Learn OpenCV by Examples

OpenCV simplified for beginners by the use of examples. Learn OpenCV with basic implementation of different algorithms.

Home | For Beginners | Table of Contents | Keywords

Putting a mask on face using OpenCV

This is a example to put a mask on your face. This is a fun application that is meant to be used for entertainment purpose. You might have seen such apps in google hangouts.

Steps:

1. Detect face from the input video frame
2. Load the mask and make the white region of the mask transparent.
3. Put the mask at the face position
4. Display the image

Any mask can be used for this purpose. I have downloaded some masks from internet which you can download here.

Download masks

Put the mask images in a folder with folder name "masks" inside the project folder.

For achieving higher speed, the parameter are adjusted according to this.

Code:

SEARCH CONTENTS OF THIS BLOG

Search

POPULAR POSTS

1 Find Contour

2 Basic drawing examples

3 Line Detection by Hough Line Transform

Face Detection using Haar-Cascade Classifier

Perspective Transform

Sobel Edge Detection

```cpp
#include "opencv2/objdetect/objdetect.hpp"
#include "opencv2/highgui/highgui.hpp"
#include "opencv2/imgproc/imgproc.hpp"

#include <iostream>
#include <stdio.h>

using namespace std;
using namespace cv;

double min_face_size=20;
double max_face_size=200;
Mat mask;

Mat detectFace(Mat src);
Mat putMask(Mat src,Point center,Size face_size);

int main( )
{
    VideoCapture cap(0);
    namedWindow( "window1", 1 );
    mask = imread("masks/5.jpg");

    while(1)
    {
        Mat frame;
        cap >> frame;
        frame=detectFace(frame);

        imshow( "window1", frame );
        // Press 'c' to escape
        if(waitKey(1) == 'c') break;
    }

    waitKey(0);
    return 0;
}

Mat detectFace(Mat image)
{
    // Load Face cascade (.xml file)
    CascadeClassifier face_cascade( "C:/OpenCV243/data/Haarcascades

    // Detect faces
    std::vector<Rect> faces;

    face_cascade.detectMultiScale( image, faces, 1.2, 2, 0|CV_HAAR_

    // Draw circles on the detected faces
    for( int i = 0; i < faces.size(); i++ )
    {   // Lets only track the first face, i.e. face[0]
        min_face_size = faces[0].width*0.7;
        max_face_size = faces[0].width*1.5;
```

Kalman Filter Implementation (Tracking mouse position)

Histogram Calculation

9 OpenCV example to convert RGB to gray / other color spaces

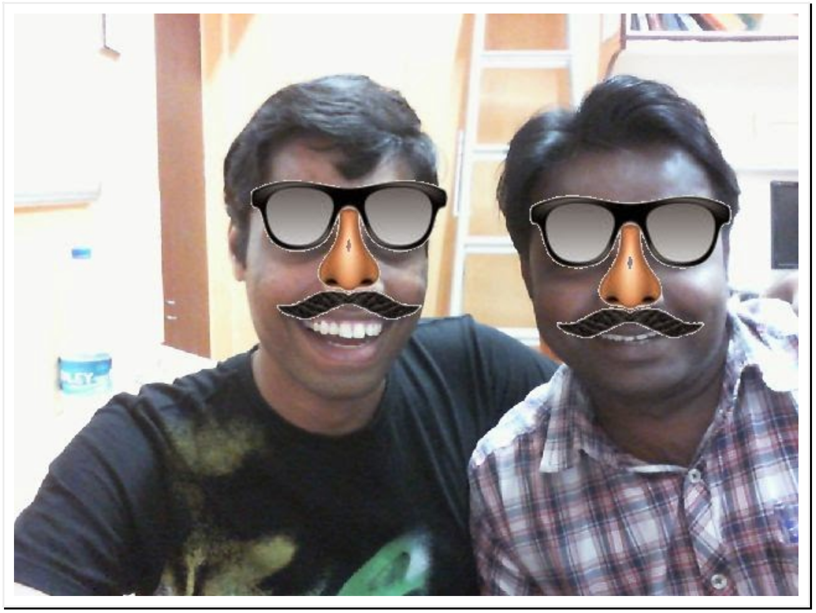
10 Hough Circle Detection

CATEGORIES

- Accessory
- Applications
- Basics
- Edge Detection
- Feature Extraction
- Filter
- Miscellaneous
- Morphological Operation

```
        Point center( faces[i].x + faces[i].width*0.5, faces[i].y +
        image=putMask(image,center,Size( faces[i].width, faces[i].h
    }
    return image;
}

Mat putMask(Mat src,Point center,Size face_size)
{
    Mat mask1,src1;
    resize(mask,mask1,face_size);

    // ROI selection
     Rect roi(center.x - face_size.width/2, center.y - face_size.wi
    src(roi).copyTo(src1);

    // to make the white region transparent
    Mat mask2,m,m1;
    cvtColor(mask1,mask2,CV_BGR2GRAY);
    threshold(mask2,mask2,230,255,CV_THRESH_BINARY_INV);

    vector<Mat> maskChannels(3),result_mask(3);
    split(mask1, maskChannels);
    bitwise_and(maskChannels[0],mask2,result_mask[0]);
    bitwise_and(maskChannels[1],mask2,result_mask[1]);
    bitwise_and(maskChannels[2],mask2,result_mask[2]);
    merge(result_mask,m );         //    imshow("m",m);

    mask2 = 255 - mask2;
    vector<Mat> srcChannels(3);
    split(src1, srcChannels);
    bitwise_and(srcChannels[0],mask2,result_mask[0]);
    bitwise_and(srcChannels[1],mask2,result_mask[1]);
    bitwise_and(srcChannels[2],mask2,result_mask[2]);
    merge(result_mask,m1 );        //    imshow("m1",m1);

    addWeighted(m,1,m1,1,0,m1);    //    imshow("m2",m1);

    m1.copyTo(src(roi));

    return src;
}
```

Result:



Labels: Applications

1 comment:

Johnson Mark March 28, 2015 at 9:13 AM

Thanks for good job. It is better if we consider the angle of face. And then the mask will roate with correspoding angle

Reply