Лекция

Тема: OpenCv (продолжение)

Image Processing

- http://docs.opencv.org/modules/imgproc/doc/imgp

  Содержит разделы:

  - Image Filtering (34)
  - Geometric Image Transormations (15)
  - Miscellaneous Image Transformations (8)
  - Histograms (17)
  - Structural Analysis and Shape Descriptors (18)
  - Motion Analysis and Object Tracking (6)
  - Feature Detection (10)
  - Object Detection (1)

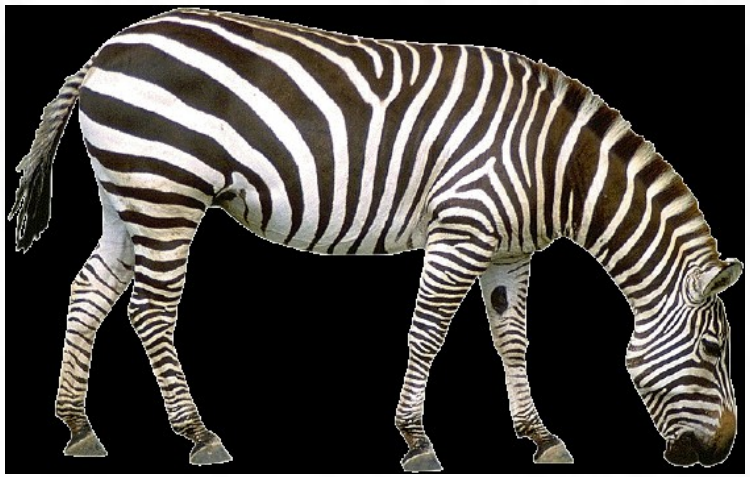
Основные задачи технического зрения

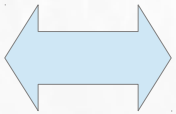
- При решении задач технического зрения приходится сталкиваться с шаблонными задачами. Шаблонные задачи, к сожалению, не всегда могут быть решены «стандартными» методами
- Рассмотрим некоторые из «шаблонных» задач и те функции (классы) из OpenCv, которые могут быть полезны.

Общая проблема задач CV

- У компьютера нет Интеллекта. Или почти нет. Или пока нет.
- Далеко не всегда есть прямая связь между значениями пикселов и объектов на сцене.

Простой случай- легкая задача(кажется)





- Соответствие между объектами визуально легко сопоставимо.
- Объект контрастен относительно фона, несмотря на сложную структуру.
- Можно относительно легко выделить контур
- Что еще?

Но простота мнима.

Об этом позже.

Но в жизни обычно

Или вот так

У некоторых свой CV на борту


© Caters News Agency

Проблема 1. Выделение контуров

- Близость цветов (яркостей) совсем не означает,что оно принадлежит к одному объекту.
- То же самое относится и к текстурам. Сложные текстуры обрабатываются наиболее сложным образом.
- Аналогично: диаметральное различие цветов не означает, что объекты разные

Эту задачу не решит ни один CV (Ирбис на охоте)

Какие проблемы вы тут видите?

Сегментация

- Сегментация- разбиение изображения на «однородные» в рамках заданных правил фрагменты.
- Цель: построить простое описание исходного изображения, которое можно было бы применить для последующего анализа.
- Выделить на изображении области, обладающие какими-либо заданными признаками

Оценка эффективности сегментации

- А) От задачи: сложный объект на изображении найден
- Б) от методики:
  - Фрагменты сегментированного изображения должны быть однородны в рамках заданного признака (набора признаков)
  - Смежные фрагменты должны существенно различаться по выбранному признаку
  - Внутренние части сегментированных фрагментов должны иметь простую форму.

Сегментация

- Цветовая сегментация
  - Цветовая (абсолютная) сегментация
  - Выращивание областей
  - Метод змейки
  - Построение границ (например,метод водораздела)
- Структурная сегментация
  - Минимизация энтропии
  - Использование априорных данных об области

Главный вопрос сегментации

- Как правильно задать признаки,по которым нужно искать однородность областей?

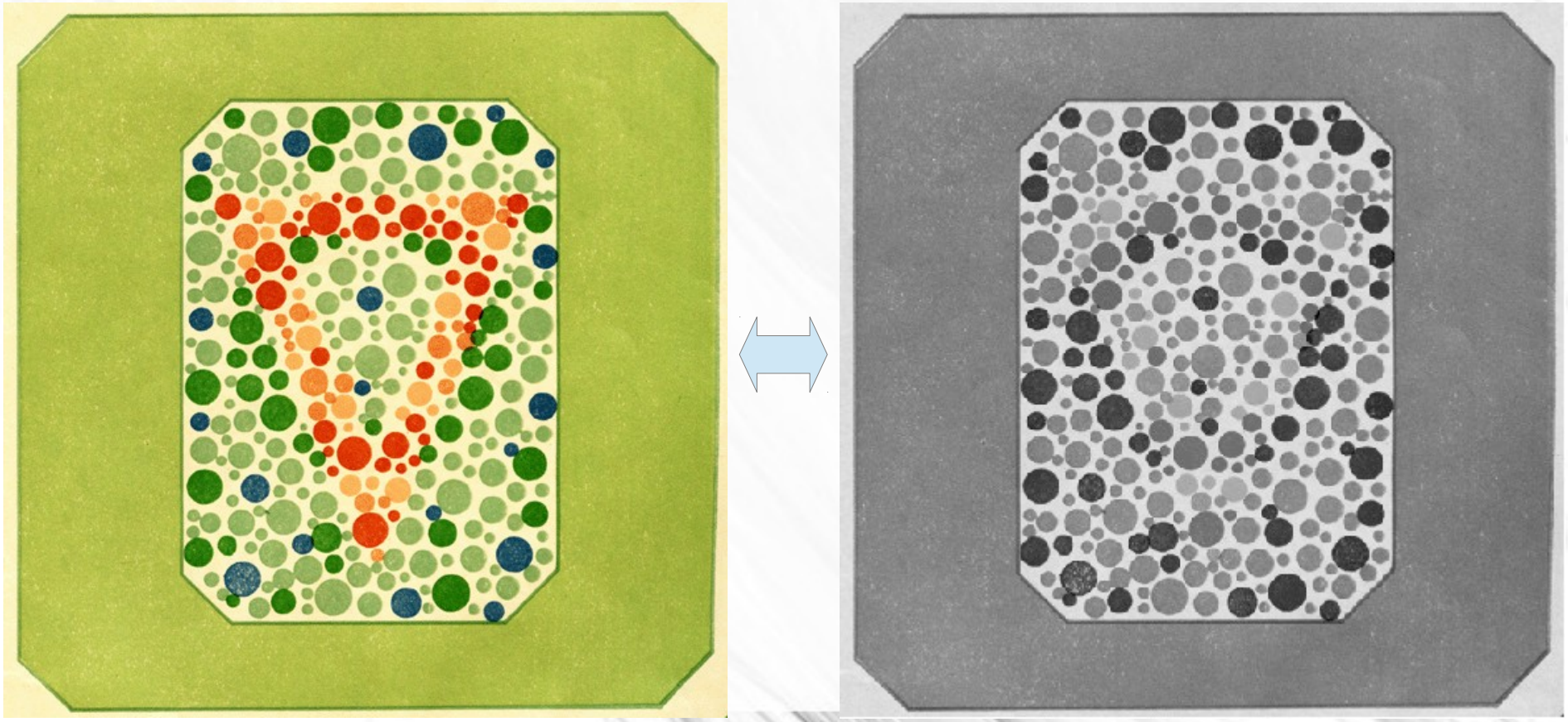


Таблица полихроматическая Е.Б. Рабкина для проверки дальтонизма (цветоощущения).

Цветовая сегментация

- Если объект лежит в области искомых цветов,то это то,что нам нужно. Если нет, то нет.
- Пример: сегментация изображений по цвету кожи человека. Как оказывается, цвета кожи человека лежат в достаточно узком диапазоне. Причем этот диапазон достаточно универсален. Удобно использовать H S V пространство (V можно не использовать)

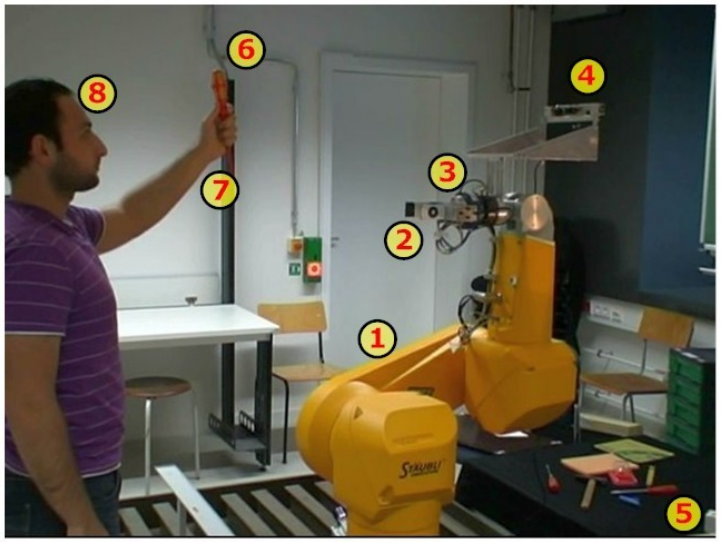
Априорная информация

- Используя априорную информацию, ширину окна для «сегментации» можно существенно сузить.
  - Ищем лицо человека
  - Считаем его цвет кожи
  - Используем в качестве доп. Априорной точки

- Стандартный диапазон[10]

  0<H<128; 59<S<175

- Модификация:



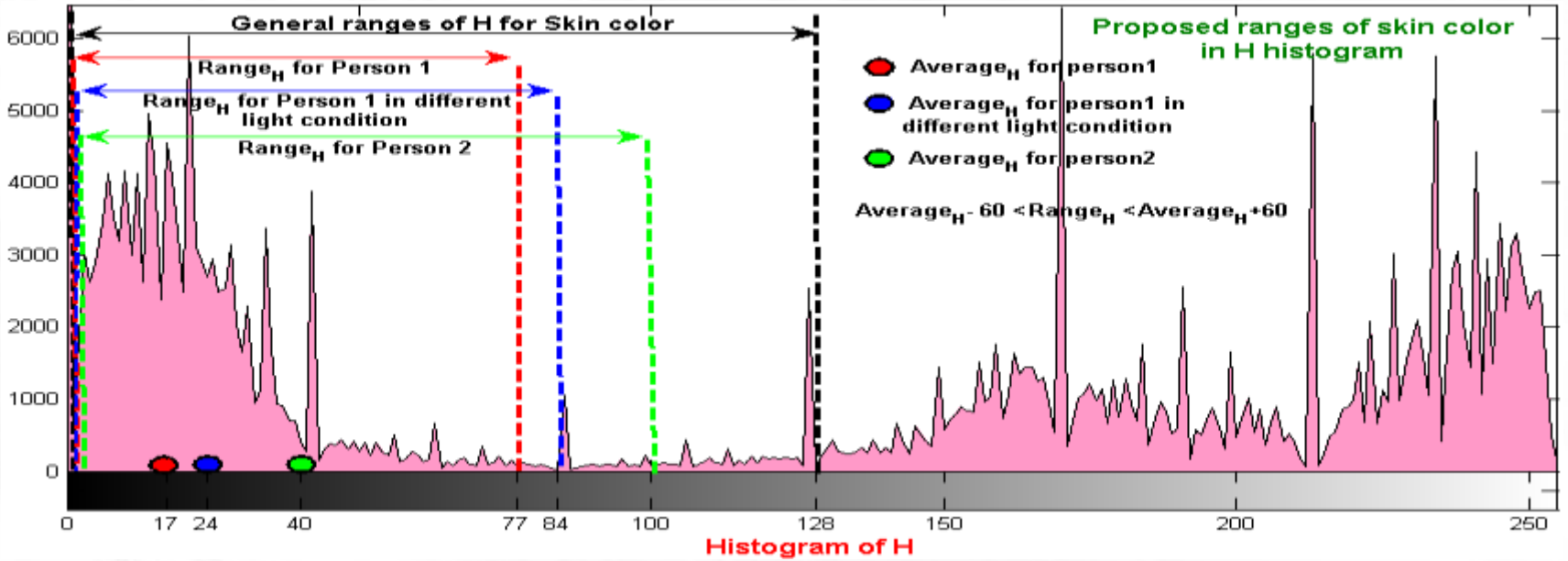
$$if\,(Face_z - 40 < Z(i,j) < Face_z + 40)\,then$$

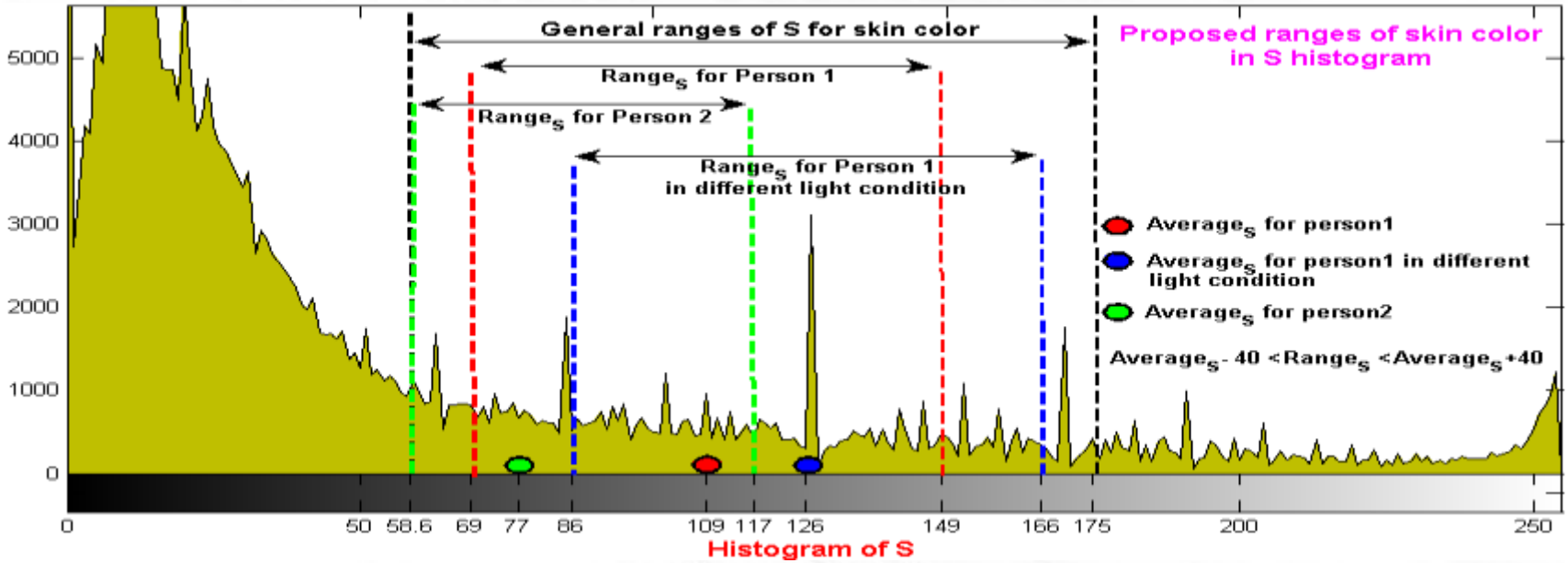
$$\{Average_H = \left(\sum_{i=FR_x}^{FR_x+FR_{Width}} \sum_{j=FR_y}^{FR_y+FR_{Len}} H(i,j) \right) / N\}$$

$$Range_H = Average_H \pm 60$$

$$Range_S = Average_S \pm 40$$

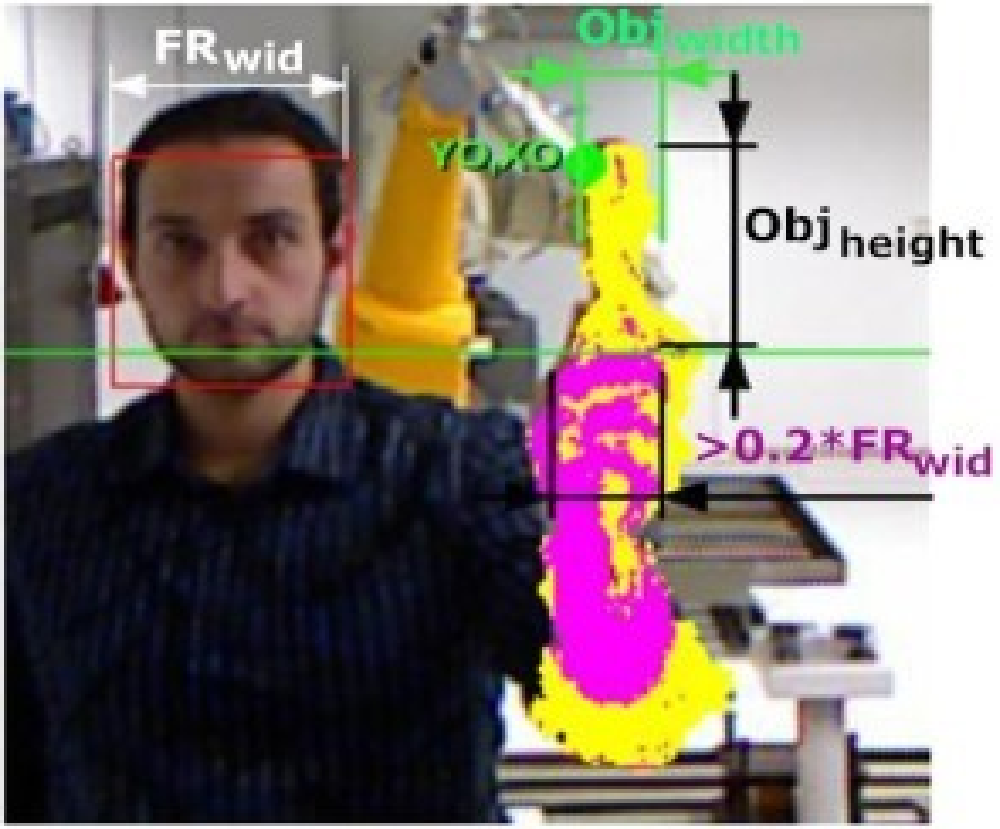
$$Range_V = Average_V \pm 60$$

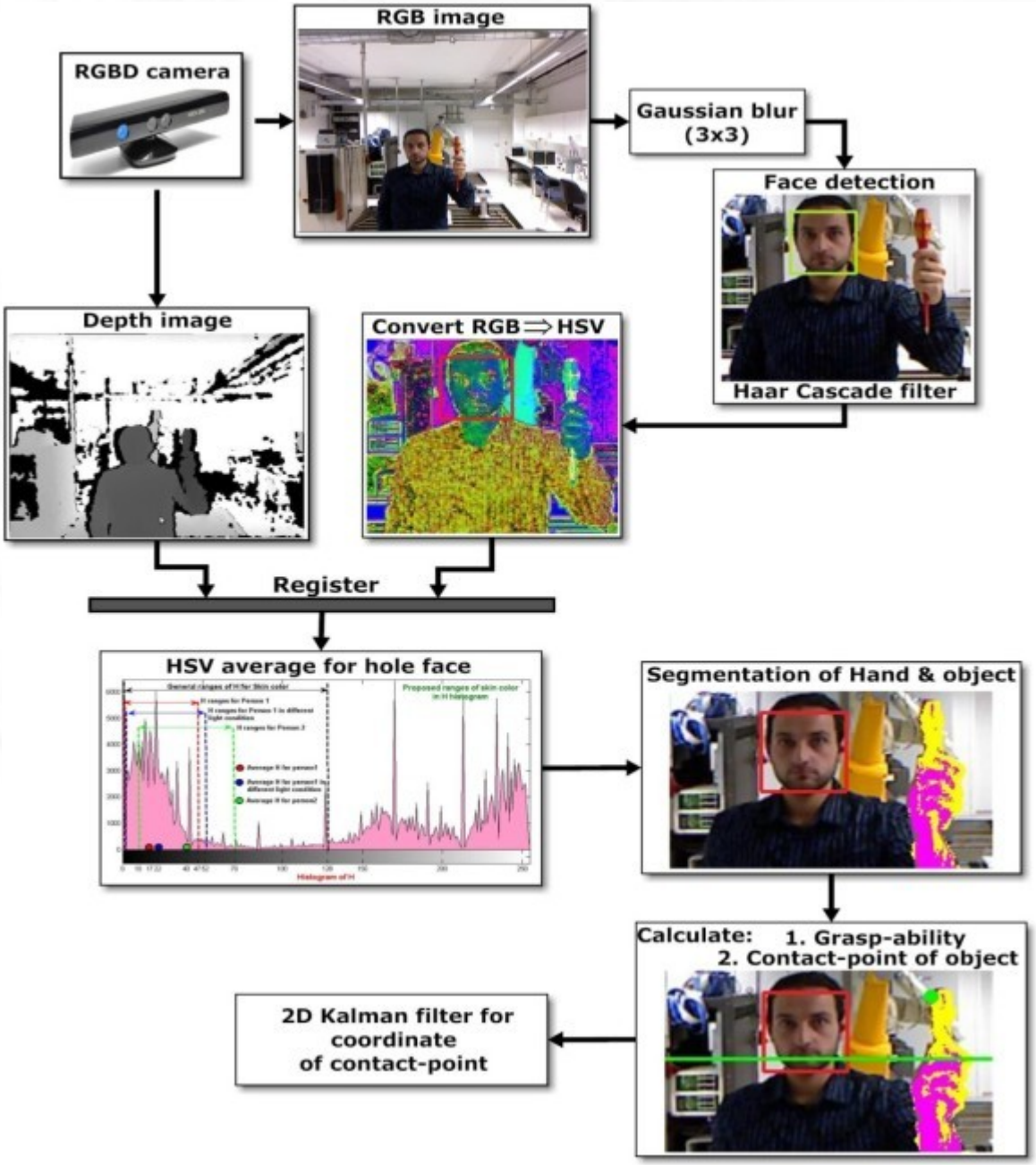

General ranges of H for Skin color
Range$_H$ for Person 1
Range$_H$ for Person 1 in different light condition
Range$_H$ for Person 2
Proposed ranges of skin color in H histogram
Average$_H$ for person1
Average$_H$ for person1 in different light condition
Average$_H$ for person2
Average$_H$ - 60 < Range$_H$ < Average$_H$ +60
6000
5000
4000
3000
2000
1000
0
0 17 24 40 77 84 100 128 150 200 250
Histogram of H


General ranges of S for skin color
Range$_S$ for Person 1
Range$_S$ for Person 2
Range$_S$ for Person 1 in different light condition
Proposed ranges of skin color in S histogram
Average$_S$ for person1
Average$_S$ for person1 in different light condition
Average$_S$ for person2
Average$_S$ - 40 < Range$_S$ < Average$_S$ +40
5000
4000
3000
2000
1000
0
0 50 58.6 69 77 86 109 117 126 149 166 175 200 250
Histogram of S

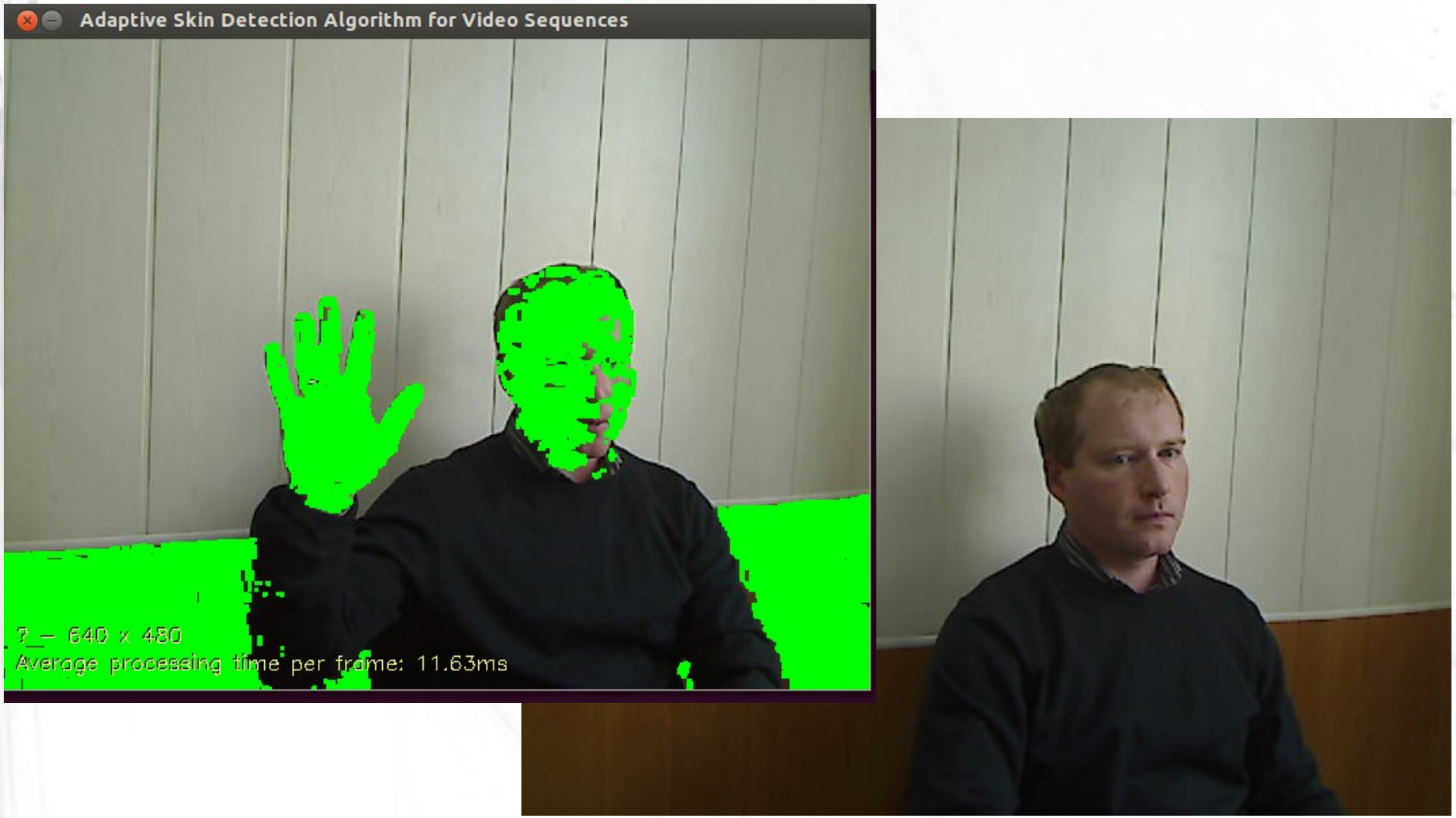
Segmentation of Model-Free Objects Carried by Human Hand:
Intended for Human-Robot Interaction Applications
Mohamad Bdiwi*
Alexey Kolker**
Jozef Suchý*

- Результат «уточненной» сегментации




RGBD camera
RGB image
Gaussian blur
(3x3)
Face detection
Haar Cascade filter
Depth image
Convert RGB⇒HSV
Register
HSV average for hole face
Segmentation of Hand & object
Calculate: 1. Grasp-ability
2. Contact-point of object
2D Kalman filter for
coordinate
of contact-point

c-example-adaptiveskindetector



Бинарная сегментация

http://docs.opencv.org/modules/imgproc/doc/miscellaneous_transformations.html?highlig

double cvThreshold(const CvArr* src, CvArr* dst, double threshold, double max_value, int threshold_type)

- **THRESH_BINARY**

$$\texttt{dst}(x,y) = \begin{cases} \texttt{maxval} & \text{if } \texttt{src}(x,y) > \texttt{thresh} \\ 0 & \text{otherwise} \end{cases}$$

- **THRESH_BINARY_INV**

$$\texttt{dst}(x,y) = \begin{cases} 0 & \text{if } \texttt{src}(x,y) > \texttt{thresh} \\ \texttt{maxval} & \text{otherwise} \end{cases}$$

- **THRESH_TRUNC**

$$\texttt{dst}(x,y) = \begin{cases} \texttt{threshold} & \text{if } \texttt{src}(x,y) > \texttt{thresh} \\ \texttt{src}(x,y) & \text{otherwise} \end{cases}$$

- **THRESH_TOZERO**

$$\texttt{dst}(x,y) = \begin{cases} \texttt{src}(x,y) & \text{if } \texttt{src}(x,y) > \texttt{thresh} \\ 0 & \text{otherwise} \end{cases}$$

- **THRESH_TOZERO_INV**

$$\texttt{dst}(x,y) = \begin{cases} 0 & \text{if } \texttt{src}(x,y) > \texttt{thresh} \\ \texttt{src}(x,y) & \text{otherwise} \end{cases}$$

Примеры

- 
- 
- Для поиска порога необходимо воспользоваться гистограммой изображения

Неравномерное освещение

$$c_2^2 = c_{2r}^2 + (u_2 - c_{2r}\operatorname{ctg}\beta_2)^2.$$

значение c_2^2 в уравнение (3.26), получим

$$(H_{ск})_{т\infty} = \frac{(u_2 - c_{2r}\operatorname{ctg}\beta_2)^2}{2g}. \qquad (3.27)$$

нию (3.25) статический напор определяется
олного и скоростного теоретических напоров

$$H_т - (H_{ск})_{т\infty} = \frac{u_2 c_{2u}}{g} - \frac{(u_2 - c_{2r}\operatorname{ctg}\beta_2)^2}{2g}.$$

ав это выражение, после подстановки

$$c_{2u} = u_2 - c_{2r}\operatorname{ctg}\beta_2$$

$$(H_{ст})_{т\infty} = \frac{u_2^2 - (c_{2r}\operatorname{ctg}\beta_2)^2}{2g}.$$

иям (3.23), (3.27) и (3.28) можно постр
имости полного напора и его составляю
На рис. 3.6 даны графики $H_{т\infty} = f(\beta$

ению с лопастями других форм. Но в общем количестве энергии, передаваемой такими лопастями, преобладает скоростная энергия. Напротив, в полной энергии, передаваемой лопастями, отогнутыми назад, преобладает энергия потенциальная (статический напор).

Способность рабочих лопастей развивать статический напор обычно характеризуют *степенью реактивности* рабочего колеса.

Степень реактивности ρ равна отношению теоретического статического напора к полному теоретическому напору, развиваемому лопастями рабочего колеса машины:

Рис. 3.6.

Адаптивный порог

- Для окрестности R пикселя I(x,y) вычисляется порог T
- Если I(x,y) > T + C, то B(x,y)=1 , иначе 0
  - T- среднее
  - T- медиана
  - T- (max-min)/2
- C- коэффициент коррекции (может быть 0)

Адаптивная процедура в Cv

http://docs.opencv.org/modules/imgproc/doc/miscellaneous_transformations.html?highlight=threshold#void%20adaptiveThreshold%28InputArray%20src,%20OutputArray%20dst,%20double%20maxValue,%20int%20adaptiveMethod,%20int%20thresholdType,%20int%20blockSize,%20double%20C%29

void **cvAdaptiveThreshold**(const CvArr* src, CvArr* dst, double max_value, int adaptive_method=CV_ADAPTIVE_THRESH_MEAN_C, int threshold_type=CV_THRESH_BINARY, int block_size=3, double param1=5)

maxValue – Non-zero value assigned to the pixels for which the condition is satisfied. See the details below.

adaptiveMethod – Adaptive thresholding algorithm to use, ADAPTIVE_THRESH_MEAN_C or ADAPTIVE_THRESH_GAUSSIAN_C . See the details below.

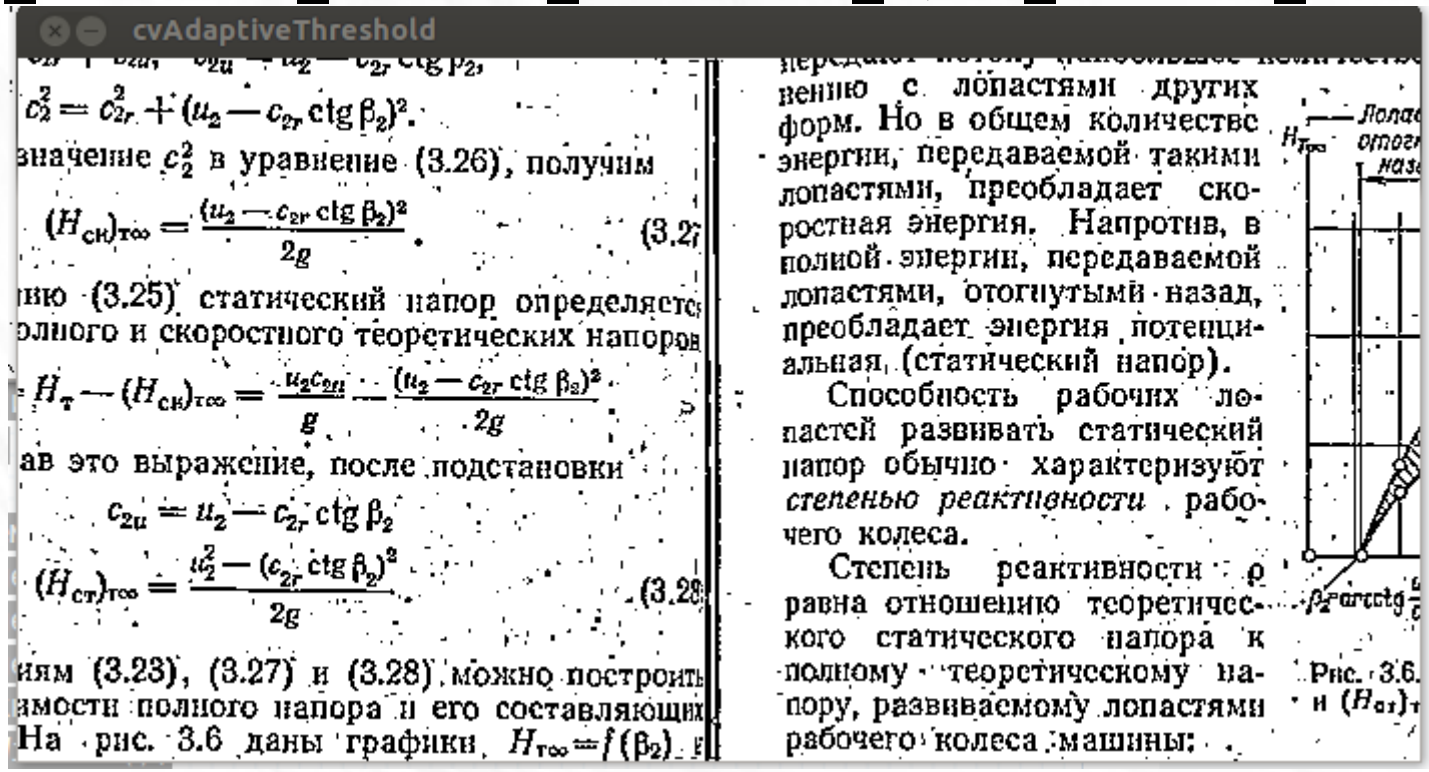
thresholdType – Thresholding type that must be either THRESH_BINARY or THRESH_BINARY_INV .

blockSize – Size of a pixel neighborhood that is used to calculate a threshold value for the pixel: 3, 5, 7, and so on.

C – Constant subtracted from the mean or weighted mean (see the details below). Normally, it is positive but may be zero or negative as well.

Результат «на лицо»

cvAdaptiveThreshold(src, dst2, 255,CV_ADAPTIVE_THRESH_GAUSSIAN_C, CV_THRESH_BINARY, 7, 5);



$$c_2^2 = c_{2r}^2 + (u_2 - c_{2r}\operatorname{ctg}\beta_2)^2.$$

значение c_2^2 в уравнение (3.26), получим

$$(H_{ск})_{т\infty} = \frac{(u_2 - c_{2r}\operatorname{ctg}\beta_2)^2}{2g}. \quad (3.2$$

вию (3.25) статический напор определяется
олного и скоростного теоретических напоров

$$H_т - (H_{ск})_{т\infty} = \frac{u_2 c_{2u}}{g} - \frac{(u_2 - c_{2r}\operatorname{ctg}\beta_2)^2}{2g}.$$

ав это выражение, после подстановки

$$c_{2u} = u_2 - c_{2r}\operatorname{ctg}\beta_2$$

$$(H_{ст})_{т\infty} = \frac{u_2^2 - (c_{2r}\operatorname{ctg}\beta_2)^2}{2g}. \quad (3.28$$

иям (3.23), (3.27) и (3.28) можно построить
амости полного напора и его составляющих
На рис. 3.6 даны графики $H_{т\infty} = f(\beta_2)$.

венною с лопастями других форм. Но в общем количестве энергии, передаваемой такими лопастями, преобладает скоростная энергия. Напротив, в полной энергии, передаваемой лопастями, отогнутыми назад, преобладает энергия потенциальная (статический напор).

Способность рабочих лопастей развивать статический напор обычно характеризуют *степенью реактивности* рабочего колеса.

Степень реактивности ρ равна отношению теоретического статического напора к полному теоретическому напору, развиваемому лопастями рабочего колеса машины:

Рис. 3.6.
и $(H_{ст})_т$

CVAPI(void) cvInRangeS(const CvArr* src, CvScalar lower,
CvScalar upper, CvArr* dst); -функция выборки пикселей из заданной области — можно также получить что-то из данной области.

Кластеризация

- Кластаризация- разделение пространства на несколько однородных зон. Граница между термином «кластеризация и сегментация» весьма тонка. Обычно «сегментация» применяется к однородным изображениям, а кластеризация- к текстурным.
- Общая проблема: критерии выбора кластеров

Кластеризация: зачем?

Кластеризация — задача машинного обучения, состоящая в разбиении заданной выборки объектов (данных) на непересекающиеся подмножества/группы (кластеры) на основе близости их признаков/значений. Т.о., каждый кластер состоит из схожих объектов

- Кластеризация позволяет:
- лучше понять данные (выявив структурные группы),
- компактное хранение данных,
- выявление новых объектов.

Метод K-means (k-средние)

- Задаем входные данные и макс число кластеров K
- Выбираем случайно K точек в векторном пространстве (случайные центры кластеров)
- Для всех точек выборки приписываем их к классу к которому они более подходят
- Перемещаем центр кластера в «центр» множества
- Останавливаемся, когда смещения всех центров масс меньше порога или истекло к-во итераций

int cvKMeans2(const CvArr* samples, int cluster_count, CvArr* labels, CvTermCriteria termcrit, int attempts=1, CvRNG* rng=0, int flags=0, CvArr* _centers=0, double* compactness=0)

- samples – Floating-point matrix of input samples, one row per sample.
- data – Data for clustering.
- cluster_count – Number of clusters to split the set by.
- K – Number of clusters to split the set by.
- labels – Input/output integer array that stores the cluster indices for every sample.
- criteria – The algorithm termination criteria, that is, the maximum number of iterations and/or the desired accuracy. The accuracy is specified as criteria.epsilon. As soon as each of the cluster centers moves by less than criteria.epsilon on some iteration, the algorithm stops.
- termcrit – The algorithm termination criteria, that is, the maximum number of iterations and/or the desired accuracy.
- attempts – Flag to specify the number of times the algorithm is executed using different initial labellings. The algorithm returns the labels that yield the best compactness (see the last function parameter).
- rng – CvRNG state initialized by RNG().

cvKMeans2

- Flag that can take the following values:
- KMEANS_RANDOM_CENTERS Select random initial centers in each attempt.
- KMEANS_PP_CENTERS Use kmeans++ center initialization by Arthur and Vassilvitskii [Arthur2007].
- KMEANS_USE_INITIAL_LABELS During the first (and possibly the only) attempt, use the user-supplied labels instead of computing them from the initial centers. For the second and further attempts, use the random or semi-random centers. Use one of KMEANS_*_CENTERS flag to specify the exact method.
- centers – Output matrix of the cluster centers, one row per each cluster center.
- _centers – Output matrix of the cluster centers, one row per each cluster center.
- compactness – The returned value that is described below.

Два примера Kmeans

Детектор границ Канни (1986)

- В данном примере мы использовали фильтр границ.
- Граница(край)- резкий переход яркости
  - Освещенность
  - Цвет
  - Глубина сцены (ориентация поверхности)
- Анализ границ- важная составляющая анализа особенностей изображения

Фильтр Канни:

- Использует первую производную от Гауссианы
- Производная плохо работает на шумах
-
- Фильтр проходит изображение в четырех направлениях:горизонтальный,вертикальный и два диагональных.
- Воспользовавшись оператором Собеля, можем получить два значения первой производной Q=arctan(Gx/Gy)
- Угол округляется до главных азимутов (0,45,...)

void cvCanny(const CvArr* image, CvArr* edges, double threshold1, double threshold2, int aperture_size=3)

Image – single-channel 8-bit input image.

edges – output edge map; it has the same size and type as image .

threshold1 – first threshold for the hysteresis procedure.

threshold2 – second threshold for the hysteresis procedure.

apertureSize – aperture size for the Sobel() operator.

L2gradient – a flag, indicating whether a more accurate L_2 norm =\sqrt{(dI/dx)^2 + (dI/dy)^2} should be used to calculate the image gradient magnitude (L2gradient=true), or whether the default L_1 norm =|dI/dx|+|dI/dy| is enough (L2gradient=false).

Оператор Собеля

- Дискретный дифференциальный оператор,вычисляющий приближение градиента яркости изображения
- Т.е. результатом работы оператора Собеля в точке области постоянной яркости будет нулевой вектор, а в точке, лежащей на границе областей различной яркости — вектор, пересекающий границу в направлении увеличения яркости.
- Основано на свертке с ядром 3x3
- -1 0 1
- -2 0 2
- -1 0 1

CvFindContours

```
Void cvFindContours( CvArr* image, CvMemStorage* storage, CvSeq** first_contour,
                     int header_size CV_DEFAULT(sizeof(CvContour)),
                     int mode CV_DEFAULT(CV_RETR_LIST),
                     int method CV_DEFAULT(CV_CHAIN_APPROX_SIMPLE),
                     CvPoint offset CV_DEFAULT(cvPoint(0,0)));
```

Важно

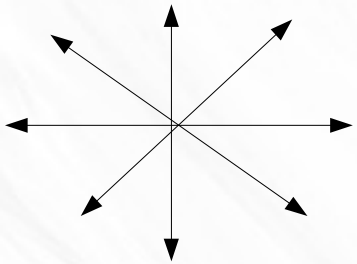
- обратите внимание, что функция cvFindContours() может находить внешние и вложенные контуры и определять их иерархию вложения.
- Функция работает с ROI
- а отобразить найденные контуры можно с помощью функции cvDrawContours():

Цепной код Фримена

- Цепные коды применяются для представления границы в виде последовательности отрезков прямых линий определённой длины и направления. В основе этого представления лежит 4- или 8- связная решётка. Длина каждого отрезка определяется разрешением решётки, а направления задаются выбранным кодом.
- (для представления всех направлений в 4-связной решётке достаточно 2-х бит, а для 8-связной решётки цепного кода требуется 3 бита)

			1	1	1			
		1				1		
	1					1		
	1					1		
		1	1	1	1			
1	1	1	2	2	2	3	4	итд

7	0	1
6		2
5	4	3



Параметры

#define CV_RETR_EXTERNAL 0 // найти только крайние внешние контуры

#define CV_RETR_LIST 1 // найти все контуры и разместить их списком

#define CV_RETR_CCOMP 2 // найти все контуры и разместить их в виде 2-уровневой иерархии

#define CV_RETR_TREE 3 // найти все контуры и разместить их в иерархии вложенных контуров

#define CV_CHAIN_CODE 0 // цепной код Фридмана

#define CV_CHAIN_APPROX_NONE 1 // все точки цепного кода переводятся в точки

#define CV_CHAIN_APPROX_SIMPLE 2 // сжимает горизонтальные, вертикальные и диагональные сегменты и оставляет только их конечные точки

#define CV_CHAIN_APPROX_TC89_L1 3 // применяется алгоритм

#define CV_CHAIN_APPROX_TC89_KCOS 4 // аппроксимации Teh-Chin

#define CV_LINK_RUNS 5 // алгоритм только для CV_RETR_LIST

Работа с контурами

Много шума- много контуров

- Обычная последовательность действий при распознавании объектов методом контурного анализа:
- 1. фильтрация
- 2. бинаризация изображения;
- 3. выделение контуров объектов;
- 4. первичная фильтрация контуров (по периметру, площади и т.п.);
- 5. эквализация контуров (приведение к единой длине, сглаживание) — позволяет добиться инвариантности к масштабу;
- 6. перебор всех найденных контуров и поиск шаблона, максимально похожего на данный контур (или же сортировка контуров по каком-либо признаку, например, площади).

Свойства

- double cvContourArea(const CvArr* contour, CvSlice slice=CV_WHOLE_SEQ, int oriented=0) - площадь контура
- double cvArcLength(const void* curve, CvSlice slice=CV_WHOLE_SEQ, int is_closed=-1) - длина контура

Что почитать (коллеги,документы,блоги)

- http://recog.ru/blog/opencv
- http://logic.pdmi.ras.ru/csclub/sites/default/files/sli
- http://docs.opencv.org/modules/imgproc/doc/imgp
- http://www.slideshare.net/kulikov_victor/3-935209
- http://www.djvu-soft.narod.ru/scan/strange_lighted.htm

К вопросу о КП