Лекция

Тема: OpenCv. Object detection.
Интересные девайсы: Kinect,Leap

Готовые реализации методов распознавания образов

- cvCalcBackProjectPatch
- CvMatchTemplate — сравнение с образцом
- CvHaarCascade — каскады Хаара (WaveLet)
- Surf
- Shift

CvMatchTemplate

- Функция сравнения регионов изображения

```
void cvMatchTemplate(
        const CvArr* image,
        const CvArr* templ,
        CvArr* result,
        int method
);
```

Image — изображение,где будет осуществляться поиск

Templ — искомый шаблон изображения (меньшей размерности,чем Image)

Result — карта результатов сравнения (массив (W-w+1)x(H-h+1);

Method: Метод сравнения.

Ключевой алгоритм: скользящее по изображению окно. В процессе движения выполняется вычисление коэффициента корреляции **перекрывающихся участков**.

Metod

a. **method=CV_TM_SQDIFF** Квадрат отклонения

$$R(x,y) = \sum_{x',y'} (T(x',y') - I(x+x',y+y'))^2$$

b. **method=CV_TM_SQDIFF_NORMED** Квадрат отклонения,нормированный

$$R(x,y) = \frac{\sum_{x',y'} (T(x',y') - I(x+x',y+y'))^2}{\sqrt{\sum_{x',y'} T(x',y')^2 \cdot \sum_{x',y'} I(x+x',y+y')^2}}$$

c. **method=CV_TM_CCORR** Корелляция

$$R(x,y) = \sum_{x',y'} (T(x',y') \cdot I(x+x',y+y'))$$

d. **method=CV_TM_CCORR_NORMED** Корелляция,нормировнная

$$R(x,y) = \frac{\sum_{x',y'} (T(x',y') \cdot I'(x+x',y+y'))}{\sqrt{\sum_{x',y'} T(x',y')^2 \cdot \sum_{x',y'} I(x+x',y+y')^2}}$$

Method

e. **method=CV_TM_CCOEFF**

$$R(x,y) = \sum_{x',y'} (T'(x',y') \cdot I(x+x',y+y'))$$

where

$$T'(x',y') = T(x',y') - 1/(w \cdot h) \cdot \sum_{x'',y''} T(x'',y'')$$
$$I'(x+x',y+y') = I(x+x',y+y') - 1/(w \cdot h) \cdot \sum_{x'',y''} I(x+x'',y+y'')$$

f. **method=CV_TM_CCOEFF_NORMED**

$$R(x,y) = \frac{\sum_{x',y'} (T'(x',y') \cdot I'(x+x',y+y'))}{\sqrt{\sum_{x',y'} T'(x',y')^2 \cdot \sum_{x',y'} I'(x+x',y+y')^2}}$$

CvMatchTemplate

- После завершения функции наилучший результат находится с помощью функции cvMinMaxLoc, причем как минимум для SQDIFF-методов, а для остальных как максимум. В случае цветного изображения каналы суммируются.
- Поскольку функция перебирает все возможные результаты, то работает довольно медленно
- Функция чувствительна к повороту,искажению,масштабированию изображения.

http://docs.opencv.org/doc/tutorials/imgproc/histograms/template_matching/template_matching.html

1. Testing our program with an input image such as:

and a template image:

Резюме

- Поскольку алгоритм не инвариантен к наклону и масштабированию в реальной жизни придется использовать дополнительные механизмы приведения объектов к подобному(размер) виду, либо выполнять несколько проходов с разными масштабами(медленно).

Каскады Хаара

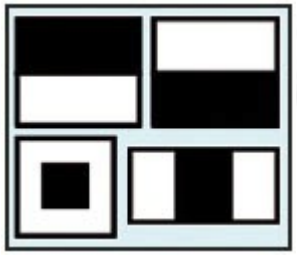
- Для поиска объектов по их ключевым признакам используется метод П.Виолы и М.Джонса. Подход использует 4 ключевых концепции

  - Простые прямоугольные функции, называемые функциями Хаара.
  - Интегральное Изображение для упрощения поиска.
  - Метод машинного обучения AdaBoost.
  - Каскадный классификатор для эффективного совмещения множественных функций.

  Особенности, которые использовали Виола и Джонс, базируются на вейвлетах Хаара. Вейвлеты Хаара представляют собой прямоугольные волны одинаковой длины (один высокий интервал и один низкий интервал). В двух измерениях, прямоугольная волна являет собой пару соседних прямоугольников – один светлый и один темный.

Функции Хаара

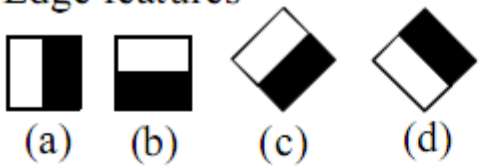
Фактически прямоугольные комбинации, используемые для визуального обнаружения объекта не являются подлинными вейвлетами Хаара. Вместо этого, они содержат прямоугольные комбинации, которые лучше подходят для визуальных задач распознавания. Из-за этой разницы, эти функции называются функциями Хаара (или Хаар-подобными функциями), а не вейвлетами. Рисунок 1 показывает те функции, что используются в OpenCV.



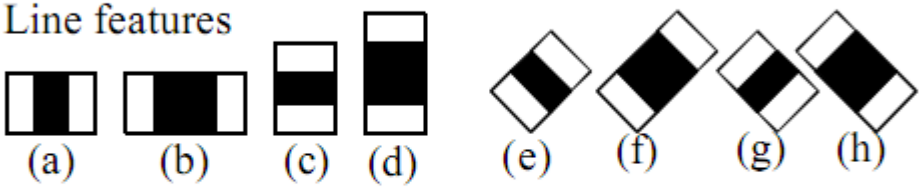
- Наличие функции Хаара определяется посредством вычитания среднего значения области темных пикселей из среднего значения области светлых пикселей. Если разница превышает порог (определяется в процессе обучения), тогда говорят, что функция является существующей.

Haar-like features

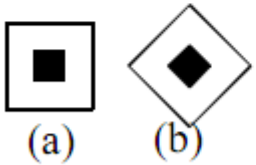
1. Edge features



2. Line features



3. Center-surround features



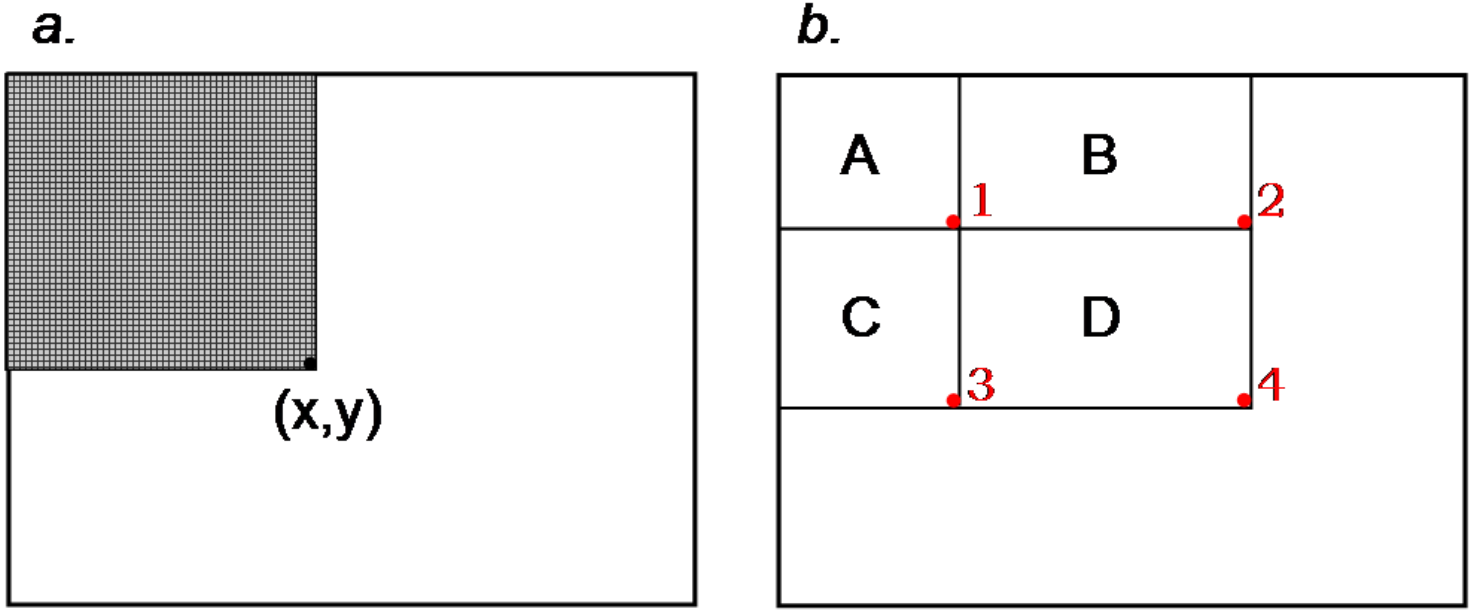
4. Special diagonal line feature used in [3,4,5]

Интегральное изображение

Чтобы эффективно определить наличие и отсутствие сотен функций Хаара на каждой локации изображении и в нескольких масштабах, Виола и Джонс (2001) использовали технологию «интегрального изображения». В общем, «интеграция» означает сложение маленьких блоков вместе.

- Интегральное значение для каждого пикселя есть сумма всех пикселей над ним и слева от него. Начиная с левого верхнего угла и совершая обход вправо и вниз, все изображение может быть интегрировано с несколькими целочисленными операциями на пиксель(его значение).

Интегральное изображение

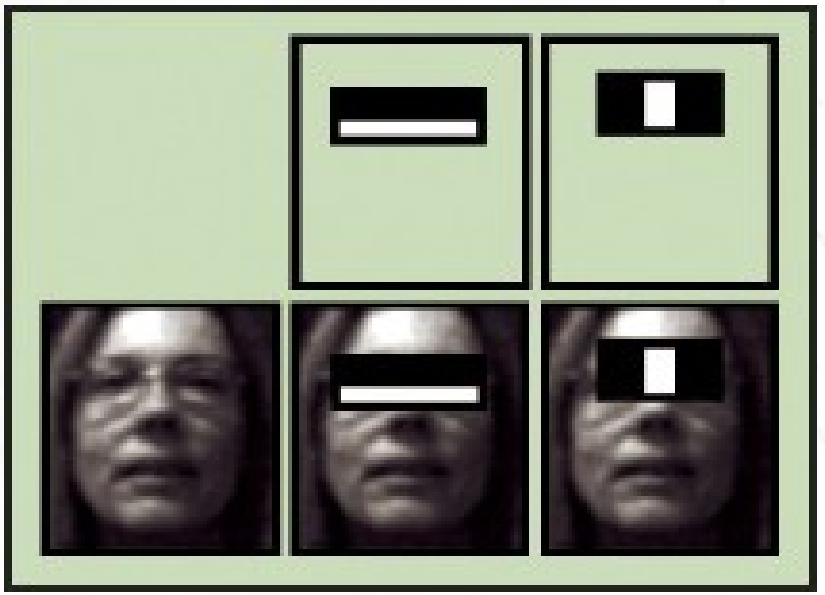


- Особенность «интегрального изображения». a) После интеграции, пиксель (x,y) содержит сумму всех пиксельных значений в заштрихованном прямоугольнике. b) Сумма пиксельных значений в прямоугольнике D есть (x4,y4)-(x2,y2)-(x3,y3)+(x1,y1).

Свойства

- Удобно, что A+B+C+D является значением «интегрального изображения» в положении 4, A+B есть значение в положении 2, A+C – значение в положении 3, и A – значение в положении 1. Так, с Интегральным Изображением, можно найти сумму пиксельных значений для любого прямоугольника в первоначальном изображении всего тремя целочисленными операциями: (x4,y4)-(x2,y2)-(x3,y3)+(x1,y1).

- Главная и основное направление использования — детектирование лиц.
- Значимые переходы- глаз-нос-глаз
- Лоб-глаза-брови



- Наличие функции Хаара определяется посредством вычитания среднего значения области темных пикселей из среднего значения области светлых пикселей. Если разница превышает порог (определяется в процессе обучения), тогда говорят, что функция является существующей.

Каскады классификаторов

- Термин «каскады» вытекает из структуры организации «образцов». Набор классификаторов состоит из множества «слабых» классификаторов с целью создания одного «сильного» - метод машинного обучения под названием AdaBoost.

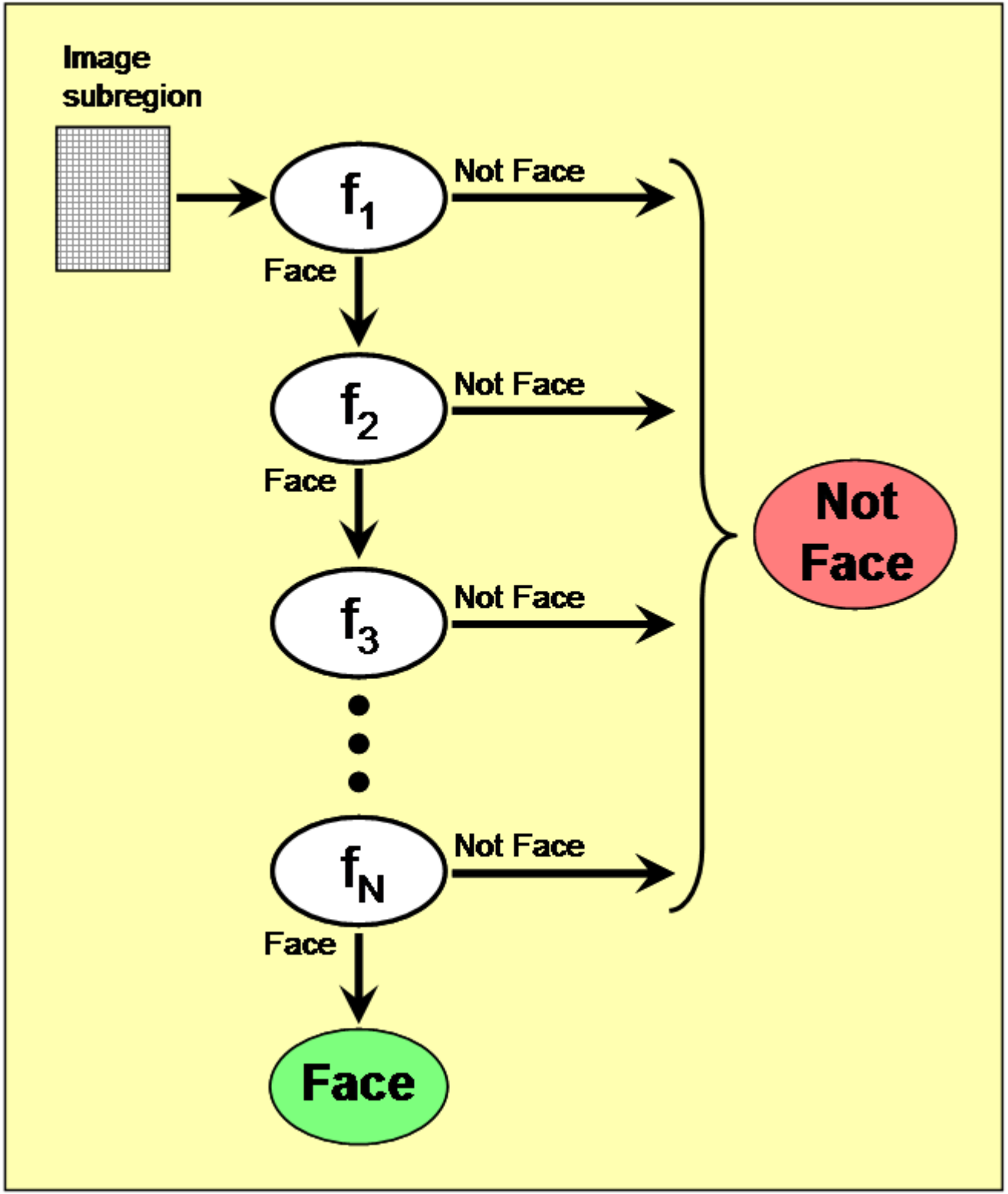


Image subregion
f_1
Not Face
Face
f_2
Not Face
Face
f_3
Not Face
f_N
Not Face
Face
Not Face
Face

Принятый порог на каждом уровне устанавливается достаточно низким, чтобы пройти все (или почти все) лицевые образцы в тренировочном наборе. Фильтры на каждом уровне обучены классифицировать тренировочные изображения, которые прошли все предыдущие этапы (обучающая выборка являет собой большую базу лиц, может быть, около тысячи или близко к этому). Во время работы, если какой-то любой из этих фильтров не пропускает область изображения, то тогда область сразу же классифицируется как «не лицо». Когда фильтр пропускает область изображения, она переходит к следующему фильтру в последовательности. Область изображения, прошедшие через все фильтры, классифицируются как «лицо». Виола и Джонс называли это фильтрацией цепи каскада.

Adaptive Boosting(AdaBoost)

- Ada Boost- адаптивное усиление. Идея- если есть набор эталонных объектов, то можно составить один более совершенный и мощный классификатор. При этом в процессе обучения акцент делается на эталоны,которые распознаются «хуже».

Использование признаков Хаара. Этап 1.Тренировка

- Для успешного обучения потребуется большое количество «отрицательных» образцов, не содержащих искомого объекта
- База оформляется в виде каталого и индексного файла (Grayscale) и передается на вход утилите обучения

Два способа задания положительного образца

1. Единичное изображение. Потребует запуска задачи процесинга,которая создаст набор изображений с различными искажениями (createsamples)
2. База данных изображений (например, лиц)

Обучение

- Для обучения потребуется утилита haartraining

- Подробности тут:
- https://docs.google.com/document/d/14r34Pd51lKZNlfJIQVRS_3kbow1OcJBKV7wTRRAW5Vg/preview?pli=1

  Обучение весьма ресурсоёмкая задача.
- На выходе получаем XML файл.

Готовые обученные выборки

- В поставке OpenCV уже есть готовые выборки.
- haarcascade_eye_tree_eyeglasses.xml haarcascade_fullbody.xml haarcascade_mcs_lefteye.xml haarcascade_profileface.xml
- haarcascade_eye.xml haarcascade_lefteye_2splits.xml haarcascade_mcs_mouth.xml haarcascade_righteye_2splits.xml
- haarcascade_frontalface_alt2.xml haarcascade_lowerbody.xml haarcascade_mcs_nose.xml haarcascade_smile.xml
- haarcascade_frontalface_alt_tree.xml haarcascade_mcs_eyepair_big.xml haarcascade_mcs_rightear.xml haarcascade_upperbody.xml
- haarcascade_frontalface_alt.xml haarcascade_mcs_eyepair_small.xml haarcascade_mcs_righteye.xml
- haarcascade_frontalface_default.xml haarcascade_mcs_leftear.xml haarcascade_mcs_upperbody.xml

Использование (см. face.cpp)

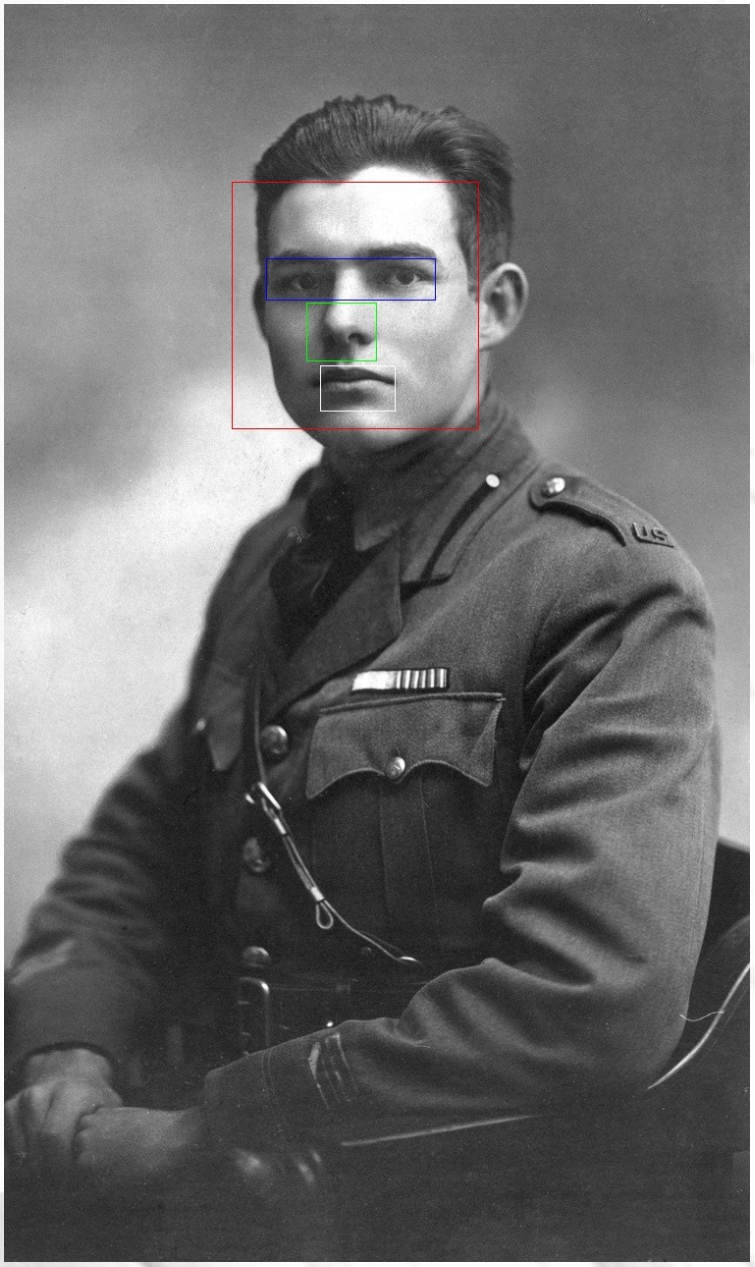
Особенности для обработки видео

1. Найденные cvRect требуют освобождения, в противном случае это приведет к «протеканию» памяти

2. Нет смысла передавать цвет, лучше поработать с контрастом.

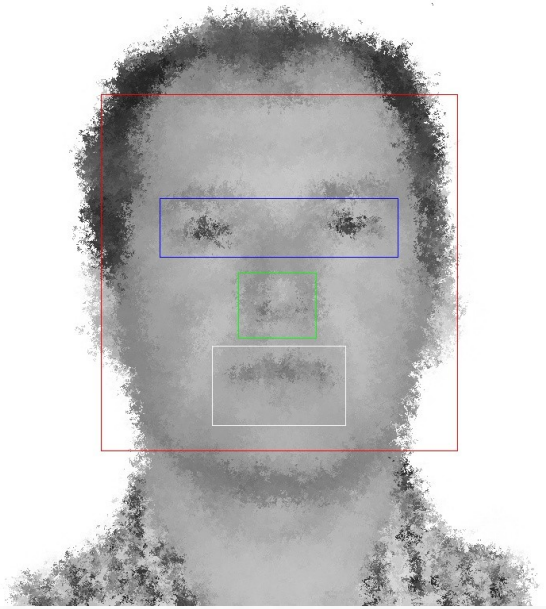
3. Чувствительность к повороту возникает примерно от 20 градусов.

На фото Эрнест Хэмингуэй 1918



Границы использования

- «Пороговая» сущность алгоритма дает возможность работать даже при весьма серьезных шумах, причем «плохое» изображение подчас лучше «хорошего»



Анализ особых точек изображения

- SURF Further Information Refer to "SURF: Speed-Up Robust Feature" * Author: Liu Liu * liuliu.1987+opencv@gmail.com

- Позволяет найти объект на изображении по заданному образцу методом анализа особых точек.
- Идея: выделить на образце особые точки, а потом искать их в обрабатываемом изображении.


Product of Italy
Object


Product of Italy
Crisp Almond Cookies
I famosi
BASTONCINI
Cantuccini
alla Mandorla
PAN DUCALE
Peso Netto 250 g
NET WT 8.8 OZ
Object Correspond

Источник материалов

- Для описания алгоритма воспользуемся великолепной статьей ftp://ftp.vision.ee.ethz.ch/publications/articles/eth_biwi_00517.pdf и ее интерпретацией http://habrahabr.ru/post/103107/

Этап1. Поиск особых точек

SURF

Метод ищет особые точки при помощи матрицы частных произвоных- матрицы Гессе

$$H(f) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1\, \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1\, \partial x_n} \\ \frac{\partial^2 f}{\partial x_2\, \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2\, \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n\, \partial x_1} & \frac{\partial^2 f}{\partial x_n\, \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix}$$

если гессиан положительно определён, то — точка локального минимума функции ,

если гессиан отрицательно определён, то — точка локального максимума функции ,

если гессиан не является знакоопределённым (принимает как положительные, так и отрицательные значения) и невырожден , то — седловая точка функции .

SURF

- Детерминант матрицы Гессе (т.н. гессиан) достигает экстремума в точках максимального изменения градиента яркости. Он хорошо детектирует пятна, углы и края линий.
- Гессиан инвариантен относительно вращения. Но не инвариантен масштабу. Поэтому SURF использует разномасштабные фильтры для нахождения гессианов.
- Для каждой ключевой точки считается направление максимального изменения яркости (градиент) и масштаб, взятый из масштабного коэффициента матрицы Гессе.
- Градиент в точке вычисляется с помощью фильтров Хаара(уже было).
- После нахождения ключевых точек, SURF формирует их дескрипторы. Дескриптор представляет собой набор из 64(либо 128) чисел для каждой ключевой точки. Эти числа отображают поведение градиента вокруг ключевой точки.

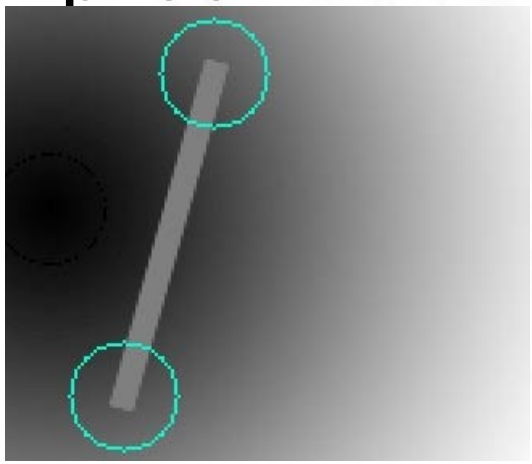
Для двумерного варианта

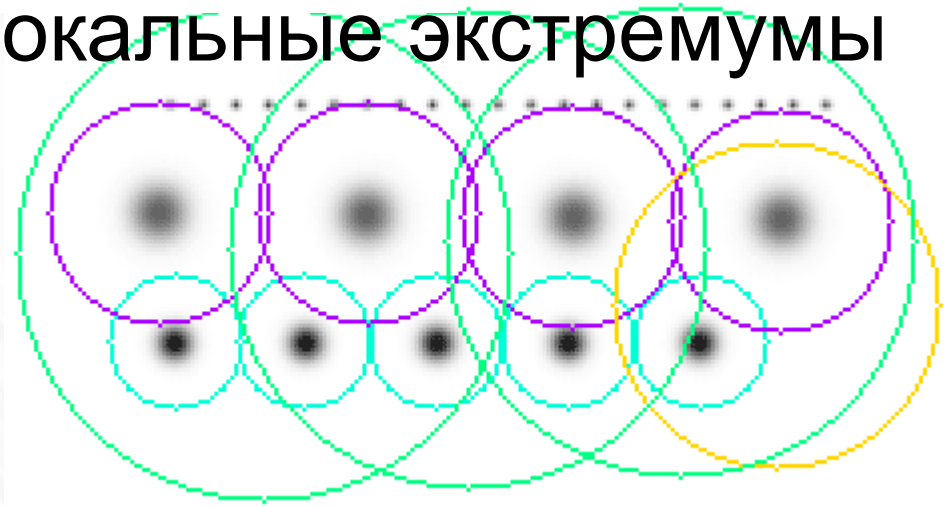
- Матрица Гессе и гессиан можно подсчитать по формуле

$$H(f(x,y)) = \begin{bmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x \partial y} \\ \frac{\partial^2 f}{\partial x \partial y} & \frac{\partial^2 f}{\partial y^2} \end{bmatrix}$$

- $$det(H) = \frac{\partial^2 f}{\partial x^2}\frac{\partial^2 f}{\partial y^2} - \left(\frac{\partial^2 f}{\partial x \partial y}\right)^2$$

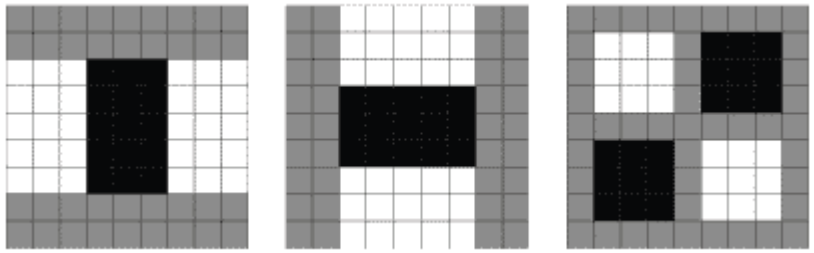
- Особые точки-локальные экстремумы яркости:





Вычисление матрицы Гессе

- Можно вычислить сверткой гауссиана (см. предыдущие лекции), но SURF использует бинаризированную версию (Fast-Hessian)



White=+1,black-2(-1), gray=0;

Гессиан вычисляется по результату 3х сверток: $det(H_{approx}) = D_{xx}D_{yy} - (0.9D_{xy})^2$

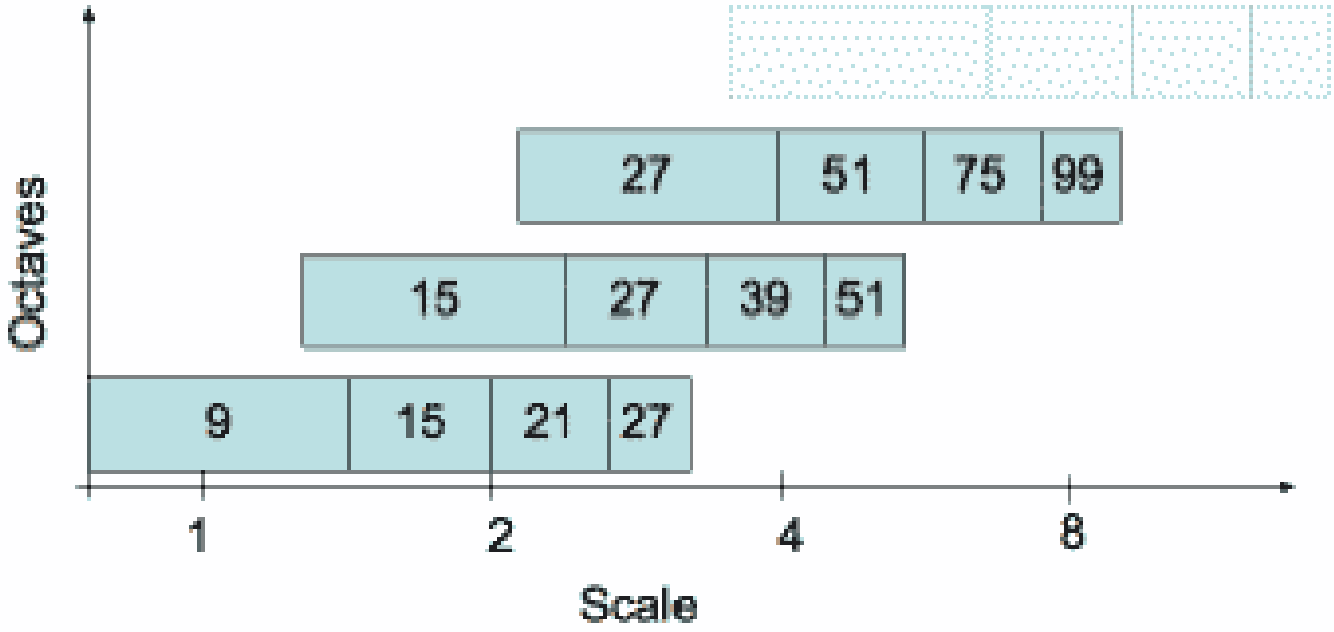
Проблема

- Гессиан инвариантен к повороту,но не к масштабу
- Алгоритм Fast-Hessian позволяет квантовать шаг 9,15,21,27 и далее с шагом 6

  Пример: две ключевые точки разного масштаба

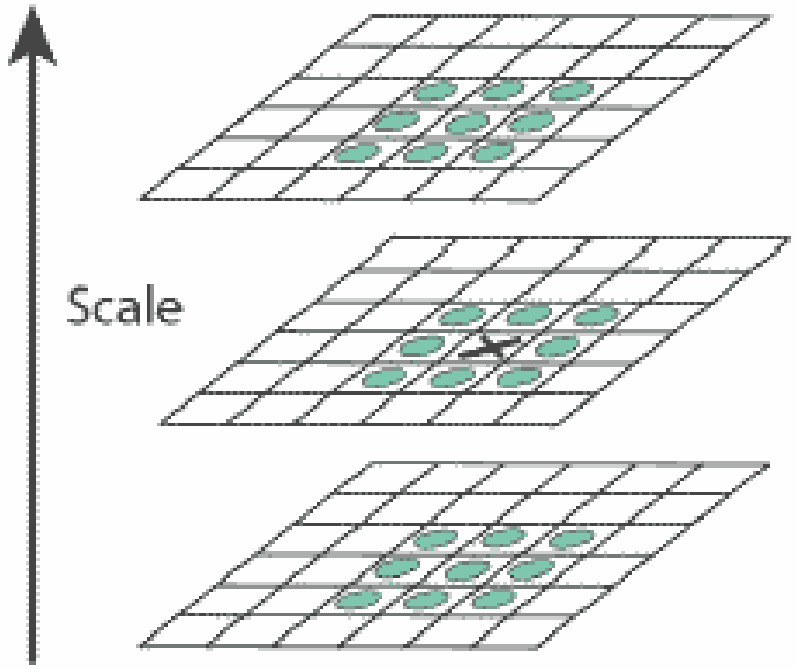
Решение: Октавы

По данным автора метода 6 октав достаточно для 1024x768



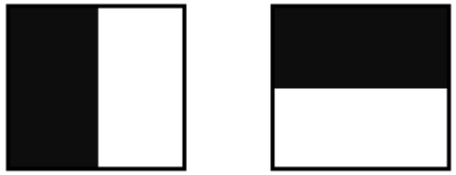
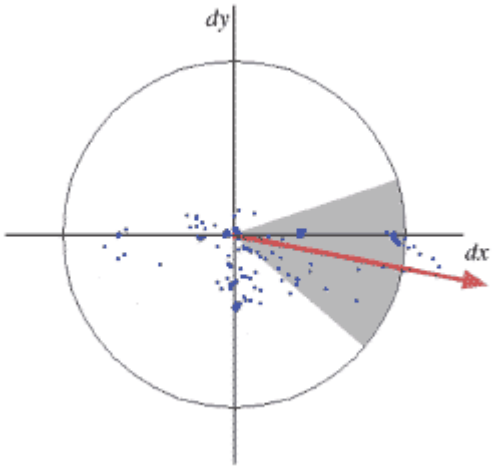
Нахождение локального максимума

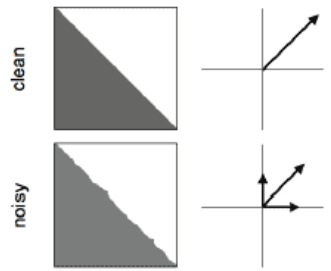
Пиксел считается локальным максимумом, если его гессиан больше чем у любого его соседа в его масштабе, а также больше любого из соседей масштабом меньше и масштабом больше (всего 26 соседей).



Градиент особой точки-инвариант к повороту

Вычислим вектор градиента особой точки, путем свертки фильтром Хаара +1/-1

- 
- 

- 

- И вращая угловое окно PI/3 найдем приоритетное направление особой точки (максимум нормы вектора)

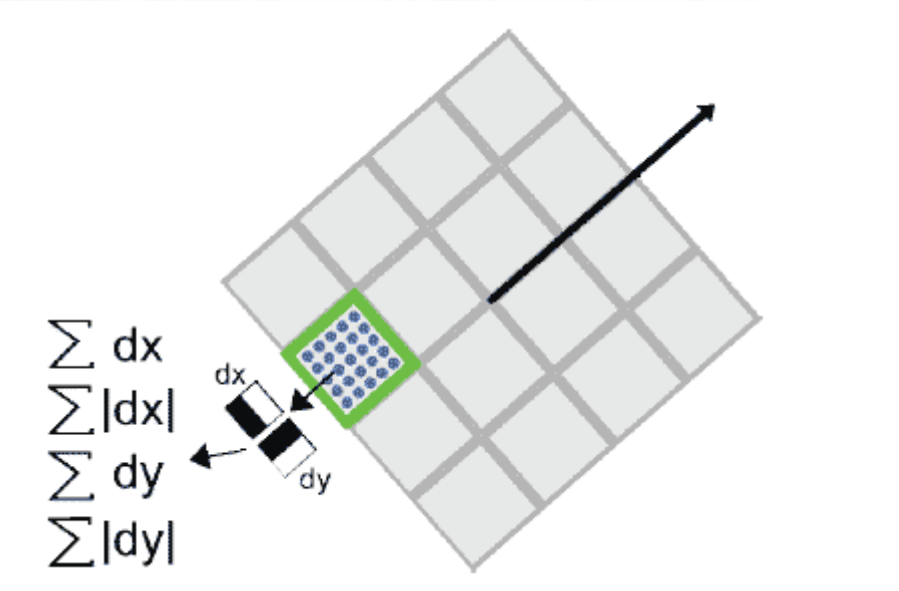
Описание особой точки

Дескриптор представляют собой массив из 64 (в расширенной версии 128) чисел, позволяющих идентифицировать особую точку. Дескрипторы одной и той же особой точки на образце и на сцене должны примерно совпадать. Метод расчета дескриптора таков, что он не зависит от вращения и масштаба.

- Для вычисления дескриптора, вокруг особой точки формируется прямоугольная область, имеющая размер 20s, где s – масштаб в котором была найдена особая точка. Для первой октавы, область имеет размер 40x40 пикселов. Квадрат ориентируется вдоль приоритетного направления, вычисленного для особой точки.
- Дескриптор считается как описание градиента для 16 квадрантов вокруг особой точки.

Повторим то же самое для 25 субзон 5x5 (2s)

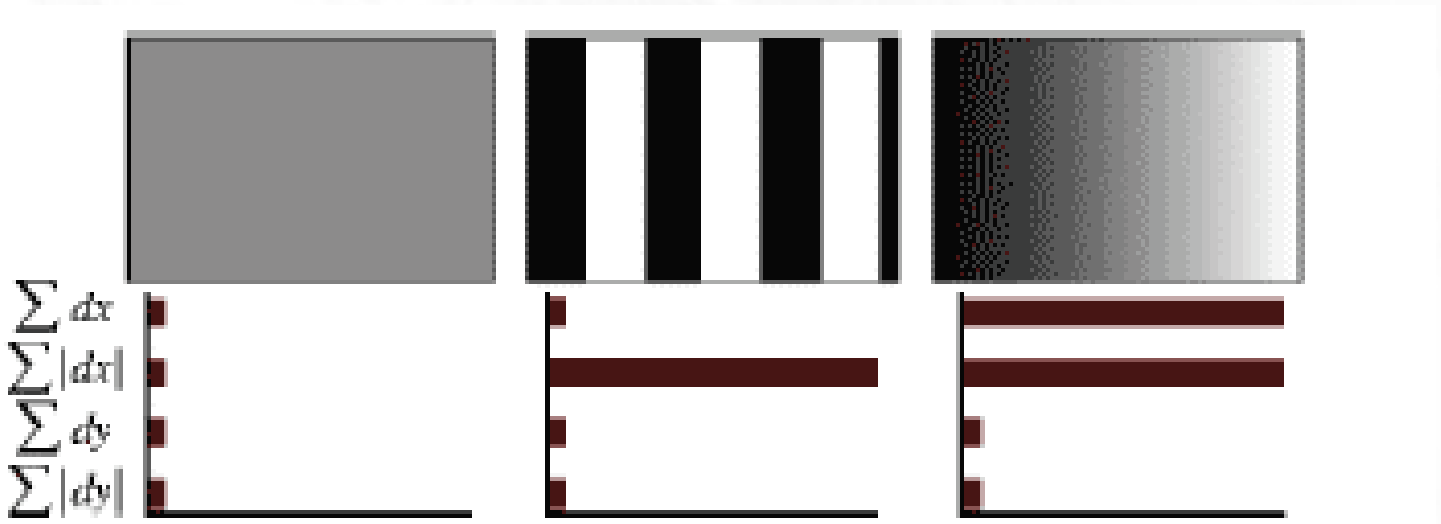
- Считаем градиент (в координатах изображения), потом поворачиваем его по ориентации основного
- Посчитали 4 значения
- ∑dX, ∑|dX|, ∑dY, ∑|dY| : 2 общих 2 суммарных



Описание

- Таким образом,описание состоит
  - Четыре компонента градиентов на каждый квадрант для 16 квадрантов (64 компонента),взвешенные на гауссиану с цетром в особой точкое и сигма 3.3
  - Знак следа(След матрицы — это сумма элементов главной диагонали матрицы) матрицы Гессе Sign(Dxx+Dyy) — дает возможность отличать светлые пятна от темных

Поведение градиентов



Summary

- SURF дает возможность искать объекты инвариантно к повороту и (относительно) к масштабу.
- В отличие от HAAR каскадов можно получить не только «Это лицо», но и «Это мое лицо», с другой стороны, нельзя найти «произвольное» лицо
- Surf не выделяет объект из фона
- Метод не будет работать для простых объектов
- Surf патентован, для коммерческого использования требуется лицензия

ORB: an efficient alternative to SIFT or SURF

- http://www.vision.cs.chubu.ac.jp/CV-R/pdf/Rublee

  Существует свободная (хоть и не такая известная) альтернатива SURF и она не одинока.

Итог

- Opencv- мощный инструмент, который позволяет решать сложные задачи готовыми, отлаженными методами
- Для «низкоуровневых» задач иногда полезно использовать математические библиотки в чистом виде, например LibGSL

Немного железа:Kinect

- Бесконтактный игровой контроллер .

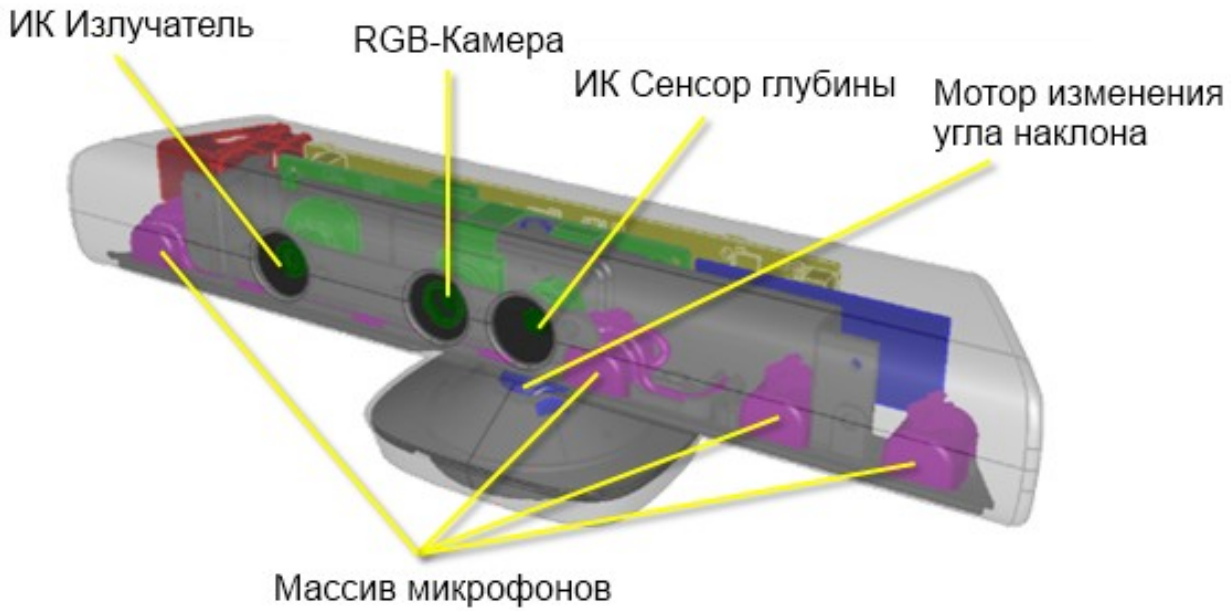
  диапазон расстояний 0.5-5м
- Точность ~1мм(0.5m)- 45мм(5m)
- Родная среда программирования: Microsoft Kinect SDK (закрыто,только MS Visual Studio)


KINECT
XBOX 360

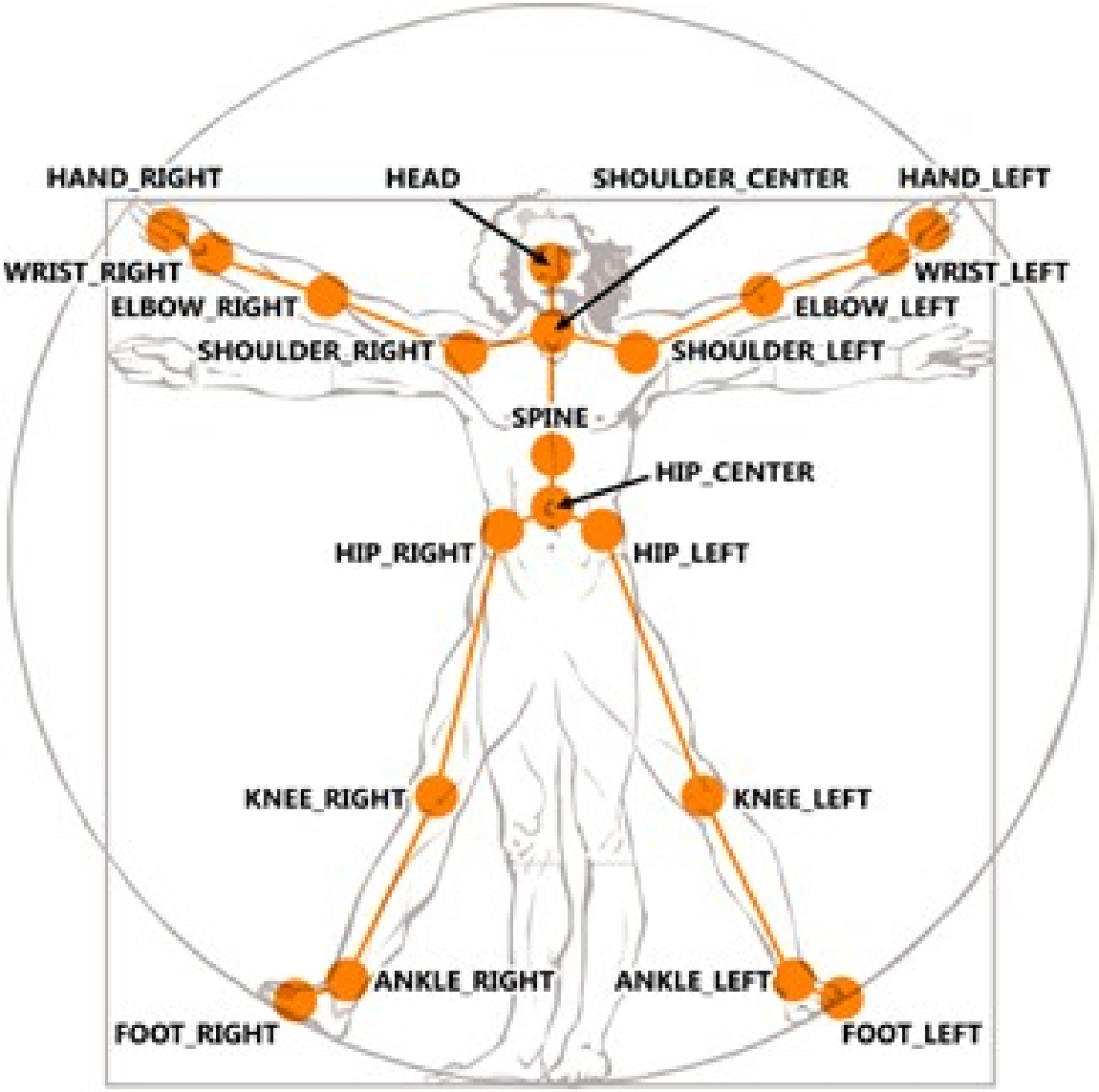
Устройство Kinect

- RGB камера
- 3D камера глубины

  (+ определение пользователя)
- Массив из 4х микрофонов
- Мотор изменения наклона девайса

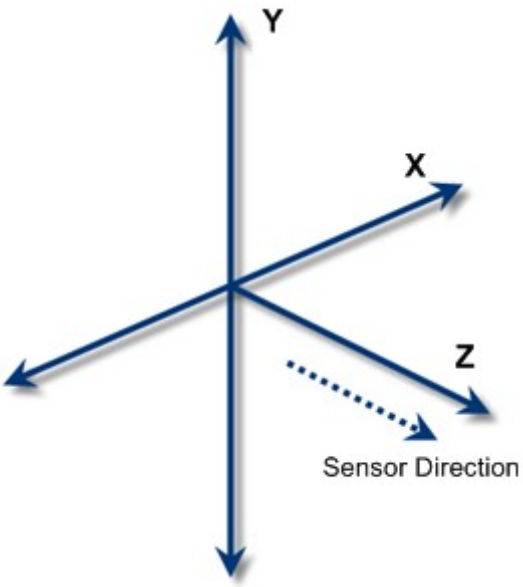


SkeletonFrame – собери меня полностью



Скелет человека и Kinect

- Скелет состоит из:
  - 20 суставов (Joint) для обычного режима
  - 10 суставов для сидячего режима
- Возможна поддержка одновременно 2х активных игроков
- Может следить одновременно за 6 игроками
- Данные, полученные из SkeletonFrame являются массивом типа Skeleton[], каждый элемент которого содержит коллекцию суставов Joints с 3D-координатами своего положения в прост-ве



Параметры сглаживания скелета

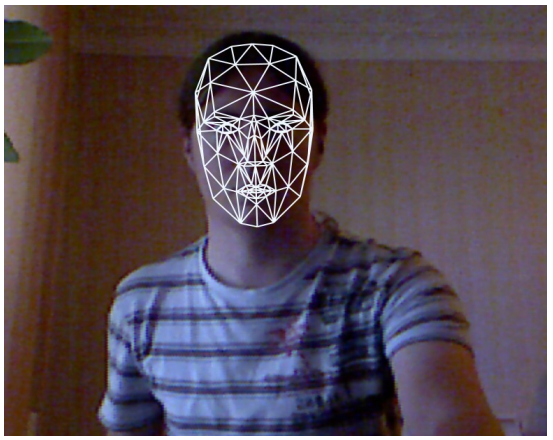
- Smoothing(сглаживание) – default 0.5
- Correction(коррекция) – default 0.5
- Prediction(прогнозирование) – default 0.5
- JitterRadius(радиус дрожания) – default 0.05
- MaxDeviationRadius(максимальный радиус отклонения) – default 0.05

Позиционирование

- Класс DepthImageFrame содержит:
- `DepthImagePoint MapFromSkeletonPoint(SkeletonPoint skeletonPoint)` — позволяет получить координаты пикселя глубины из сустава скелета
- `ColorImagePoint MapToColorImagePoint(int depthX, int depthY, ColorImageFormat colorImageFormat)` — позволяет получить координаты пикселя цвета из координат пикселя глубины
- `SkeletonPoint MapToSkeletonPoint (int depthX, int depthY)` — позволяет получить координаты сустава скелета из координат пикселя глубины
- Актуально для преобразований координат между разными системами координат.

FaceTracking API

- В основе алгоритма распознавания лежит алгоритм Active appearance model (AAM – активная модель внешности/активная видовая модель)
- При работе с Face Tracking API, разработчик посылает структуру данных, содержащую информацию с RGBA-изображением и массив данных глубин. В результате обработки этих данных, разработчик получает на выходе коллекцию 2D-координат точек элементов лица
- При сопоставлении лица и распознанных точек лица, используется Candide-3 маска

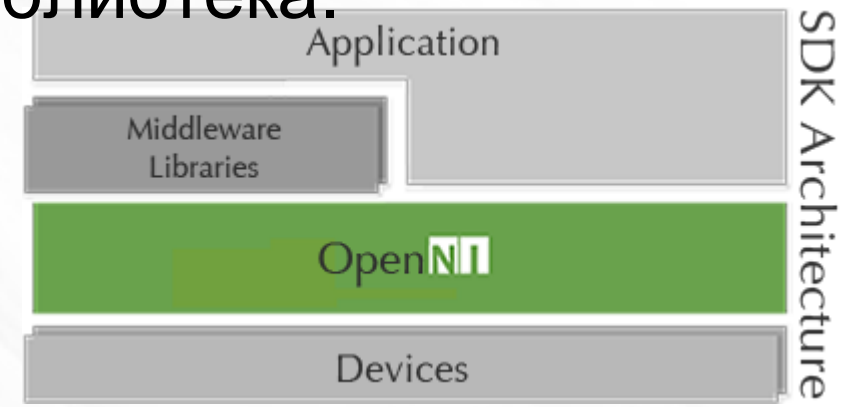


Альтернативы:

Freenect http://openkinect.org/wiki/Main_Page

- Простая компактная открытая Linux библиотека. Только получение данных с камер и датчика глубины,управление

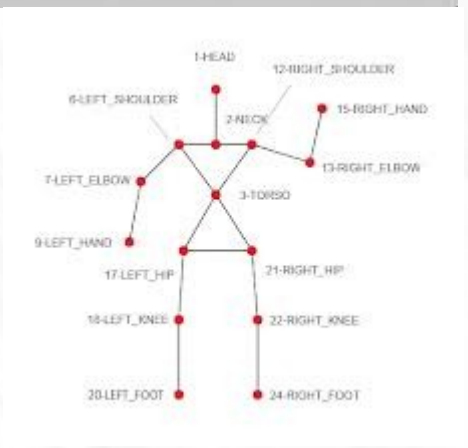
- OpenNI http://www.openni.org/
  - Весьма непростая библиотека.



Open NI

- Поддерживает:
  - Kinect
  - Asus Xion
  - (Сенсор везде один- израильская компания PrimeSense)
- Встроенные функции распознавания жестов и скелета (открытые разработчику)

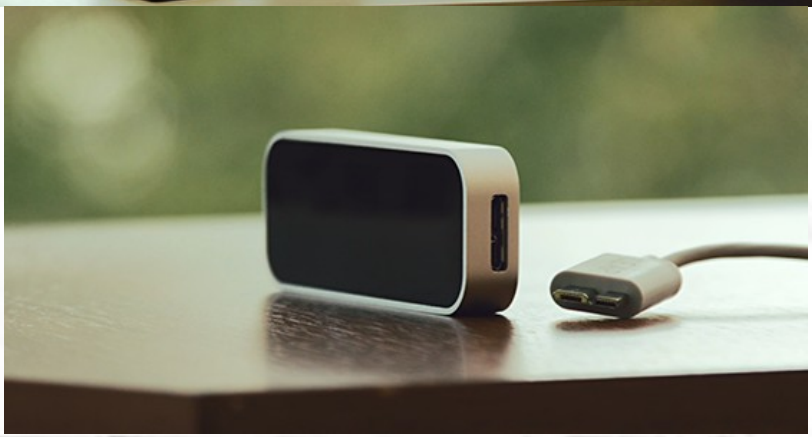
Примеры OpenNI детектирования



LeapMotion

- http://www.youtube.com/watch?feature=player_de
- https://www.leapmotion.com/

- Заточен только под жесты
- Работает с двумя руками
- Дальность до 20 см
- Родные ОС Win&Mac



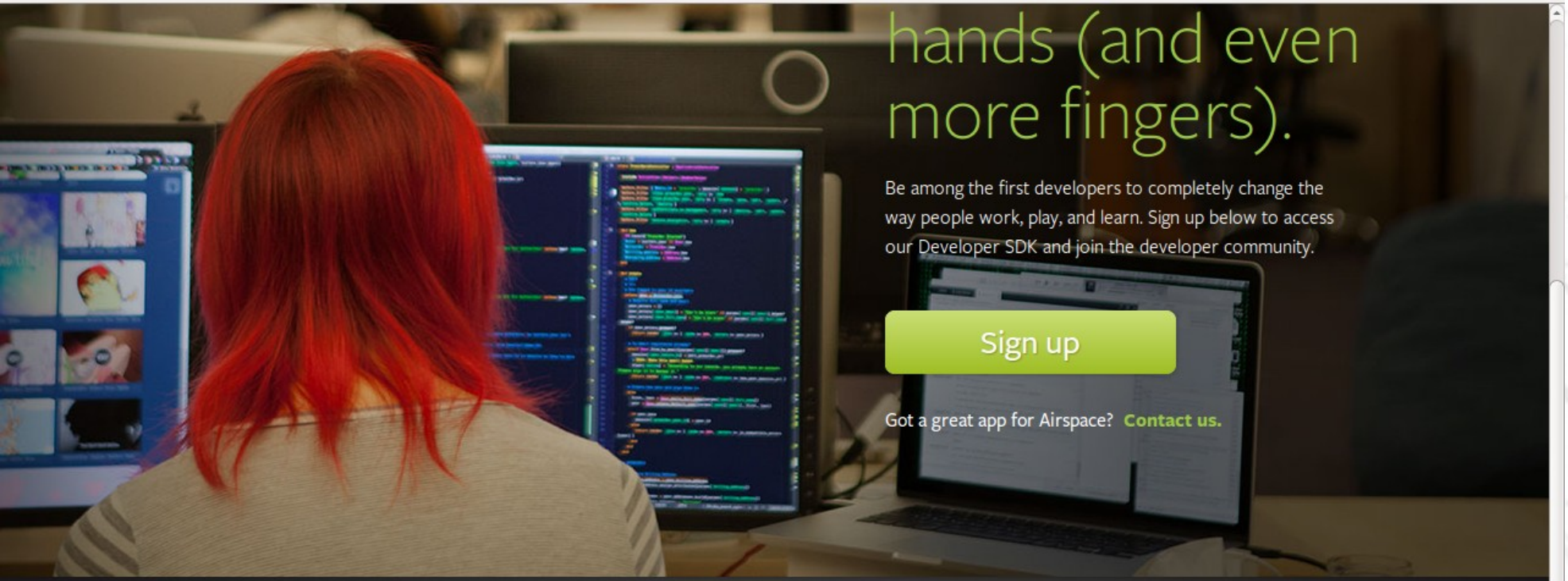
Leap Motion | Develop Your Own 3D Motion & Depth Sensor Apps - Mozilla Firefox

Файл Правка Вид Журнал Закладки Инструменты Справка

leap linux sdk - Поиск в Go... | Leap Motion | Develop Your...

https://www.leapmotion.com/developers

Google



hands (and even more fingers).

Be among the first developers to completely change the way people work, play, and learn. Sign up below to access our Developer SDK and join the developer community.

Sign up

Got a great app for Airspace? **Contact us.**

Be part of something huge.

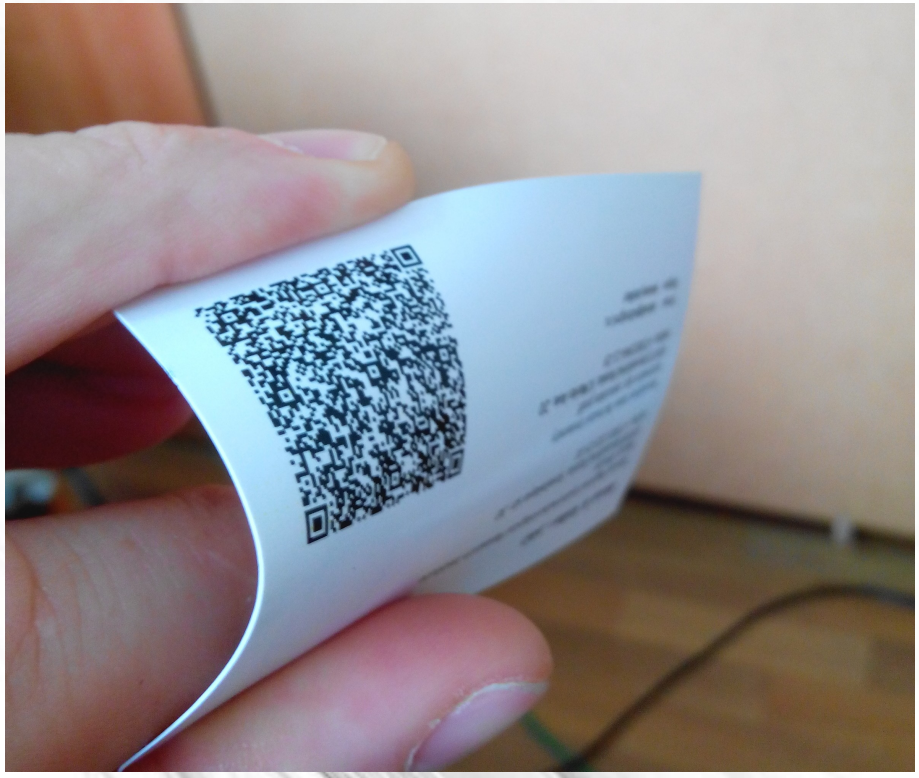
The potential of the Leap Motion Controller is huge for developers. It's a chance to do something that never could've been done before. You'll help drive the future of our device, our platform, our technology, and our company. Best of all, you'll reach a massive audience who are as excited to discover your apps as you are to create them.

When you join our developer community, you'll have everything you need to build amazing apps for Leap Motion. We'll give you a full SDK, simple submission guidelines, a complete library of resources, and full-on support from our team. You'll find endless inspiration from your fellow developers in our forums.

Already a Leap Motion Developer? **Sign in.**

Курсовая работа — вариант 1.

Поиск Bar- кода на изображении. Bar код может иметь произвольную ориентацию и искажение.



Курсовая работа вариант 2

Денежки любят счет!

Разработать приложение,способное по данным изображения подсчитать количество и номинал монет.

Что почитать (источники и ссылки)

- Intel Perceptual Computing SDK
- http://www.cognotics.com/opencv/servo_2007_se
- http://research.microsoft.com/en-us/um/people/vic
- http://habrahabr.ru/post/67937/
- http://note.sonots.com/SciSoftware/haartraining.ht
- http://opencvfacedetect.blogspot.ru/
- http://habrahabr.ru/post/103107/