**Постановка задачи**

Нужно реализовать pipeline для определения скорости автомобиля формулы 1 используя записи статичных видео камер.

**Проблема**

Для рассчета скорости движущегося автомобиля будем использовать простую формулу:

Мы будем детектировать движущийся автомобиль и замерять время, за которое он пройдет путь. Решение данной задачи разбивается на три подзадачи:

1. распознать автомобиля

1.1. с помощью обученного классификатора CascadeClassifier или

1.2. преобразовать dataset изображений машин формулы 1 в матрицу чисел с помощью гистограммы направленных компонентов (HOG) и обучить свой SVM классификатор или другой

1. запоминать автомобиль и следить за его перемещением
2. определить расстояния, перевести расстояние с изображения в координаты реального мира

**Технологии**

1. OpenCV

* чтение видео
* взаимодействие с видео для отрисовки оконтовки захваченных автомобилей
* CascadeClassifier для детектирования автомобилей
* перевести frame видео из цветного в серый
* для записи выходного видео

1.2. skimage - для преобразования фрейма в матрицу чисел с помощью HOG (можно самому написать)

sklearn - svm классификатор или другие

Dataset должен содержать достаточное количество изображений для хорошего обучения, различных вариаций, форм автомобилей. В том числе перевернутые, под разным углом фотографии для лучшего обучения модели, так как для человеческого разума определить машину легко, но для компьютера нет. Если показать перевернутую машину, это будет уже другая матрица признаков.

Также фотографии должны быть одного размера и небольшого формата, 64х64 пикселя например, чтобы процесс обучения не занял очень много времени. Никаких лишних деталей не иметь на изображении.

1. dlib

* для захвата объекта и слежением за его перемешением

1. Нужно понять сколько пикселей на изображении для каждой камеры будет в реальной жизни в метрах. Для этого либо пройтись по трассам и сделать замеры для каждой камеры, либо кажется возможным вариант найти какой-то ориентир на изображении, от которого мы сможем оттолкнуться, то есть сказать сколько метр содержит в себе пикселей на этом изображении. Наверняка для формулы 1 ширина дорог строго ограничина, поэтому зная ширину и количество пикселей на изображении ширины дороги, мы можем сказать сколько один метр в себе содержит пикселей для каждой камеры. Допустим дальше мы будем знать, pixel\_per\_meter for each camera like

*dictionary = { camera\_id: int, pixel\_per\_meter: int }*

**Сроки и технические требования компьютера**

Не думаю что модель будет долго обучаться. 20000 объектов в тренеровочном датасете хватит. 10к положительной отмеченное изображение и 10к нет.

Если будут проблемы с временем обучения, можно воспользоваться методами понижения размерности признаков PCA, LDA, etc. Или можно воспользоваться нейронной сетью YOLO3, которая по эффективности обгоняет другие методы детектирования объектов.

По времени выполнения задания, первичный код думаю можно показать уже через 2-3 дня, если все необходимые данные будут даны. Полностью задачу думаю за неделю можно выполнить.

**Алгоритм и код**

<https://github.com/MAGLeb/Interesting/blob/master/test_tasks/assian/vechicle%20car%20detection%20speed.ipynb>

The problem

Design a system for analysing speed of F1 cars of your team’s competitors.

Background: you are a computer vision engineer at one of the F1 teams. During the

race, your pilot needs to know the speed of competitors’ cars, but they are hiding

this information. Your team has decided you will use computer vision for this problem.

Resources available: several static (not moving) RGB cameras are set up statically at

different parts of the track.

You have unlimited access to the cameras input before the race, as well as similar

footage of historic races, but on different tracks.

System output: speed for every car that is visible by our cameras at the given moment.

What needs to be done

We expect a document with the proposed solution to the task. It must be submitted in 24 hours max. You

can write the text in English or Russian (or mix it).

Example:

The whole solution pipeline consists of subtasks A, B, C:

1. To solve the subtask A, use technology X:

<what exactly will be done...>

<describe how you would collect and preprocess data and why...>

<describe which programming technology and framework you would use and why...>

<how much time it would take you...>

<system requirements...>

<possible obstacles that might prevent meeting the deadline and how to overcome

them...>

<plan B: what if your solution doesn’t work when implemented...>

<resources (e.g. more data) required for possible enhancements...>

<an illustration/figure>

<little code samples if it can help explaining the solution to us>

2. To solve the subtask A, use technology Y

....

All details are optional and you are free to choose your own format.

Also, we don’t expect you to write the code unless you really want to.

There will be a separate interview for evaluating your coding skills.

Evaluation

Consists of two steps: we first evaluate the written essay and in the case of success ask additional

questions during a live interview.