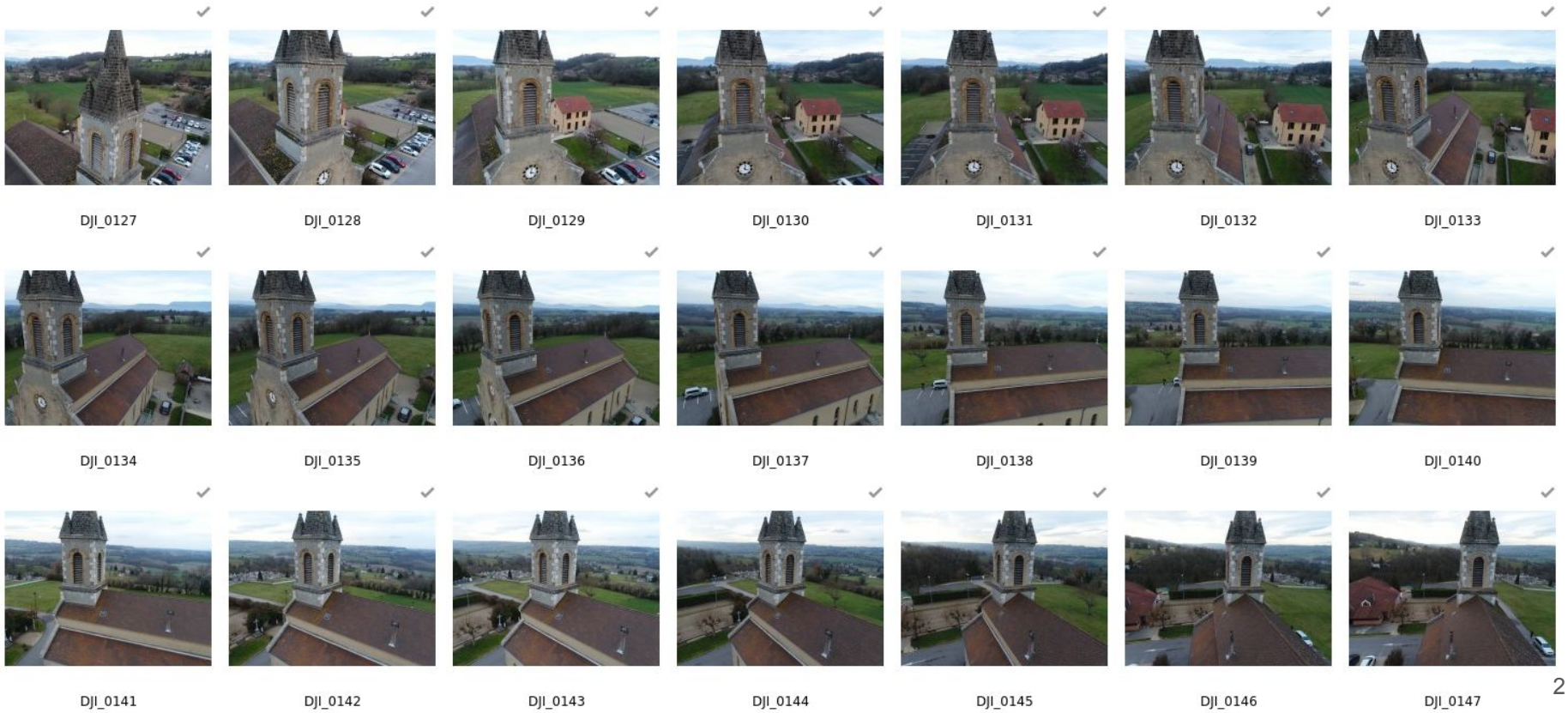
Agisoft Metashape

Полярный Николай
polarnick@agisoft.com

Metashape

Основная задача:

По множеству фотографий восстановить трехмерную модель.



Metashape

Основная задача:
По множеству фотографий восстановить трехмерную модель.

Данные предоставил Stéphane Prodent

Предложенные задачи

1. Ускорение сопоставления изображений (BoW)
2. 3D реконструкция по видео (оптический поток)

1. Ускорение сопоставления изображений (BoW)

В **Metashape** как и в любом **structure from motion** все начинается с неструктурированной кучи изображений.

Требуется понять какие из них наблюдают одинаковые поверхности.

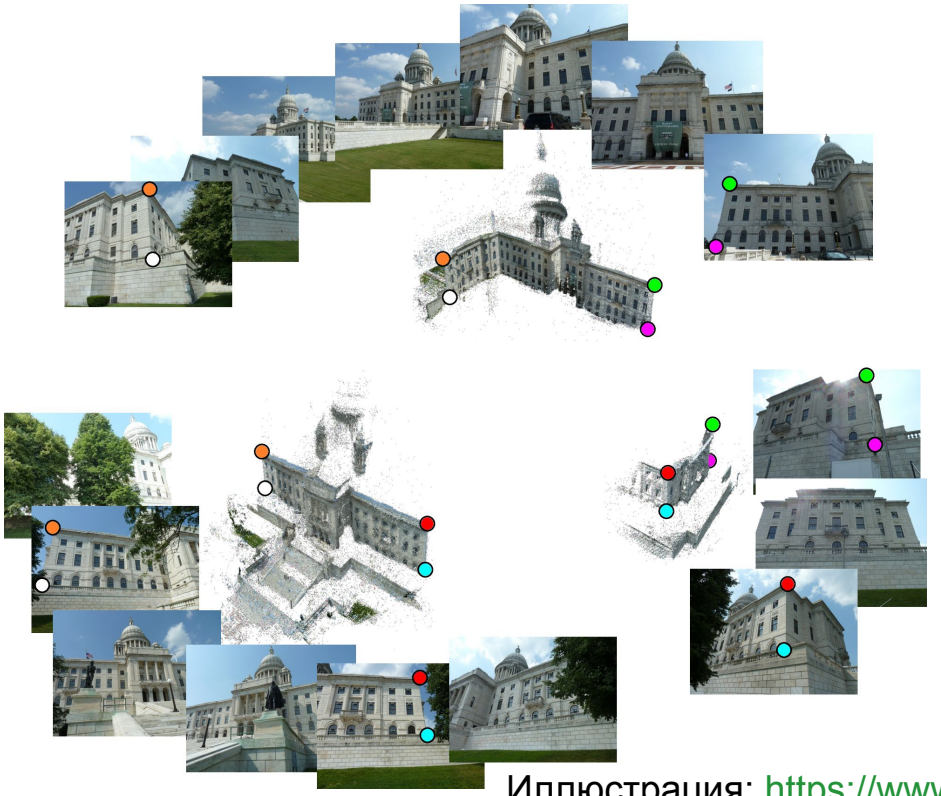


Иллюстрация: https://www.cvg.ethz.ch/research/

1. Ускорение сопоставления изображений (BoW)

1) Детектируются характерные инвариантные точки
2) Проверяется сопоставление для каждой пары изображений
3) Успешно сопоставленные изображения наблюдают одно и то же

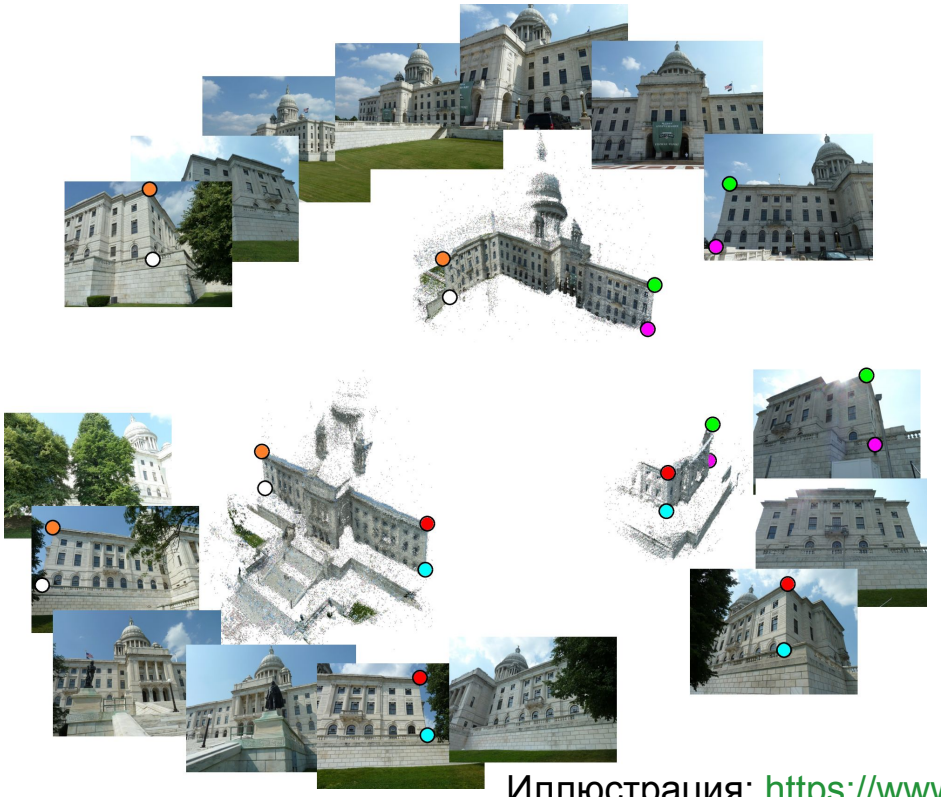


Иллюстрация: https://www.cvg.ethz.ch/research/

1. Ускорение сопоставления изображений (BoW)

1) Детектируются характерные инвариантные точки
2) Проверяется сопоставление **для каждой пары изображений**
3) Успешно сопоставленные изображения наблюдают одно и то же

$$O(N^2)$$

А изображений тысячи и десятки тысяч!

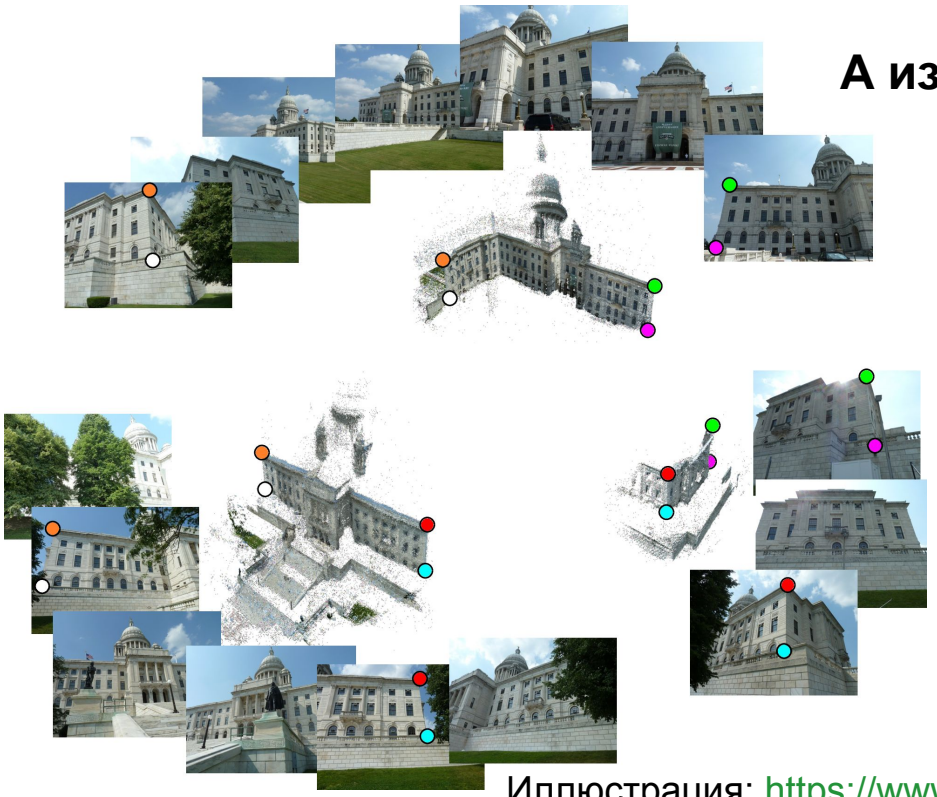


Иллюстрация: https://www.cvg.ethz.ch/research/

1. Ускорение сопоставления изображений (BoW)

Хочется вместо сопоставления каждой пары сопоставлять лишь те пары у которых есть хоть какие-то шансы.

На данный момент используется **грубое предварительное сопоставление** каждой пары в меньшем разрешении.

1. Ускорение сопоставления изображений (BoW)

Хочется вместо сопоставления каждой пары сопоставлять лишь те пары у которых есть хоть какие-то шансы.

На данный момент используется **грубое предварительное сопоставление** каждой пары в меньшем разрешении.

Дает существенное **неасимптотическое** ускорение.

Но хочется быстрее.

1. Ускорение сопоставления изображений (BoW)

Можно хранить некоторый словарь ключевых точек по всем изображениям, и очередную картинку пытаться сопоставить не с каждой картинкой независимо, а **сразу со всеми картинками** (посредством неточного сравнения со словарем, глобальным по всем фотографиям).

Такой подход называется **Bag-of-Words [1] [2] [3] [4] [5]:**

- **kd-tree** поверх сжатого пространства дескрипторов ключевых точек
- **подсчет веса “неповторимости”** и оттого “различительной способности” конкретного дескриптора

1. Ускорение сопоставления изображений (BoW)

Задача:

- **Попробовать DBoW2 на больших реальных наборах фотографий**
- **Как часто и почему пары теряются?**
- **Вытянуть максимальное качество (т.к. offline обработка, а не SLAM)**

Попутные задачи:

- Кодить на **C++** и **Python**
- Использовать **OpenCV** для ключевых точек и дескрипторов
- **Читать статьи**
- Визуализировать граф сопоставлений изображений
- Как быстро работает?
- Какие ключевые точки и дескрипторы лучше всего работают?
- Как соотносится с другими реализациями?
- Правда ли что **approximate nearest neighbour** неприменим? **[6]**

1. Ускорение сопоставления изображений (BoW)

Ссылки:

[1] Building Rome in a Day, Agarwal et al., 2009

[2] Bags of Binary Words for Fast Place Recognition in Image Sequences, 2012

[3] https://github.com/dorian3d/DBoW2

[4] https://github.com/raulmur/ORB_SLAM2

[5] https://github.com/tum-vision/lsd_slam

[6] New approximate nearest neighbor benchmarks

2. 3D реконструкция по видео (оптический поток)

На данный момент в Metashape видео можно автоматически нарезать на кадры с фиксированной частотой и обработать их как обычные фотографии.

Данные предоставил Simon Brown

2. 3D реконструкция по видео (оптический поток)

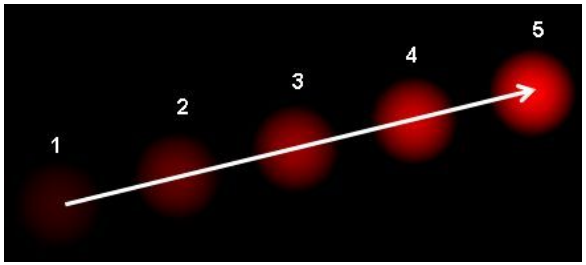
- Кадров может быть взято **слишком много/слишком мало**
- Ключевые точки **могут не сопоставиться** т.к. кадры нечеткие

Данные предоставил Simon Brown

2. 3D реконструкция по видео (оптический поток)

Предлагается учесть специфику видео:

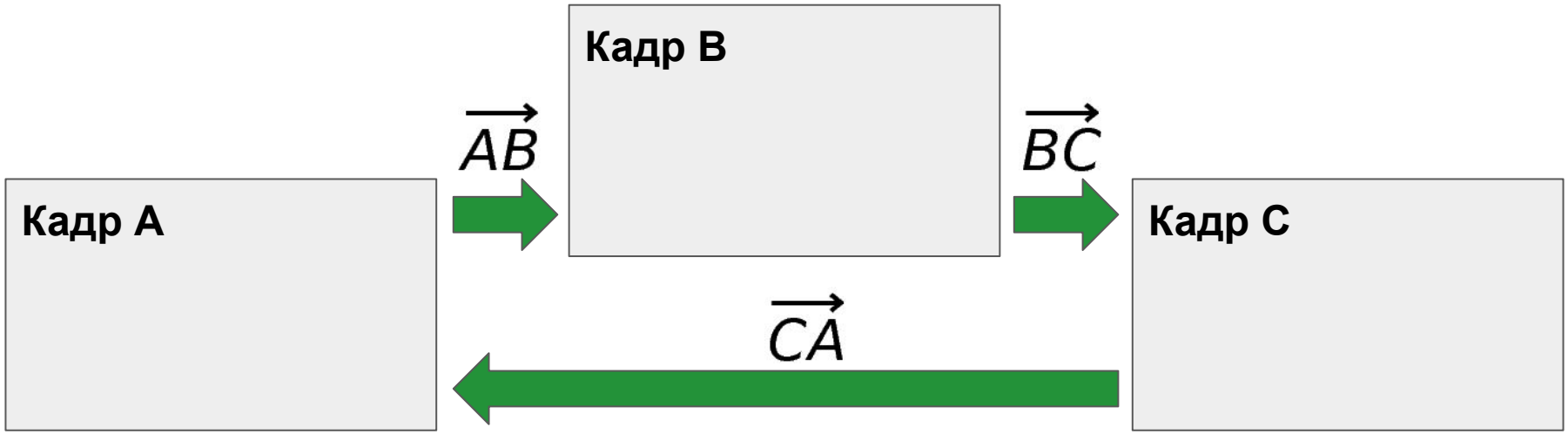
От кадра к кадру картинка меняется слабо, поэтому можно на попиксельной основе найти сопоставления между последовательными кадрами **методом оптического потока**: **[1] [2]**



2. 3D реконструкция по видео (оптический поток)

Проблема: плотный оптический поток не признает ошибок, в результате в каждом пикселе найдет хоть что-то, **и часто это что-то - неверно**.

Идея: давайте найдем поток **A**⟶**B**, **B**⟶**C** и **C**⟶**A**, и проверим на согласованность:



Т.е. пусть оптический поток в пикселе найден верно если для этого пикселя:

$$\|\overrightarrow{AB} + \overrightarrow{BC} + \overrightarrow{CA}\| < \varepsilon$$

2. 3D реконструкция по видео (оптический поток)

Задача:

- Оптическим потоком по видео сопоставить точки по адаптивному подмножеству кадров **[1] [2]**
- Если сдвиги слишком большие/маленькие - брать кадры чаще/реже
- Посмотреть какие другие идеи можно почерпнуть из SLAM алгоритмов **[3] [4]**

Попутные задачи:

- Кодить на **C++**
- Использовать **OpenCV** как минимум для первого прототипа **[2]**
- **Читать статьи**
- Как быстро работает? Найти или написать оптический поток на GPU?
- Какие ключевые точки и дескрипторы лучше всего работают?
- Сравнить со state-of-the-art **SLAM** реализациями **[3] [4]**

2. 3D реконструкция по видео (оптический поток)

Ссылки:

[1] wiki/Optical flow

[2] OpenCV/Dense optical flow

[3] https://github.com/raulmur/ORB_SLAM2

[4] https://github.com/tum-vision/lsd_slam

Организационные детали

- Язык: **C++** (для первой задачи еще **Python**)
- Адрес офиса: Дегтярный Переулок, 11 лит. Б
- С любыми вопросами можно писать на polarnick@agisoft.com или http://t.me/PolarNick239
- К предложенным темам есть тестовые задания - вышлю по запросу
- Во всех практиках нужно будет читать статьи

Вопросы?

Agisoft

Полярный Николай

polarnick@agisoft.com

Предложенные задачи

1. Ускорение сопоставления изображений (BoW)
2. 3D реконструкция по видео (оптический поток)

Конец

На следующих слайдах тема которая скорее-всего будет предложена в будущем, но наша пропускная способность - два студента, поэтому предложить три темы не готовы.

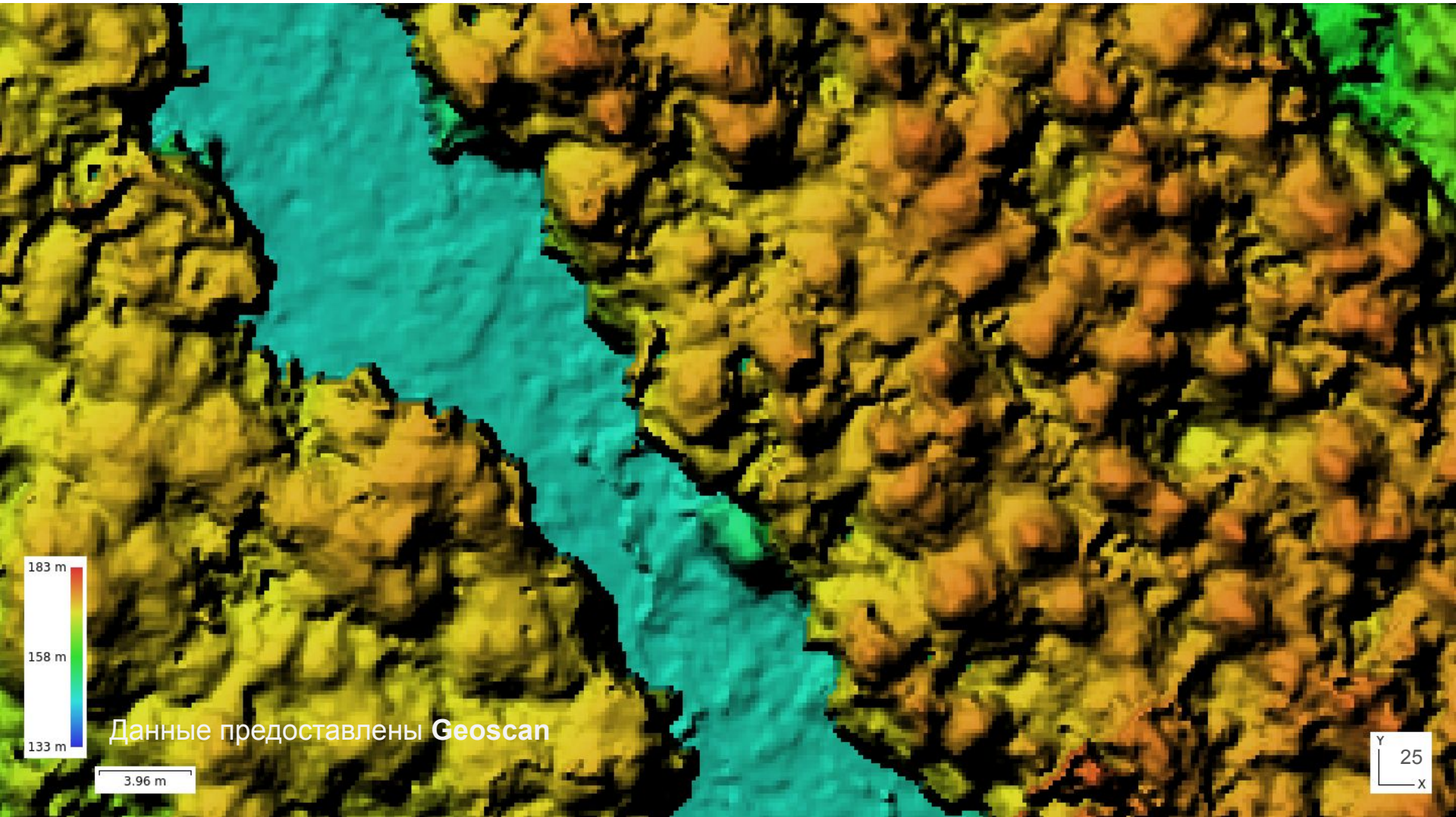
Т.е. дальше слайдов нет, хоть они и есть.

3. Подсчет деревьев (классическими методами)

Иллюстрация: https://blog.dronedeploy.com/product-release-wrap-up-april-2019-9a67f8bbfd0f

3. Подсчет деревьев (классическими методами)

3. Подсчет деревьев (классическими методами)



3. Подсчет деревьев (классическими методами)

Предполагается **по карте высот** обнаружить верхушки деревьев и таким образом подсчитать примерное число деревьев:

1) Взять карту высот
2) Сгладить ее гауссом с радиусом свертки примерно равным радиусу дерева
3) Выделить локальные экстремумы
4) Оставить из них те, чья окрестность похожа на верхушку дерева (параболоиду)

Это никогда не будет работать во всех случаях, но попытаться как-то поддержать хотя бы хорошие случае хочется. И вдохновляясь идеями из статей попытаться зайти как можно дальше.

Дальше предполагается использовать этот алгоритм как трамплин (генератор обучающей выборки) для методов машинного обучения.