**ImageRotation\_OpenCV\_C++**

**Winter Programming Bootcamp : Project 01**

#### Aim:

To rotate an image about a specific angle entered by user using OpenCV in C++.

#### Intoduction:

OpenCV (Open Source Computer Vision Library: [http://opencv.org](http://opencv.org/)) is an open-source library that includes several hundreds of computer vision algorithms.

#### Rotating Image using built-in function:

OpenCV has a built-in function **getRotationMatrix2D()** which can be used to rotate the images.

Syntax of getRotationMatrix2D():

Mat cv::getRotationMatrix2D( Point2f center, double angle, double scale);

#### Problems in using built-in function:

By rotating the image using built function image got cropped. It happens becuase when we rotate a image its center, height and width of frame also change. Due to change of height, width of frame, we do not able to view complete image and it appeared to be cropped but actually it is not cropped, it just a small windows showing image.

#### Rotating Image without Cropping:

To handle this issue, after rotating the image using built-in function, we will resize the window using some mathematics based on Pythgorus Theorm. We will find out the new height, width as well as center of the rotated image.

// finding the new height, width of window

double abs\_cos = abs(r.at<double>(0,0));

double abs\_sin = abs(r.at<double>(0,1));

double height = src.size().height;

double width = src.size().width;

int boundW = (int)(height \* abs\_sin + width \* abs\_cos);

int boundH = (int)(height \* abs\_cos + width \* abs\_sin);

// finding the new center of window

r.at<double>(0,2) += boundW / 2- width / 2;

r.at<double>(1,2) += boundH / 2 - height / 2;

// applie an affine transforation to image.

warpAffine(src, dst, r, Size(boundW, boundH));

In this case, we are combining a rotation with a translation. A transformation that can be expressed in the form of a matrix multiplication (linear transformation) followed by a vector addition (translation). The affine transformation of the rotation has to be combined with the affine transformation of translation, from the center of the original image to the center of the destination image.

In this case, we will get the correct ouput.

