# [**Drawing rectangle or line using mouse events in open cv using python**](http://stackoverflow.com/questions/22140880/drawing-rectangle-or-line-using-mouse-events-in-open-cv-using-python)

URL: <http://stackoverflow.com/questions/22140880/drawing-rectangle-or-line-using-mouse-events-in-open-cv-using-python>

You can refer below C++ code which I used to crop image

#include <iostream>

#include "opencv2/opencv.hpp"

#include <stdio.h>

using namespace std;

using namespace cv;

Mat src,img,ROI;

Rect cropRect(0,0,0,0);

Point P1(0,0);

Point P2(0,0);

const char\* winName="Crop Image";

bool clicked=false;

int i=0;

char imgName[15];

void checkBoundary(){

//check croping rectangle exceed image boundary

if(cropRect.width>img.cols-cropRect.x)

cropRect.width=img.cols-cropRect.x;

if(cropRect.height>img.rows-cropRect.y)

cropRect.height=img.rows-cropRect.y;

if(cropRect.x<0)

cropRect.x=0;

if(cropRect.y<0)

cropRect.height=0;

}

void showImage(){

img=src.clone();

checkBoundary();

if(cropRect.width>0&&cropRect.height>0){

ROI=src(cropRect);

imshow("cropped",ROI);

}

rectangle(img, cropRect, Scalar(0,255,0), 1, 8, 0 );

imshow(winName,img);

}

void onMouse( int event, int x, int y, int f, void\* ){

switch(event){

case CV\_EVENT\_LBUTTONDOWN :

clicked=true;

P1.x=x;

P1.y=y;

P2.x=x;

P2.y=y;

break;

case CV\_EVENT\_LBUTTONUP :

P2.x=x;

P2.y=y;

clicked=false;

break;

case CV\_EVENT\_MOUSEMOVE :

if(clicked){

P2.x=x;

P2.y=y;

}

break;

default : break;

}

if(clicked){

if(P1.x>P2.x){ cropRect.x=P2.x;

cropRect.width=P1.x-P2.x; }

else { cropRect.x=P1.x;

cropRect.width=P2.x-P1.x; }

if(P1.y>P2.y){ cropRect.y=P2.y;

cropRect.height=P1.y-P2.y; }

else { cropRect.y=P1.y;

cropRect.height=P2.y-P1.y; }

}

showImage();

}

int main()

{

cout<<"Click and drag for Selection"<<endl<<endl;

cout<<"------> Press 's' to save"<<endl<<endl;

cout<<"------> Press '8' to move up"<<endl;

cout<<"------> Press '2' to move down"<<endl;

cout<<"------> Press '6' to move right"<<endl;

cout<<"------> Press '4' to move left"<<endl<<endl;

cout<<"------> Press 'w' increas top"<<endl;

cout<<"------> Press 'x' increas bottom"<<endl;

cout<<"------> Press 'd' increas right"<<endl;

cout<<"------> Press 'a' increas left"<<endl<<endl;

cout<<"------> Press 't' decrease top"<<endl;

cout<<"------> Press 'b' decrease bottom"<<endl;

cout<<"------> Press 'h' decrease right"<<endl;

cout<<"------> Press 'f' decrease left"<<endl<<endl;

cout<<"------> Press 'r' to reset"<<endl;

cout<<"------> Press 'Esc' to quit"<<endl<<endl;

src=imread("src.png",1);

namedWindow(winName,WINDOW\_NORMAL);

setMouseCallback(winName,onMouse,NULL );

imshow(winName,src);

while(1){

char c=waitKey();

if(c=='s'&&ROI.data){

sprintf(imgName,"%d.jpg",i++);

imwrite(imgName,ROI);

cout<<" Saved "<<imgName<<endl;

}

if(c=='6') cropRect.x++;

if(c=='4') cropRect.x--;

if(c=='8') cropRect.y--;

if(c=='2') cropRect.y++;

if(c=='w') { cropRect.y--; cropRect.height++;}

if(c=='d') cropRect.width++;

if(c=='x') cropRect.height++;

if(c=='a') { cropRect.x--; cropRect.width++;}

if(c=='t') { cropRect.y++; cropRect.height--;}

if(c=='h') cropRect.width--;

if(c=='b') cropRect.height--;

if(c=='f') { cropRect.x++; cropRect.width--;}

if(c==27) break;

if(c=='r') {cropRect.x=0;cropRect.y=0;cropRect.width=0;cropRect.height=0;}

showImage();

}

return 0;

}

# [**Drawing rectangle OpenCV: don't understand a detail**](http://answers.opencv.org/question/23002/drawing-rectangle-opencv-dont-understand-a-detail/)

**URL:** [**http://answers.opencv.org/question/23002/drawing-rectangle-opencv-dont-understand-a-detail/**](http://answers.opencv.org/question/23002/drawing-rectangle-opencv-dont-understand-a-detail/)

**CODE**

#include <opencv2/core/core.hpp>

#include <opencv2/highgui/highgui.hpp>

#include <opencv2/imgproc/imgproc.hpp>

#include <stdlib.h>

#include <stdio.h>

#include <math.h>

#include <string.h>

using namespace cv;

using namespace std;

void my\_mouse\_callback( int event, int x, int y, int flags, void\* param );

bool destroy=false;

Rect box;

bool drawing\_box = false;

void draw\_box(Mat \* img, Rect rect)

{

rectangle(\*img, Point(box.x, box.y), Point(box.x+box.width,box.y+box.height),Scalar(0,0,255) ,2);

Rect rect2=Rect(box.x,box.y,box.width,box.height);

}

// Implement mouse callback

void my\_mouse\_callback( int event, int x, int y, int flags, void\* param )

{

Mat\* frame = (Mat\*) param;

switch( event )

{

case CV\_EVENT\_MOUSEMOVE:

{

if( drawing\_box )

{

box.width = x-box.x;

box.height = y-box.y;

}

}

break;

case CV\_EVENT\_LBUTTONDOWN:

{ drawing\_box = true;

box = Rect( x, y, 0, 0 );

}

break;

case CV\_EVENT\_LBUTTONUP:

{ drawing\_box = false;

if( box.width < 0 )

{ box.x += box.width;

box.width \*= -1;

}

if( box.height < 0 )

{ box.y += box.height;

box.height \*= -1;

}

draw\_box(frame, box);

}

break;

default:

break;

}

}

int main()

{

String name = "Box Example";

namedWindow( name );

box = Rect(0,0,1,1);

VideoCapture capture(0) ;

if (!capture.isOpened())

{

printf("!!! Failed CaptureFromCAM\n");

return 1;

}

Mat image;

capture.read(image);

if (!image.data)

{

printf("!!! Failed \n");

return 2;

}

Mat temp = image.clone();

setMouseCallback(name, my\_mouse\_callback, &image);

while( 1 )

{

temp = image.clone();

if (drawing\_box)

draw\_box(&temp, box);

imshow(name, temp);

if (waitKey(15) == 27)

break;

}

return 0;

}

**IMPORTANT**

[**http://www.cs.iit.edu/~agam/cs512/lect-notes/opencv-intro/opencv-intro.html#SECTION00055000000000000000**](http://www.cs.iit.edu/~agam/cs512/lect-notes/opencv-intro/opencv-intro.html#SECTION00055000000000000000)

**Color Detection & Object Tracking**

URL: http://opencv-srf.blogspot.com/2010/09/object-detection-using-color-seperation.html  
Object detection and segmentation is the most important and challenging fundamental task of computer vision.  It is a critical part in many applications such as image search, scene understanding, etc. However it is still an open problem due to the variety and complexity of object classes and backgrounds.  
  
The easiest way to detect and segment an object from an image is the color based methods . The object and the background should have a significant color difference in order to successfully segment objects using color based methods.

**Simple Example of Detecting a Red Object**

In this example, I am going to process a video with a red color object and create a binary video by thresholding the red color. (Red color area of the video is assigned to '1' and other area is assigned to '0' in the binary image so that you will see a white patch wherever the red object is in the original video)  
  
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////  
**#include <iostream>**  
**#include "opencv2/highgui/highgui.hpp"**  
**#include "opencv2/imgproc/imgproc.hpp"**  
 **using namespace cv;**  
**using namespace std;**  
 **int main( int argc, char\*\* argv )**  
**{**  
**VideoCapture cap(0); //capture the video from web cam**  
 **if ( !cap.isOpened() )  // if not success, exit program**  
**{**  
**cout << "Cannot open the web cam" << endl;**  
**return -1;**  
**}**  
 **namedWindow("Control", CV\_WINDOW\_AUTOSIZE); //create a window called "Control"**  
 **int iLowH = 0;**  
 **int iHighH = 179;**  
 **int iLowS = 0;**  
 **int iHighS = 255;**  
 **int iLowV = 0;**  
 **int iHighV = 255;**  
 **//Create trackbars in "Control" window**  
 **cvCreateTrackbar("LowH", "Control", &iLowH, 179); //Hue (0 - 179)**  
 **cvCreateTrackbar("HighH", "Control", &iHighH, 179);**  
 **cvCreateTrackbar("LowS", "Control", &iLowS, 255); //Saturation (0 - 255)**  
 **cvCreateTrackbar("HighS", "Control", &iHighS, 255);**  
 **cvCreateTrackbar("LowV", "Control", &iLowV, 255); //Value (0 - 255)**  
 **cvCreateTrackbar("HighV", "Control", &iHighV, 255);**  
 **while (true)**  
**{**  
**Mat imgOriginal;**  
 **bool bSuccess = cap.read(imgOriginal); // read a new frame from video**  
 **if (!bSuccess) //if not success, break loop**  
**{**  
**cout << "Cannot read a frame from video stream" << endl;**  
**break;**  
**}**  
 **Mat imgHSV;**  
 **cvtColor(imgOriginal, imgHSV, COLOR\_BGR2HSV); //Convert the captured frame from BGR to HSV**  
  
 **Mat imgThresholded;**  
 **inRange(imgHSV, Scalar(iLowH, iLowS, iLowV), Scalar(iHighH, iHighS, iHighV), imgThresholded); //Threshold the image**  
  
 **//morphological opening (remove small objects from the foreground)**  
 **erode(imgThresholded, imgThresholded, getStructuringElement(MORPH\_ELLIPSE, Size(5, 5)) );**  
 **dilate( imgThresholded, imgThresholded, getStructuringElement(MORPH\_ELLIPSE, Size(5, 5)) );**  
 **//morphological closing (fill small holes in the foreground)**  
 **dilate( imgThresholded, imgThresholded, getStructuringElement(MORPH\_ELLIPSE, Size(5, 5)) );**  
 **erode(imgThresholded, imgThresholded, getStructuringElement(MORPH\_ELLIPSE, Size(5, 5)) );**  
 **imshow("Thresholded Image", imgThresholded); //show the thresholded image**  
 **imshow("Original", imgOriginal); //show the original image**  
 **if (waitKey(30) == 27) //wait for 'esc' key press for 30ms. If 'esc' key is pressed, break loop**  
**{**  
**cout << "esc key is pressed by user" << endl;**  
**break;**  
**}**  
**}**  
 **return 0;**  
  
**}**  
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

# [**How to draw a circle on an image using mouse click/events?**](http://answers.opencv.org/question/30518/how-to-draw-a-circle-on-an-image-using-mouse-clickevents/)

**URL: http://answers.opencv.org/question/30518/how-to-draw-a-circle-on-an-image-using-mouse-clickevents/**

#include <opencv2/imgproc/imgproc.hpp>

#include <opencv2/highgui/highgui.hpp>

#include <opencv2/opencv.hpp>

#include <iostream>

using namespace cv;

void function(int event, int x, int y, int flags, void\* param)

{

switch(event){

case CV\_EVENT\_LBUTTONDOWN:

if(flags & CV\_EVENT\_FLAG\_CTRLKEY)

{

printf("Left button clicked in coordinates %d, %d with CTRL\n",x,y);

}

else if(flags==CV\_EVENT\_LBUTTONDOWN)

{

printf("Left button clicked in coordinates %d, %d\n",x,y);

}

break;

}

}

int main( int argc, char\*\* argv )

{

int key,parameter=5;

Mat image, dst;

/// Read image ( same size, same type )

image = imread("lena.jpg");

if( !image.data )

{

printf("Error \n"); return -1;

}

else

{

printf("Image has been loaded with size %dx%d.\n",image.size);

}

circle(image,cvPoint(100,100),50,Scalar(0,255,0),1);//draw circle in point 100,100

namedWindow("My window", 1);

imshow("My window",image);

cvSetMouseCallback("My window",function,&parameter);

std::cout << "what do you want to do?: \n";

while(1)

{

key=cvWaitKey(10);//waits

switch(key)

{

case 'X':

printf("Pressed X \n");

imwrite("image\_circle.jpg",image);

return 1;

break;

case 27:

return 1;

break;

}

}

return 0;

}