**Дано:**

4 видеофайла с камер, расположенных над железнодорожными путями. Размещение камер схематически представлено на рисунке 1.

4 файла-аннотации.

Название видео совпадает с названием аннотации. Одна строка аннотации характеризует один кадр видеофайла.

Видео рассинхронизированы[[1]](#footnote-1), в них пропущены некоторые кадры. Частота кадров: 5

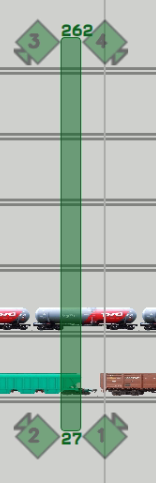


Рис. 1. Расположение камер

**Обязательное задание:**

Реализовать GUI с синхронным воспроизведением 4х видео.

*Дополнительные задания (для junior требуется реализовать одно из тестовых, middle – 2, senior - все):*

* Добавить регулировку скорости воспроизведения: 0,2х – 1к/с, 1х – 5 к/с, 10х – 50 к/с.
* реализовать клиент серверную архитектуру (сервер синхронизирует видео и отправляет кадры клиенту пачкой из 4 кадров, по одному с каждого видео; клиент читает и отображает кадры). Необходимость обусловлена запуском 4х клиентов одновременно без кратного увеличения ресурсов на чтение видео.
* добавить на изображение кадра метку, по которой пользователю можно понять, что кадр старый
* реализовать отправку видео в чат-бот (например, в телеграм) по запросу /get\_video.

**Результат** с Readme.md, где указана инструкция по разворачиванию и запуску, должен быть выложен в github.

**Срок выполнения –** 2 недели с момента получения задания.

*Инструменты:*

На ваш вкус. Рекомендуемый стек: RabbitMQ/ZMQ/Gstreamer. Язык программирования Python3 или C++.

**P.S.** Вопросы приветствуются

1. Синхронизация - сопоставление кадров по временным меткам. Если для момента времени у какого-то видео отсутствует кадр, то отобразить предыдущий кадр. При наличии кадра пропускать его нельзя. [↑](#footnote-ref-1)