Семинар 5.2
Характеристические точки

Разработал: Максимов А.Н.

Версия 1. 01/2017

Содержание

- Параллельное выполнение и OpenCV
- Характеристические точки
- Дополненная реальность и электронные микроскопы :)

Методы выделения характеристических точек.

Предложены различные методы выделения и сопоставления характеристичесих точек:

Наиболее известные методы выделения:

- SIFT
- SURF
- ORB
- Klt

См:

http://docs.opencv.org/3.0-beta/doc/py_tutorials/py_feature2d/py_matcher/py_matcher.html

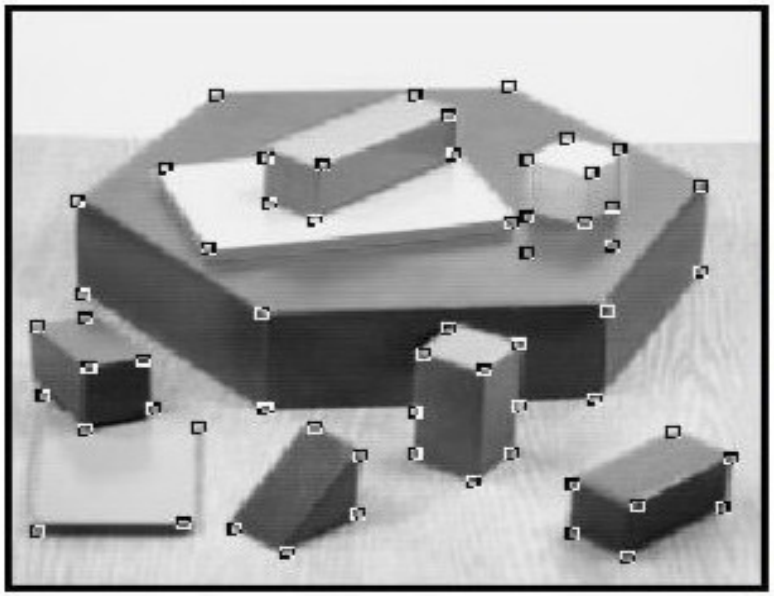
Детекторы углов

Детекторы углов

- Moravec
- Harris
- Shi-Tomasi
- Förstner
- SUSAN
- Trajkovic
- FAST
- CSS
- Детектор, основанный на глобальных и локальных свойствах кривизны
- CP

https://habrahabr.ru/post/244541/

Пример работы



https://habrahabr.ru/post/244541/

SIFT

SIFT (Scale-invariant feature transform) is an algorithm in computer vision to detect and describe local features in images. The algorithm was patented in the US by the University of British Columbia[1] and published by David Lowe in 1999

SIFT

```
#include "opencv2/xfeatures2d.hpp"
  // Создать SIFT детектор
  cv::Ptr<Feature2D> f2d = xfeatures2d::SIFT::create();
  //cv::Ptr<Feature2D> f2d = xfeatures2d::SURF::create();
  //cv::Ptr<Feature2D> f2d = ORB::create();
  // Сравниваем кадр из потока с образцом
  //-- Step 1: Detect the keypoints:
  std::vector<KeyPoint> keypoints_1, keypoints_2;
  f2d->detect( frame, keypoints_1 );              // Найдем характеристические точки для фрейма
  f2d->detect( img_templete, keypoints_2 );   // Найдем характеристические точки для фрейма (можно сделать
      заранее)
  //-- Step 2: Вычислить описатели  (feature vectors)

  Mat descriptors_1, descriptors_2;

   f2d->compute( img_1, keypoints_1, descriptors_1 );
  f2d->compute( img_2, keypoints_2, descriptors_2 );
  //-- Step 3: Сопоставить вектора описателей при помощи BFMatcher :
  BFMatcher matcher;
  std::vector< DMatch > matches;
  matcher.match( descriptors_1, descriptors_2, matches );
```

SURF

SURF (Speeded Up Robust Features) – алгоритм выделения на изображении характеристических точек и расчета дескрипторов для этих точек

Описание алгоритма может быть найдено тут:

ftp://ftp.vision.ee.ethz.ch/publications/articles/eth_biwi_00517.pdf

SURF

```
vector<KeyPoint> keypoints_object, keypoints_scene; // keypoints
Mat descriptors_object, descriptors_scene; // descriptors (features)
//-- Steps 1 + 2, detect the keypoints and compute descriptors, both in one method
Ptr<SURF> surf = SURF::create( minHessian );
surf->detectAndCompute( img_object, noArray(), keypoints_object, descriptors_object );
surf->detectAndCompute( img_scene, noArray(), keypoints_scene, descriptors_scene );
//-- Step 3: Matching descriptor vectors using FLANN matcher
FlannBasedMatcher matcher; // FLANN - Fast Library for Approximate Nearest Neighbors
vector< vector< DMatch> > matches;
matcher.knnMatch( descriptors_object, descriptors_scene, matches, 2 ); // find the best 2 matches of each
      descriptor
timer.Stop();printf( "Method processImage() ran in: %f msecs.\n", timer.Elapsed() );
//-- Step 4: Select only goot matches
std::vector< DMatch > good_matches;
for (int k = 0; k < std::min(descriptors_scene.rows - 1, (int)matches.size()); k++) {
    if ( (matches[k][0].distance < 0.6*(matches[k][1].distance)) &&((int)matches[k].size() <= 2 &&
      (int)matches[k].size()>0) ) {
        // take the first result only if its distance is smaller than 0.6*second_best_dist
        // that means this descriptor is ignored if the second distance is bigger or of similar
        good_matches.push_back( matches[k][0] );
    }
```

RISoft

Выделение лица в OpenCv

Пример кода выделения лиц:

https://github.com/slucey-cs-cmu-edu/Detect_Lena.git

Опасности OpenCv :)

1. Можно сделать много понимая мало
2. Нужно знать C++ (на самом деле нет)
3. Как перейти от прототипа к продуктовому коду?

Прочесть.

1. Саттер Г. Решение сложных задач на C++. - Глава 5.

Задания

Task6_video. Реализовать программу “кротовая нора”

- читает видео файл “пистолет и плакат” (приложен ко вчерашней презентации);
- выделить линии принцельной планки пистолета при помощи LSD
- выделить мушку ступенчатым преобразованием (оранжевая)
- уточнить положение линии прицеливания путем отбрасывания кандидатов в линии прицеливания, которые не пересекают мушку
- определить направление (азимут) и угол наклона ствола
- добавить возможность временно (на 1-2 секунды) рисовать темную окружность пулевого отверстия по нажатию на клавишу (Enter)

Порядок выполнения

1. Необходимо создать репозиторий на github и прислать ссылку на репозиторий на rt.practic@dev.rtsoft.ru
2. Для каждой задачи необходимо создавать свой каталог, например, task1
3. Код задачи должен собираться. Желательно, чтобы добавить Makefile

Литература по Win32

1. http://www.bogotobogo.com/cplusplus/multithreading_win32A.php
2. http://icourse.cuc.edu.cn/networkprogramming/lectures/Unit3_MultithreadingMFC.pdf

Литература по C++

1. http://thispointer.com/c-11-multithreading-part-1-three-different-ways-to-create-threads/

2. https://habrahabr.ru/post/182610/

3. Потоки, блокировки и условные переменные в C++11 [Часть 2] https://habrahabr.ru/post/182626/

Литература по OpenCV

1. http://16423.courses.cs.cmu.edu/slides/Lecture_2.pdf
(тут есть ссылка на выделение лиц !!!)
2. Что нового есть в OpenCV 3.0
http://visilab.etsii.uclm.es/personas/oscar/Publications/New_functionality_OpenCV3.pdf
3. http://www.coldvision.io/2016/06/27/object-detection-surf-knn-flann-opencv-3-x-cuda/
(статья с хорошими примерами)
4. http://study.marearts.com/2015/06/opencv-30-rc1-example-source-code-for.html

Дополненная реальность и электронные микроскопы

1. patentimages.storage.googleapis.com/pdfs/US20130221218.pdf