Семинар 1.3 Дополненная реальность - работа с видео

Разработал: Максимов А.Н.

Версия 2. 06/2017

Содержание

- Структуры данных
  - динамический массив
  - список
- Пространства имен, исключения
- OpenCv
- cv::Mat
- Выделение контрастных объектов при помощи ступенчатого преобразования
- Задача “кротовья нора”
- Пример полупрозраные фигуры

Идея этого курса

Идея этого курса появилась, когда мой коллега прислал ссылку на ролик о дополненной реальности

https://www.youtube.com/watch?v=kPMHcanq0xM

Идея этого курса

Добавим видео вместо календаря, нарисуем выстрел и ….

Как это реализовать?

Очень упрощенно:

- Для на начала определим пространство корридора
- Выделим линии границ пола и пототолка (они сходятся на линии горризонта)
- Для выделения воспользуемся детектором линий LSD или преобразованием хаффа
- Нарисуем их на изображении

Захват видео с камеры

```
#include "opencv2/opencv.hpp"
using namespace cv;
int main(int, char**)  {
    VideoCapture cap(0); // Открываем камеру или файл
    if(!cap.isOpened())     // check if we succeeded
        return -1;
    Mat edges;
    namedWindow("edges",1);
    while(1) {
        Mat frame;
        cap >> frame; // get a new frame from camera
        cvtColor(frame, edges, COLOR_BGR2GRAY);     // Перевод в градации серого
        GaussianBlur(edges, edges, Size(7,7), 1.5, 1.5);  // Размытие
        Canny(edges, edges, 0, 30, 3);                    // Выделение границ
        imshow("edges", edges);
        if(waitKey(30) >= 0) break;
    }
    return 0;
}
```

Пример из http://docs.opencv.org/3.0-beta/modules/videoio/doc/reading_and_writing_video.html

Чтение видео из файла

```
#include "opencv2/highgui/highgui.hpp"
using namespace cv;
int main(int argc, char* argv[]) {
    VideoCapture cap("IMG_2096.MOV"); // open the video file for reading
    if ( !cap.isOpened() )          return -1;
    //cap.set(CV_CAP_PROP_POS_MSEC, 300);      //start the video at 300ms
    double fps = cap.get(CV_CAP_PROP_FPS);     //get the frames per seconds of the video
     std::cout << "Frame per seconds : " << fps << std::endl;
    namedWindow("MyVideo",CV_WINDOW_AUTOSIZE); //create a window called "MyVideo"
    while(1) {
        Mat frame;
        bool bSuccess = cap.read(frame); // read a new frame from video
        if (!bSuccess) {
           std::cout << "Cannot read the frame from video file" << std::endl;
           break;
        }
        imshow("MyVideo", frame); //show the frame in "MyVideo" window
        if(waitKey(30) == 27) {
               break;
        }
    }
}
```

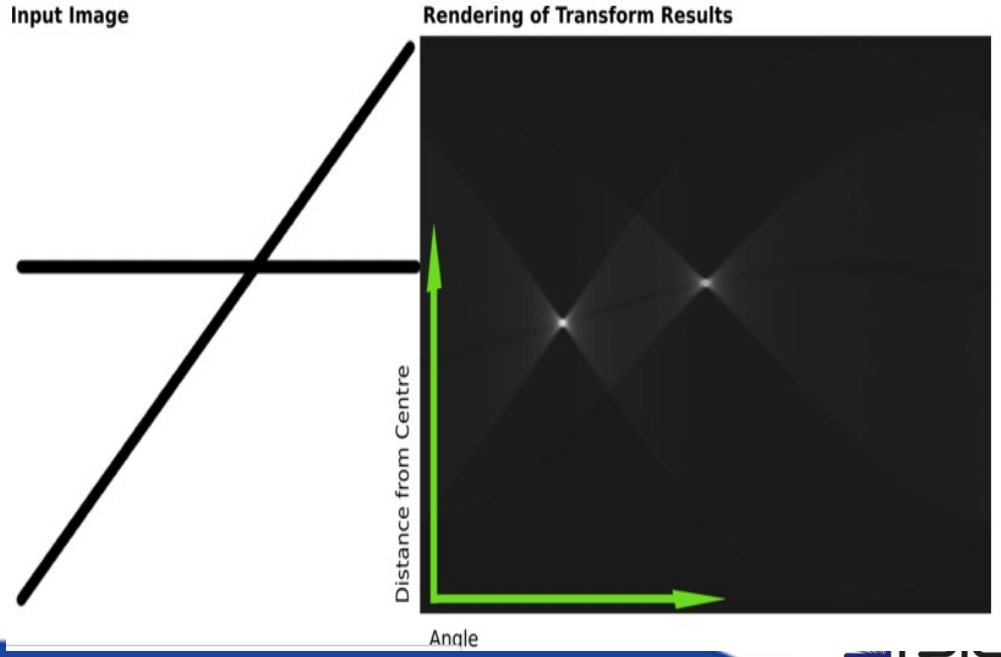
Пример из http://opencv-srf.blogspot.ru/2011/09/capturing-images-videos.html

Алгоритмы выделения линий

Алгоритмы выделения линий из изображения:

1. **Преобразование Хаффа** - преобразование Хафа является линейным преобразованием для обнаружения прямых (есть для элипсов и окружностей). Прямая может быть задана уравнением y = mx + b и может быть вычислена по любой паре точек (x, y) на изображении. Главная идея преобразования Хафа — учесть характеристики прямой не как уравнение, построенное по паре точек изображения, а в терминах её параметров, то есть m — коэффициента наклона и b — точки пересечения с осью ординат. Исходя из этого прямая, заданная уравнением y = mx + b, может быть представлена в виде точки с координатами (b, m) в пространстве параметров. Для

2. **LSD (line segment detector)**



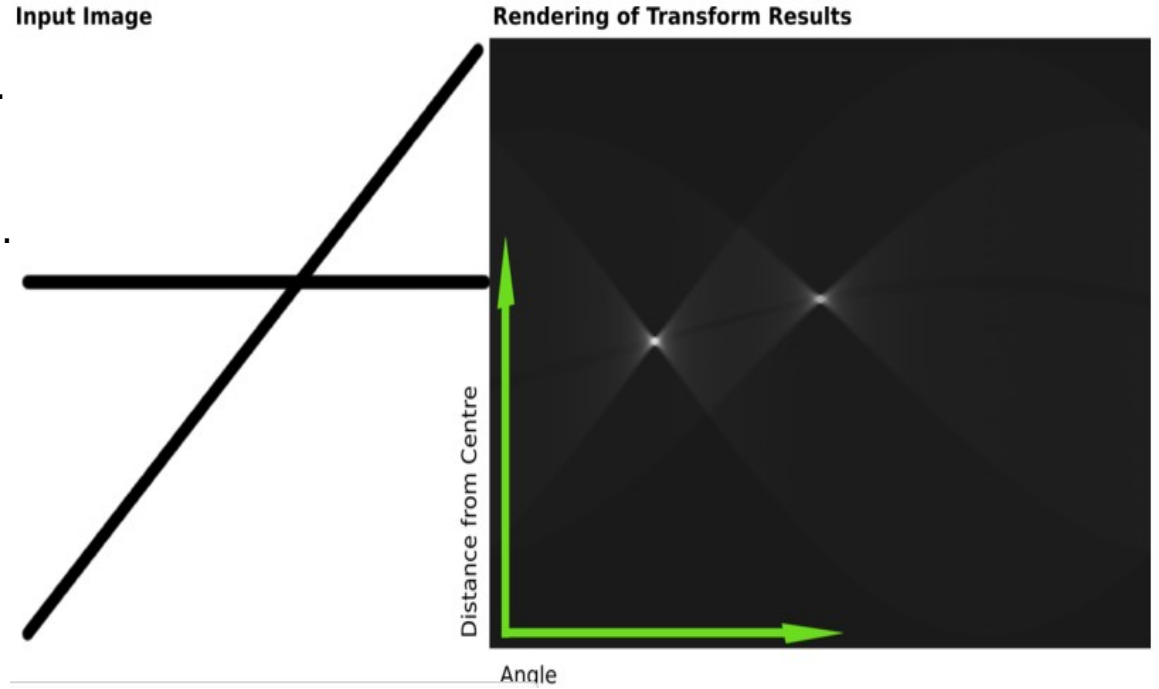
Алгоритмы выделения линий

Алгоритмы выделения линий из изображения:

1. **Преобразование Хаффа** - преобразование Хафа является линейным преобразованием для обнаружения прямых (есть для элипсов и окружностей). Прямая может быть задана уравнением y = mx + b и может быть вычислена по любой паре точек (x, y) на изображении. Главная идея преобразования Хафа — учесть характеристики прямой не как уравнение, построенное по паре точек изображения, а в терминах её параметров, то есть m — коэффициента наклона и b — точки пересечения с осью ординат. Исходя из этого прямая, заданная уравнением y = mx + b, может быть представлена в виде точки с координатами (b, m) в пространстве параметров. Для
2. **LSD (line segment detector)**

Преобразование Хаффа

Преобразование Хаффа(патент 1962 г. Поля Хафа) - Прямая, заданная уравнением y = mx + b, может быть представлена в виде точки с координатами (b, m) в пространстве параметров. Удобней представить прямую с помощью других параметров, известных как r и θ (theta). Параметр r — это длина радиус-вектора ближайшей к началу координат точки на прямой (т.е. нормали к прямой, проведенной из начала координат), а θ (heta) — это угол между этим вектором и осью абсцисс. При таком описании прямых не возникают бесконечные параметры.



Вычислительная сложность O(N3).

Существуют разные модификации.

Наиболее быстрая:

Progressive Probabilistic Hough Transform

LSD

LSD (Line Segment Detector) это детектор отрезков прямых с линейным временем поиска. Он быстрее и лучше ищет, чем преобразование Хаффа

Предложен:

Rafael Grompone von Gioi, Jérémie Jakubowicz, Jean-Michel Morel, and Gregory Randall, LSD: a Line Segment Detector, Image Processing On Line, vol. 2012. http://dx.doi.org/10.5201/ipol.2012.gjmr-lsd

Почитать можно тут:

http://www.ipol.im/pub/art/2012/gjmr-lsd/article.pdf

Добавлен в OpenCV 3.0 в модуль imgproc

Пример использования:

[opencv_source_code]/samples/cpp/lsd_lines.cpp

Выделение границ

Фильтр Собеля — дискретный дифференциальный оператор, вычисляющий приближённое значение градиента яркости изображения. Результатом применения оператора Собеля в каждой точке изображения является либо вектор градиента яркости в этой точке, либо его норма.

https://en.wikipedia.org/wiki/Sobel_operator

void cvSobel(const CvArr* src, CvArr* dst, int xorder, int yorder, int aperture_size=3)

Фильтр Канни Оператор обнаружения границ изображения. Использует многоступенчатый алгоритм для обнаружения широкого спектра границ в изображениях.

http://docs.opencv.org/3.0-beta/doc/py_tutorials/py_imgproc/py_canny/py_canny.html

void Canny(InputArray image, OutputArray edges, double threshold1, double threshold2, int apertureSize=3, bool L2gradient=false)

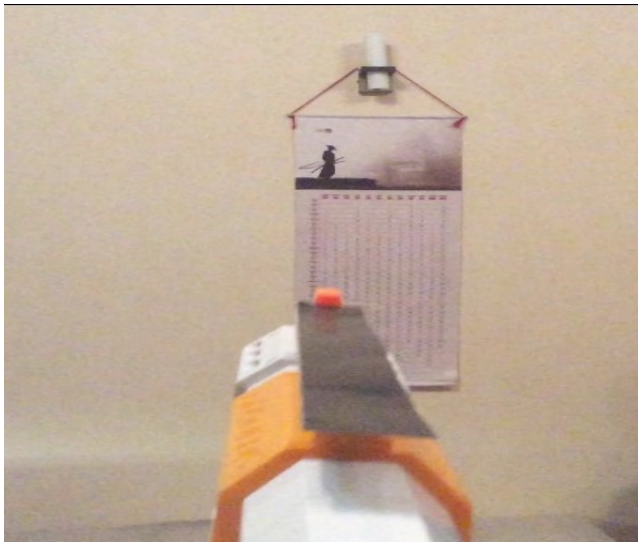
Применение фильтра Канни

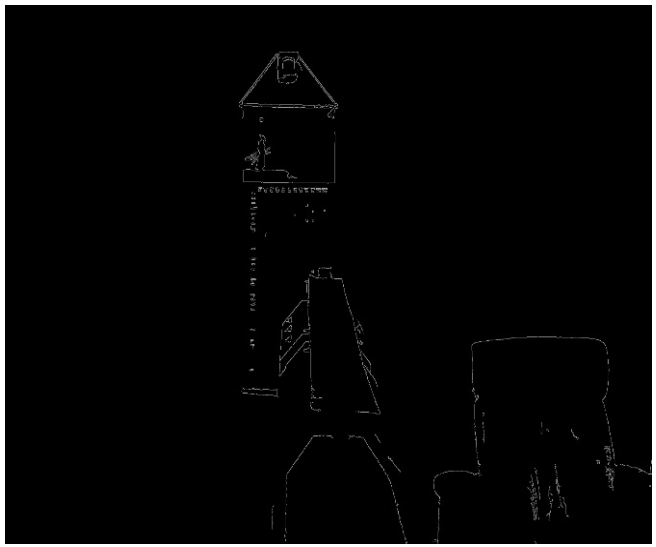
```
Mat imgContours;
double thresh = xxx;
// trydifferent values to see effect
Canny(imgInput,imgContours,0.4*thresh,thresh);
```

Результат фильтра Canny

```
Mat gray_image;
Mat edge, draw;

cvtColor( frame, gray_image, COLOR_BGR2GRAY );
Canny( gray_image, edge, 50, 150, 3);
resize(edge,edge,cv::Size(800,600));
imshow("Canny", edge);
```



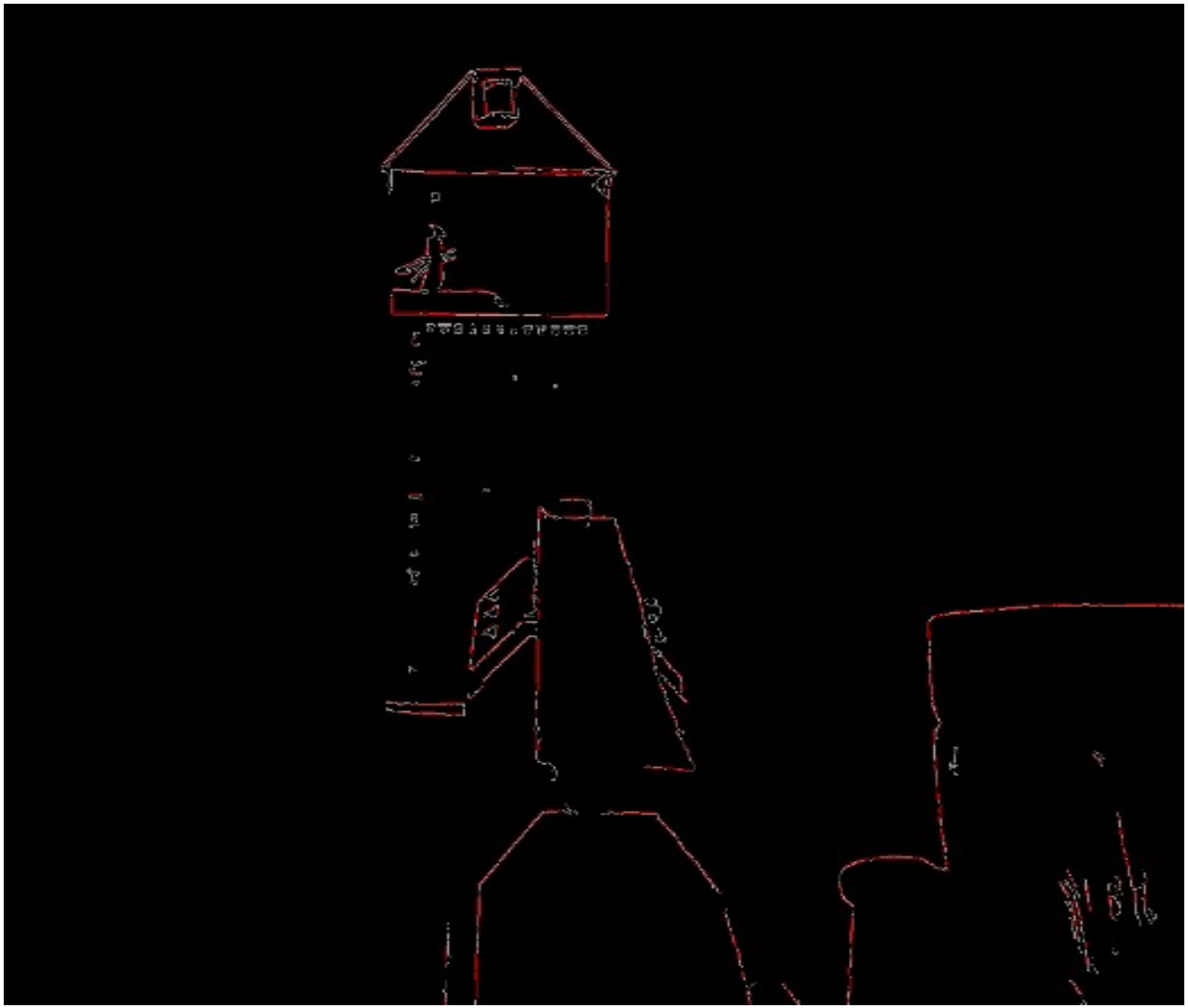


Пример выделения линий используя LSD

```
Ptr<LineSegmentDetector> ls = createLineSegmentDetector(LSD_REFINE_STD);
while(1) {
    Mat frame;
    std::vector<Vec4f> lines_std;
    bool bSuccess = cap.read(frame); // read a new frame from video
    if (!bSuccess) {
       std::cout << "Cannot read the frame from video file" << std::endl;
       break;
    }
    Mat gray_image;
    cvtColor( frame, gray_image, COLOR_BGR2GRAY );
    ls->detect(gray_image, lines_std); // Detect the lines
    ls->drawSegments(frame, lines_std);
    cv::resize(frame,frame,cv::Size(800,600));
    imshow("MyVideo", frame); //show the frame in "MyVideo" window
    if(waitKey(30) == 27) {
            break;
     }
   }
}
```

Результа выделения линий используя LSD

Линии выделенные LSD отображены красным



RISoft

Результат выделения прицельной планки при помощи ступенчатого преобразования

Ступенчатое преобразование, canny, LSD (красный)

Прочесть.

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

Задания

Task6_video. Реализовать программу “кротовая нора”

- читает видео файл “корридор” (приложен к сегодняшней);
- выделить линии принцельной пола и потолка на изображении
- отрисовать линии пола и потолка поверх изображение зеленым цветом

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 C++

1. Бьерн Страуструп. Язык программирования C++.

2. Тут есть несколько неплохих туториалов. http://www.cprogramming.com/tutorial.html

3. Introduction to Programming Concepts in C++ (хорошие базовые лекции по C++ на английском)

http://staffwww.fullcoll.edu/brippe/csci123/lectures.aspx

4. http://rsc-team.ru/index.pl?rzd=2&group=lection&ind=22

5. Г. Саттер Решение сложных задач на C++.

6. Связанный список http://pumpkinprogrammer.com/2014/06/13/c-tutorial-intro-to-linked-lists/

Литература по OpenCV

1. http://cvgl.stanford.edu/BAVM14/slides/Bradski.pdf - хороший обзор по OpenCV
2. Усатый мужик http://sublimerobots.com/2015/02/dancing-mustaches/
3. OpenCV video editing tutorial https://www.solarianprogrammer.com/2015/06/04/ opencv-video-editing-tutorial/
4. Чтение видео http://docs.opencv.org/3.0-beta/modules/videoio/doc/reading_and_writing_video.html
5. Чтение из файла http://opencv-srf.blogspot.ru/2011/09/capturing-images-videos.html
6. Выделение линий https://marcosnietoblog.wordpress.com/2012/04/28/line-segment-detection-opencv-c-source-code/
7. http://docs.opencv.org/3.0-beta/modules/line_descriptor/doc/tutorial.html
8. Пример использования LSD http://docs.opencv.org/trunk/d8/dd4/lsd_lines_8cpp-example.html#a15

Литература по OpenCV

9. Преобразование Хаффа http://inside.mines.edu/~whoff/courses/EENG512/lectures/HoughInOpenCV.pdf
10. Zero-parameter, automatic Canny edge detection with Python and OpenCV http://www.pyimagesearch.com/2015/04/06/zero-parameter-automatic-canny-edge-detection-with-python-and-opencv/ (Автоматом подбирает настройки Канни !!!)
11. Поиск границ на цветных изображениях http://www.algorithmist.ru/2011/03/colorful-image-edge-detection.html
12. http://www.pyimagesearch.com/2016/04/04/measuring-distance-between-objects-in-an-image-with-opencv/
13. Learning OpenCV 3: Computer Vision in C++ with the OpenCV Library (Книга !!!)