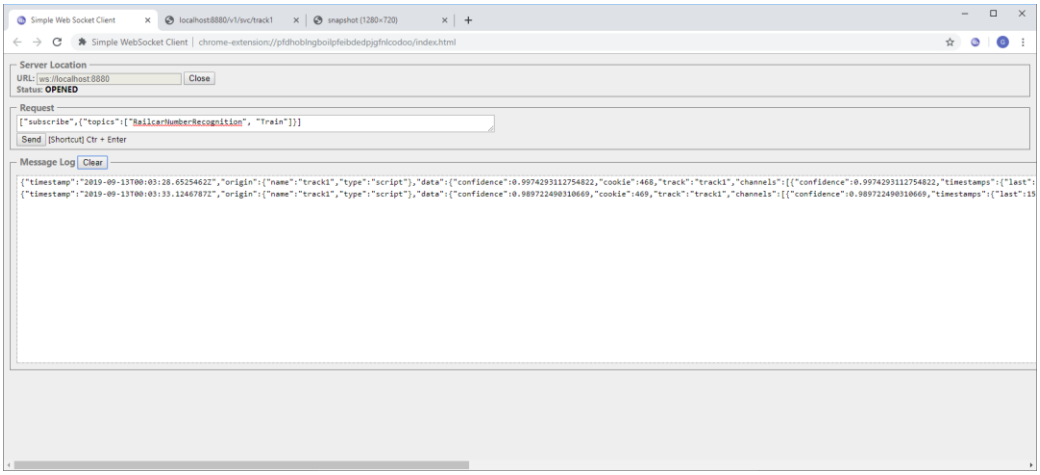
- Этот и следующий шаг нужен только для демонстрации: Скачать демонстрационный ролик по ссылке ftp://ftp.viinex.com/demo-video/rlpr/rlwuz.mp4
- Скопировать файл rlwuz.mp4 в папку etc/rlw.
- Установить драйвер SenseLock ftp://ftp.viinex.com/viinex-3.0/senselock_windows_3.1.0.0.zip , если он еще не был установлен. Вставить в компьютер USB ключ Viinex 3.0.
- Запустить службу Viinex 3.0.
- В браузере Chrome открыть демонстрационный веб-интерфейс системы распознавания номеров ЖД вагонов, доступный по ссылке http://localhost:8880/vnxrlw/index.html. В веб-приложении по этой ссылке должны быть доступны все сконфигурированные ЖД пути (после установки по умолчанию это 1 путь), для каждого из которых будет доступен режим мониторинга в реальном времени (отображение живого видео от всех видеоканалов для данного ЖД пути, и получение в реальном времени событий о результатах распознавания номеров вагонов), а также режим просмотра последней истории (просмотр информации о вагонах в последних зарегистрированных составах: время прохождения состава, результаты распознавания вагонов, а также просмотр и экспорт архивного видео и изображений вагонов по всем видеоканалам для данного ЖД пути).



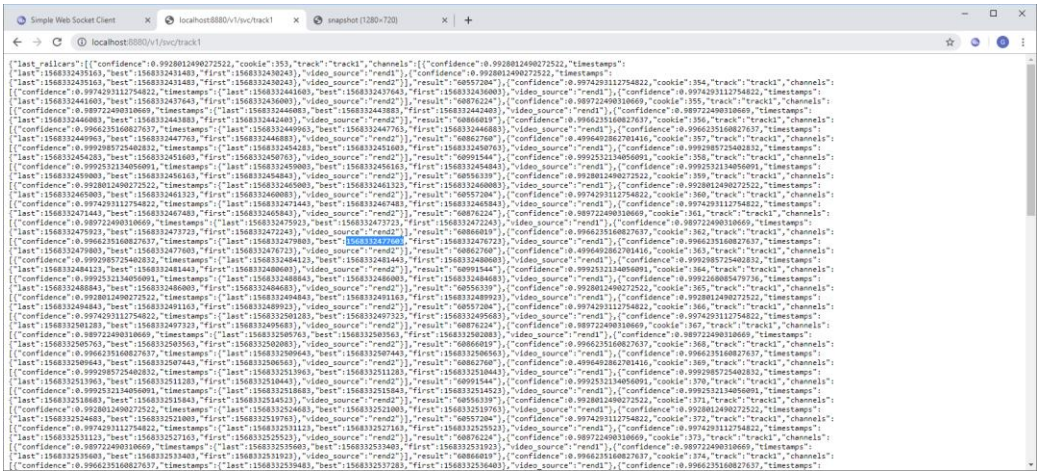


- Открыть webrtc видеопоток от эмулируемой камеры 1 или камеры 2 (http://localhost:8880/#/webrtc-video/webrtc0/cam1).

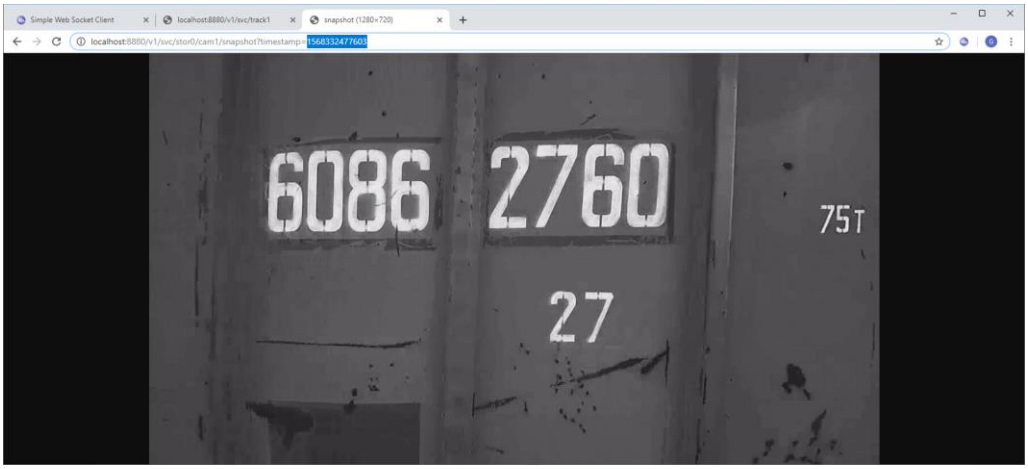
- Этот и следующий шаг нужен только для тестирования функциональности получения событий. Установить в браузере Chrome расширение Simple WebSocket Client https://chrome.google.com/webstore/detail/simple-websocket-client/pfdhoblngboilpfeibdedpjgfnlcodoo .
- Запустить расширение Simple WebSocket Client. Подключиться к Viinex (в поле Server Location URL ввести ws://localhost:8880 и нажать Open). Подписаться на события о распознавании номеров ЖД вагонов (в поле Request ввести `["subscribe",{"topics":["RailcarNumberRecognition", "Train"]}]`). Нажать кнопку Send. В нижнем окне “Message log” должны начать приходить события о распознанных номерах.



- Получить последние распознанные номера с помощью HTTP запроса: в браузере открыть ссылку http://localhost:8880/v1/svc/track1



- Получить наилучший кадр из видеоархива по временной метке: выбрать в результатах предыдущего запроса значение метки времени timestamps.best для одного из вагонов (выделено на изображении), и используя это значение, открыть ссылку http://localhost:8880/v1/svc/stor0/cam1/snapshot?timestamp=1568332477603 (в параметре timestamp= должно быть подставлено актуальное значение времени).



Компоненты и внутреннее устройство системы

Система распознавания номеров грузовых железнодорожных вагонов и цистерн на базе Viinex 3.0 содержит реализацию модуля консолидации результатов распознавания, который принимает команды на распознавание и управляет распознаванием в подчиненных внешних процессах. Указанный модуль имеет тип ridrcons и описан в разделе 2.1.17 общей документации к Viinex 3.0, доступной по ссылке https://www.viinex.com/ViinexGuide.pdf .

Объект ridrcons принимает команды на начало и окончание распознавания каждого отдельного вагона с использованием программного интерфейса Updateable (см.раздел 3.16 общей документации), занимается декодированием видеопотоков от связанных с ним IP источников, публикует эти видеопотоки в общей области памяти, доступной другим процессам; управляет жизненным циклом подчиненных процессов, которые занимаются непосредственно видеоаналитикой; собирает результаты анализа отдельных кадров от подчиненных процессов, консолидирует эти результаты, и выдает финальный результат по вагону при поступлении команды на окончание распознавания по данному вагону. Настройка объекта ridrcons существенно упрощена по сравнению со схемой из внешнего процесса и нескольких объектов renderer, которая применялась в решении для распознавания номеров ЖД вагонов в Viinex 2.0. Указанные объекты в Viinex 3.0 создавать не требуется; достаточно на каждый обслуживаемый ЖД путь создать объект ridrcons, указав в его конфигурации связанные источники видео. Объект ridrcons автоматически создаст видеодекодеры и подчиненные процессы.

Вместе с тем, хотя объект ridrcons и может быть использован напрямую, -- рекомендуется использовать его в связке с управляющим скриптом, который может реализовывать необходимую логику реакции системы на события от оптических датчиков, вызовов API, событий, эмулируемых субтитрами и т.п., а также управлять записью видео в архив, накапливать недавние результаты распознавания, предоставлять эти результаты простейшему пользовательскому интерфейсу для отображения. В решении для распознавания номеров ЖД вагонов в Viinex 3.0 роль управляющего скрипта не изменилась, по сравнению с Viinex 2.0.

Видеоаналитика

Видеоаналитика, как внешний процесс, реализована в исполняемом файле iv-ridr.exe, запускается объектом ridrcons автоматически, в количестве, требуемом в соответствии с конфигурацией, и обслуживает один видеоканал распознавания (то есть запускается по одному такому процессу на каждый видеоканал распознавания). Процессы iv-ridr.exe работают целиком под управлением объекта ridrcons; взаимодействие пользователя с ними не требуется. Будучи отдельным по отношению к Viinex процессом. Данный процесс получает декомпрессированное видео для анализа. Для этого используется механизм «локального транспорта», коротко описанный в гл.5 общей документации. Источниками декомпрессированного видео являются видеодекодеры, автоматически создаваемые объектом ridrcons при запуске. При получении декодированного кадра через локальный транспорт процесс iv-ridr.exe начинает анализировать этот кадр, и по окончании анализа возвращает объекту

ridrcons результаты анализа по данному кадру. Решение о том, какие кадры предъявлять для анализа, и каким образом объединять результаты (по разным кадрам от одного видеоканала, и от разных видеоканалов) принимает объект ridrcons. В свою очередь, этот объект принимает внешние команды посредством интерфейса Updateable, описанного в разделах 3.18.2 и 4.3.6 общей документации. В ответ на команду окончания распознавания номера текущего вагона, объект ridrcons синхронно возвращает результат распознавания по этмоу вагону (т.е. как результат того же HTTP запроса или JavaScript вызова). Вместе с тем, объект ridrcons является источником событий, и отправляет результаты распознавания всем своим подписчикам, асинхронно. Этой функциональнсти в принципе достаточно для использования объекта ridrcons для распознавания номеров ЖД вагонов по внешнему сигналу (вызову API), без дополнительных компонентов.

Управление распознаванием и логика работы

Объект ridrcons реализует лишь минимальную необходимую функциональность для распознавания номеров ЖД вагонов, он получает на вход простые команды, выдает результат распознавания, не запоминая его, и практически не имеет внутреннего состояния. Этот объект рекомендуется рассматривать как внутренний компонент системы распознавания номеров ЖД вагонов. Взаимодействие с другими частями системы и более сложную логику реализует управляющий скрипт. Механизм встроенных в Viinex 3.0 скриптов описан в главе 4 общей документации. Особенностью такого способа построения логики системы распознавания номеров вагонов в Viinex 3.0 является то, что компания Viinex предоставляет клиентам достаточную для типовых решений реализацию управляющего скрипта, однако эта реализация является открытой, поставляется в виде исходного кода, и при необходимости доступна для модификации.

Управляющий скрипт создается как объект Viinex типа «script»; исходный код типовой реализации управляющего скрипта устанавливается в папку Program Files\Viinex\etc\rlw в файл rlwlogic.js . Экземпляр объекта «script», загружающий исходный код из данного файла, должен быть создан для каждого ЖД пути, для которого требуется распознавание номеров вагонов. В соответствии с принятой в Viinex логикой, соответствующий объект script должен быть связан с другими объектами Viinex, с которыми он должен взаимодействовать, в разделе links. Это всегда будет объект ridrcons, который осуществляет распознавание, а также, как правило, объект Recording Controller (для управления записью видео в архив), иногда объект типа “modbus” или другой источник событий DigitalInput (для получения сигналов от оптического датчика), а также веб-сервер (для публикации событий с результатами распознавания, а также для обработки управляющих API-запросов). В демонстрационный конфигурациях вместо объекта modbus может фигурировать объект h264sourceplugin, который является одновременно источником и видео, и субтитров, которые служат для имитации сигналов от оптического датчика, синхронизированных с видеопотоком.

Типовая реализация управляющего скрипта в ходе работы принимает события от оптического датчика (либо субтитры, в случае имитации сигналов от датчика), интерпретирует эти события и принимает решение о включении либо выключении распознавания по своему ЖД пути. Если в конфигурации скрипта выбрано управление распознаванием через API, а не автоматически по событиям, то скрипт обрабатывает управляющие HTTP API запросы (см. раздел 3.16.2 документации), и в качестве реакции на них включает или выключает распознавание. Для управления распознаванием скрипт взаимодействует с объектом ridrcons посредством понятных последнему команд через интерфейс Updateable. Результаты распознавания, полученные от объекта ridrcons, управляющий скрипт аккумулирует, публикует в веб-сервере (см. раздел 3.16.1 документации), и отправляет в качестве событий через интерфейс WebSocket. В случае, если управление распознаванием осуществляется внешнием ПО через API, -- скрипт повторяет логику объекта ridrcons, и возвращает результат распознавания немедленно, синхронно, как результат команды на окончание распознавания по вагону. Это позволяет не использовать WebSockets в ряде сценариев использования системы.

Скрипт также осуществляет управление записью видео в архив, включая запись в момент появления состава на ЖД пути, и выключая ее после того, как состав проехал.

Настройка системы распознавания вагонов

Режим работы

Режим работы системы распознавания номеров вагонов в тестовой конфигурации управляется файлом rlw3.json.

Для того чтобы не потерять изменения конфигурации, рекомендуется при установке Viinex 3.0 для разработки ПО и внедрения – выбирать размещение конфигурации в папке "etc/conf.d". Установочная программа не изменяет и не удаляет файлы, которые находятся в etc/conf.d, но удаляет и заменяет файлы, находящиеся в папке etc и etc/rlw при удалении/переустановке Viinex 3.0.

Наиболее важной настройкой системы распознавания номеров вагонов является режим управления распознаванием – по датчику, по API вызову, либо по субтитрам, имитирующим срабатывание датчика. Режим задается значением свойства "init.mode" объекта script, соответствующего ЖД пути. Это свойство может быть установлено в значение "ray", "api", "subtitles" для перечисленных случаев. Дополнительно к этому, при выборе вариантов "ray" или "subtitles", в конфигурации должен присутствовать соответствующий одноименный раздел, указывающий, какие именно события будут инициировать включение и выключение распознавание номера очередного вагона. В частности, в разделе "ray" для управляющего скрипта, работающего по сигналу от оптических датчиков, устанавливается название источника событий о срабатывании датчиков в конфигурации Viinex (как правило, это объект для связи с контроллером Modbus, но может быть и другой источник событий типа DigitalInput), номер дискретного входа на данном котроллере, а также флаг invert, позволяющий инвертировать логику включения/выключения распознавания при замыкании/размыкании контакта.

В случае режима работы "api", дополнительные настройки не требуются. В этом случае управляющему скрипту ("track1" в тестовой конфигурации) можно послать HTTP запрос POST, содержащий JSON объект с булевым полем "recognize" (true для начала распознавания, и false для окончания), и опциональным значением cookie, в котором может находиться произвольное 64-разрядное целое значение. Это значение служит для того, чтобы впоследствии можно было идентифицировать результат распознавания, соответствующей конкретной команде.

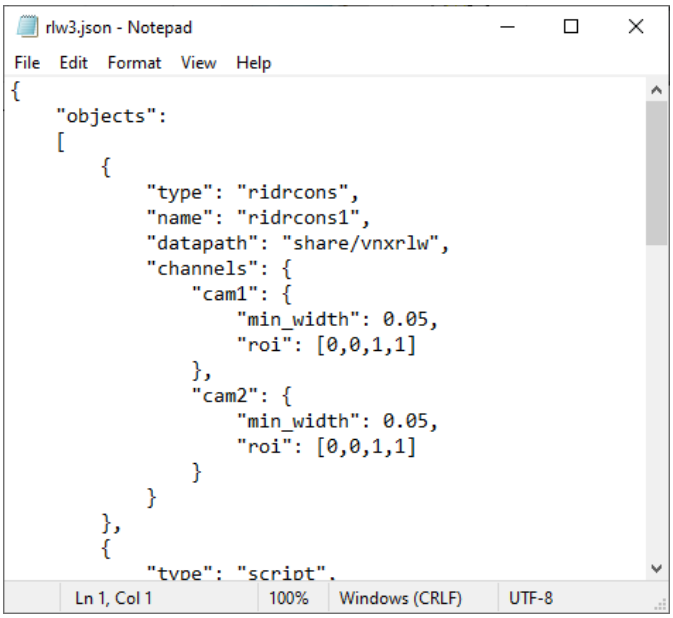
Опциональной частью конфигурации управляющего скрипта является раздел "substitute". Его назначение состоит в том, чтобы в результатах распознавания, получаемых от движка, заменить названия источников видео с названия производных служебных объектов video renderer (с которыми может работать движок) на названия исходных камер. Это может быть необходимо, если скрипт в зависимости от полученных событий переключает видеопотоки, которые используются для распознавания. В простейшей конфигурации в Viinex 3.0 настройка substitute может быть опущена.

Кроме того, в конфигурации управляющего скрипта может присутствовать раздел meta. Как и для любых других объектов Viinex, этот раздел не передается самому экземпляру объекта и не влияет на его работу, но виден через HTTP API Viinex, и, в частности, служит для настройки работы демонстрационного пользовательского интерфейса системы распознавания номеров ЖД вагонов, который доступен на веб-сервере Viinex по ссылке http://localhost:8880/vnxrlw/index.html. В разделе

meta для управляющего скрипта должны присутствовать значения: meta.type, равное “RawilwayTrack” – тег, по которому веб-приложение среди других объектов Viinex идентифицирует скрипт, связанный с ЖД путем; meta.name и meta.desc – строковые значения, задающие название ЖД пути и его описание, выводимое в пользовательском интерфейсе; meta.channels – список видеоканалов, связанных с данным ЖД путем, который используется для отображения видеопотоков в интерфейсе; наконец, булевское значение meta.api позволяет указать, следует ли в пользовательском интерфейсе отображать кнопку для управления распознаванием, если скрипт настроен на работу в управляемом режиме (“mode”: “api”).

Настройки движка распознавания

Настройки движка распознавания содержатся в конфигурации объекта ridrcons, соответствующего данному ЖД пути, и описаны в разделе 2.1.17 общей документации. Конфигурация объекта ridrcons должна содержать параметр channels, который должен быть ассоциативным массивом, задающим соответствие между именем источника видео, связанного с объектом ridrcons (каналом распознавания), и конфигурацией для указанного канала. В Viinex 3.0 в качестве конфигурации канала распознавания можно задать минимальную ширину номера, который будет искать алгоритм, -- в долях от ширины изображения, -- а также ROI (прямоугольную область для поиска и распознавания номера).



Видеоканалы, используемые для распознавания, также должны быть связаны с требуемым объектом ridrcons в разделе links конфигурации Viinex.

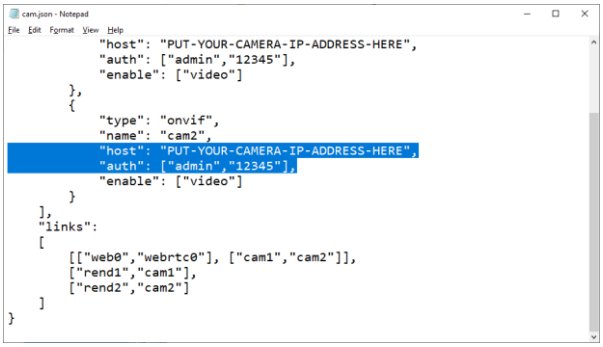
Настройка других объектов Viinex 3.0

В целом, настройка других объектов Viinex 3.0 для работы системы распознавания номеров ЖД вагонов не отличается от настройки Viinex для других задач, и описана в общей документации. Здесь приведен короткий список настроек, которые необходимо менять при развертывании системы на любом объекте.

Демонстрационная конфигурация Viinex содержит настройку для распознавания номеров по двум камерам, и разбита на относительно независимые части, как описано в разделе 2.5 документации.

Источники видео

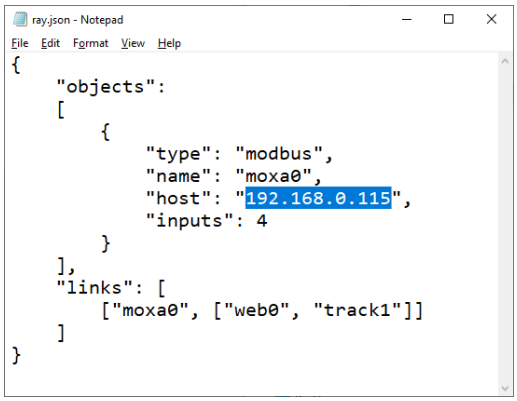
В демонстрационной конфигурации настройка источников видео описана в файлах cam.json и/или fileplayer.json. Это взаимно исключающие файлы: они используют одинаковые идентификаторы cam1, cam2 для источников видео. В файле cam.json определена настройка для подключения системы к двум ONVIF камерам. При использовании этого варианта, в данном файле необходимо изменить значения “host” (установить в нем реальные IP адреса видеокамер в локальной сети), а также, вероятно, значения “auth” (установить в нем логин и пароль для соответствующей камеры). Подробно настройка подключения к ONVIF камерам описана в разделе 2.1.2 документации.



При использовании демонстрационного видеоролика вместо камер, может быть использован файл fileplayer.json. В нем может потребоваться изменение пути к файлу, содержащему демонстрационный ролик. В демонстрационной конфигурации предполагается, что ролик будет помещен в папку etc/rlw, то есть рядом с файлами демонстрационной конфигурации.

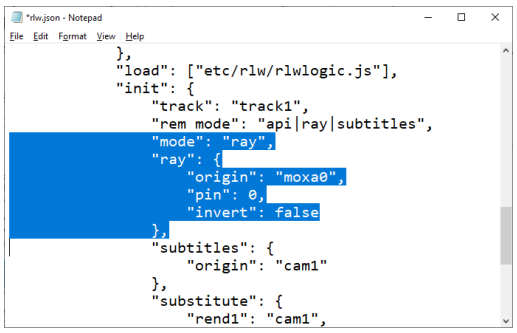
Файлы cam.json и fileplayer.json содержат взаимно исключающие элементы конфигурации, поэтому при использовании одного из этих файлов, второй должен быть исключен из конфигурации: удален из папки, содержащей конфигурационные файлы, либо его расширение должно быть изменено с .json на какое-либо иное (например, .json_, с подчеркиванием на конце).

Источники событий



При использовании оптического датчика для разделения ЖД состава на вагоны, интеграция такого датчика с Viinex возможна через подключение специализированного Ethernet-контроллера (стандартного Modbus TCP, или реализующего другой поддерживаемый в Viinex протокол VK Module Socket 1). Подробоности настройки подключения к такому контроллеру описаны в разделе 2.1.11 документации.

В демонстрационной конфигурации подключение Modbus контроллера описано в файле ray.json. В реальной инсталляции в данном файле необходимо установить значение IP адреса контроллера, а также, возможно, количество сухих контактов, которые нужно считывать.



Конкретный номер GPIO контакта, который будет использован для управления распознаванием номеров вагонов на каждом отдельном ЖД пути, указывается в настройках управляющего скрипта в разделе init.ray (свойство origin для указания идентификатора объекта Modbus котроллера, и свойство pin для указания номера сухого контакта). Там же может быть указано свойство invert, инвертирующего логику интерпретации сигнала от оптического датчика. Может потребоваться изменение данного свойства в зависимости от установки датчика (нормально замкнут либо нормально разомкнут).

В случае имитации сигнала от датчиков посредством дорожки с субтитрами в демонстрационном видеоролике, в настройках управляющего скрипта выбирается “mode”: “subtitles”, и в разделе init.subtitles в поле origin должен быть указан идентификатор объекта h264sourceplugin, который является также источником событий-субтитров. Формат субтитров для имитации сигнала от луча прост: в субтитрах должно быть текстовое значение «0» (одна цифра ноль) для имитации сигнала о замыкании оптического датчика (т.е. вагона нет), и текстовое значение «1» (либо 2, 3, и т.д.), -- числовое значение, отличное от нуля, записанное в десятичной системе счисления без дополнительных символов, пробелов и т.п., -- для имитации сигнала о размыкании оптического датчика (т.е. вагон есть).

Видеоархив и запись

Настройки видеоархива и записи содержатся в демонстрационной конфигурации в файле rec.json. Это типовые настройки полностью описанные в разделах 2.1.6 и 2.1.7 документации. В реальной установке в файле rec.json может потребоваться изменить путь к папке с видеоархиваом Viinex (свойство folder у элемента с типом storage), а также предельные значения для хранимого объема видео, в абсолютных цифрах – свойство max_size_gb, -- размер на диске, в гигабайтах, отведенный под видеоархив, -- либо в относительных цифрах, -- свойство keep_free_percents, -- проценты свободного места на диске, которые следует держать свободными при заполнении архива. Подробно смысл этих настроек описан в разделе 2.1.6 документации.

Объект recctl управляет записью группой камер в видеоархив. Отдельный экземпляр этого объекта нужно создать для каждого ЖД пути. В настройках объекта recctl может потребоваться изменение значения prerecord, задающего, сколько секунд видео должно быть записано ДО получения управляющим скриптом информации о начале ЖД состава.

Кроме того, при добавлении камер, в том числе обзорных, эти новые камеры должны быть связаны в разделе links конфигурации Viinex с объектом recctl для соответствующего ЖД пути (в примере выше – строка `"links": [..., ["recctl1", ["cam1","cam2"]] ]`, -- список связанных камер может быть расширен.

Программный интерфейс

Взаимодействие с системой распознавания номеров вагонов в Viinex 3.0 осуществляется по протоколу HTTP и WebSocket. Данное взаимодействие, специфичное для задачи распознавания номеров вагонов, осуществляет управляющий скрипт. Исходный код данного скрипта поставляется в открытом виде и доступен для изменения клиентами, поэтому разработчики могут расширить API системы при необходимости.

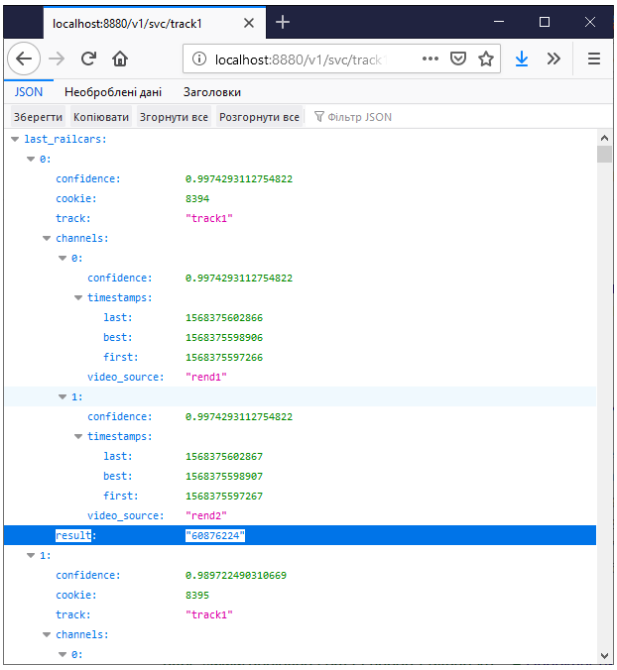
Получение результатов по запросу

Для получения недавних результатов распознавания по каждому отдельному ЖД пути можно использовать HTTP запрос GET к управляющему скрипту, обслуживающему данный ЖД путь. Результаты распознавания возвращаются в форме JSON объекта, содержащего самый последний результат распознавания (в поле last_result), а также масив из некоторого числа последних результатов распознавания (в поле last_railcars). Структура объектов, описывающих результат распознавания по каждому вагону, в данных полях одинакова.

Результат распознавания по каждому вагону содержит поля:

- track – идентификатор ЖД пути, на котором был получен результат
- result – собственно текстовое поле, содержащее распознанный номер

- confidence – числовое значение в диапазоне от 0 до 1, характеризующее уверенность движка распознавания в результате, в порядковой шкале, т.е. смысл имеет не само значение данного числа, а отношения порядка между этими числами для различных результатов распознавания: чем больше значение confidence, тем больше уверенность движка в результате
- cookie – целое 64-разрядное число, идентифицирующее акт распознавания. При работе в автоматическом режиме, когда управляющий скрипт инициирует начало и окончание распознавания при получении событий от оптического датчика, значение cookie выбирается управляющим скриптом (в типовой реализации это просто целочисленный счетчик). При работе в управлямом режиме, когда распознавание инициируется через API, значение cookie может быть задано внешним ПО, и, таким образом, может служить для идентификации результата распознавания конкретного вагона



- channels – массив, содержащий для каждого вагона информацию по деталям распознавания по каждому из видеоканалов. А именно, по каждому из видеоканалов указывается:
    - video_source – идентификатор объекта renderer, на видеопотоке от которого был получен результат
    - confidence – уверенность движка в результате по видео с данного видеоканала
    - timestamps – объект, содержащий три временные метки, first, last и best, указывающие время первого кадра, обработанного для данного вагона по данному видеоканалу, время последнего кадра для данного вагона, а также временную метку кадра, на котором был получен наилучший результат. Эти временные метки могут быть использованы для получения снапшотов и проигрывания видеороликов, записанных в видеоархиве.

Получение результатов распознавания в реальном времени

Для считывания результатов распознавания по мере того, как эти результаты возникают в ходе работы системы, можно использовать протокол WebSocket и соответствующий программный интерфейс, описанный в разделе 3.17 документации. После подключения по протоколу WebSocket к веб-серверу Viinex, приложение должно подписаться на интересующие его события, используя, например, команду

```
["subscribe",{"topics":["RailcarNumberRecognition", "Train"]}]
```

После этого приложение начнет получать в свой WebSocket канал события о результатах распознавания, а также о начале и окончании прохождения состава, генерируемые управляющим скриптом системы распознавания номеров ЖД вагонов. Каждое событие о результатах распознавания является отдельным результатом распознавания по отдельному вагону. Содержащиеся в нем поля описаны в предыдущей секции настоящего документа.

Получение результата распознавания при управлении распознаванием через API

Если управляющий скрипт настроен на управление распознаванием через API, он повторяет поведение объекта ridrcons в том смысле, что при получении команды на окончание распознавания по текущему вагону, -- этот результат распознавания немедленно выдается как результат соответствующего HTTP API вызова. Формат результата соответствует описанному выше. Это отличие (в сравнении с Viinex 2.0) позволяет в сценариях, когда распознаванием управляет клиентское ПО, -- избежать необходимости подписываться на получение событий от Viinex через WebSocket. При этом, конечно,

результат распознавания «синхронным» способом (как результат API вызова) сможет получить только один компонент ПО, -- тот, который этот вызов осуществляет. Вместе с тем, получение результатов через интефейс WebSocket остается доступно в любом случае, для любого числа клиентов.

Получение снапшотов и видео из видеоархива

После окончания записи видео в архив, из последнего могут быть получены отдельные кадры, а также проиграны либо экспортированы видеоролики, на которых зарегистрировано прохождение отдельных вагонов либо ЖД состава в целом.

Программный интерфейс для получения кадров и роликов из видеоархива описан в разделах 3.7.1, 3.5.5 и 3.5.6 документации Viinex 3.0. В частности, для получения кадра из видеоархива следует использовать HTTP GET запрос к URL вида

http://localhost:8880/v1/svc/stor0/cam1/snapshot?timestamp=1568332477603

где параметр timestamp должен содержать временную метку кадра, полученную, например, из свойства timestamps.best результата распознавания вагона, как описано в параграфе «Получение результатов по запросу» настоящего раздела. Идентификаторы stor0 и cam1 в данном запросе также должны соответствовать конкретному видеоархиву и видеокамере в конфигурации Viinex.

Для проигрывания и экспорта видеороликов из архива Viinex можно использовать, соответственно, URL вида

http://localhost:8880/v1/svc/stor0/cam1/stream.m3u8?begin=1568377513426&end=1568377517106

http://localhost:8880/v1/svc/stor0/cam1/export?format=isom&begin=1568377513426&end=1568377517106

где параметры begin и end должны содержать временную метку начала и конца запрашиваемого видеофрагмента. Первая ссылка (stream.m3u8) выдаст HLS поток и может быть использована для проигрывания видео в браузере с помощью, например, скрипта hls.js. Вторая ссылка позволяет скачивать (экспортировать) MP4 файл, генерируемый непосредственно в момент скачивания, на лету.

Управление распознаванием через API

Если, вместо использования сигналов от оптических датчиков для управления распознаванием в автоматическом режиме, у внешнего ПО есть необходимость управлять началом и окончанием распознавания, -- управляющий скрипт должен в конфигурации иметь свойство, задающее режим работы “mode”, установленное в значение “api”.

В результате такой настройки, управляющий скрипт может принимать HTTP запросы POST в свой адрес (http://localhost:8880/v1/svc/track1 в демонстрационной конфигурации). В теле такого запроса скрипт будет ожидать JSON объект, содержащий обязательное булевское поле “recognize”, принимающее значение true для начала распознавания, и false для окончания распознавания вагона. Кроме того, данный объект может содержать поле “cookie”, заполненное произвольным 64-разрядным целым числом. Это число будет впоследствии указано в результате распознавания соответствующего вагона. Поскольку результаты распознавания возвращаются асинхронно (как события, либо по запросу, но все равно вне связи с запросом, инициировавшим распознавание), -- этот механизм позволяет связать команды, которые были даны для распознавания номера конкретного вагона, с результатами распознавания этого вагона.