OpenCV rotation function mapping

Kabir Chabbra : Diploma Business Analytics

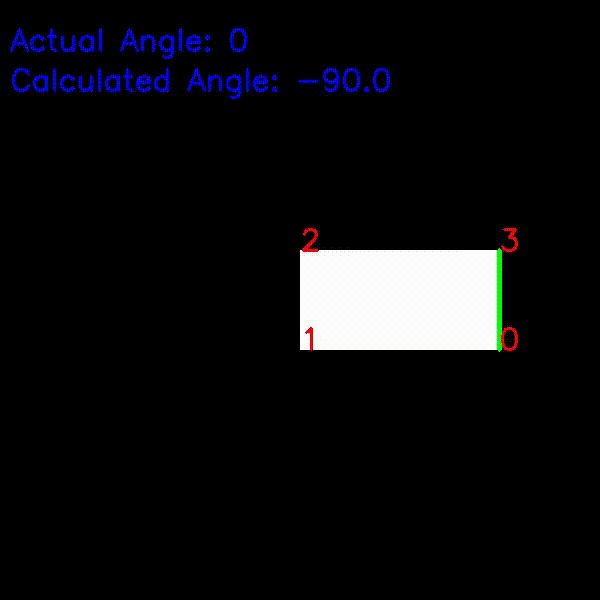
Abdul Akram Shaik : Masters in Quant fin

#### **Abstract:**

***This report outlines the approach taken to address the problem statement of mapping angles reported by the*** *minAreaRect()* ***function in the OpenCV library to the actual orientation of objects in an image. The*** *minAreaRect()* ***function reports angles only between 0 and -90 degrees, making it necessary to establish a mapping between these reported angles and the actual orientation of objects rotated from 0 to 360 degrees. Our approach involves generating synthetic data for rectangles of different sizes rotated randomly and capturing the angles reported by*** *minAreaRect()* ***along with the coordinates of the other three vertices. This data will serve as the basis for developing a mapping or function that accurately translates the reported angles to the true orientation of objects.***

**Keywords:**

* OpenCV
* Minarearect()
* Reported angle
* Object orientation
* Decision tree regression
* Synthetic data

******

#### **Introduction:**

The problem statement revolves around the discrepancy between the angles reported by the minAreaRect() function in the OpenCV library and the actual orientation of objects in an image. This discrepancy arises due to the way minAreaRect() assigns angles to the sides of the minimum bounding rectangle, with the bottom-most vertex being labeled as 0 and subsequent vertices numbered 1, 2, and 3 clockwise. To address this issue, our project aims to establish a mapping or function that correlates the reported angles with the actual orientation of objects rotated from 0 to 360 degrees.

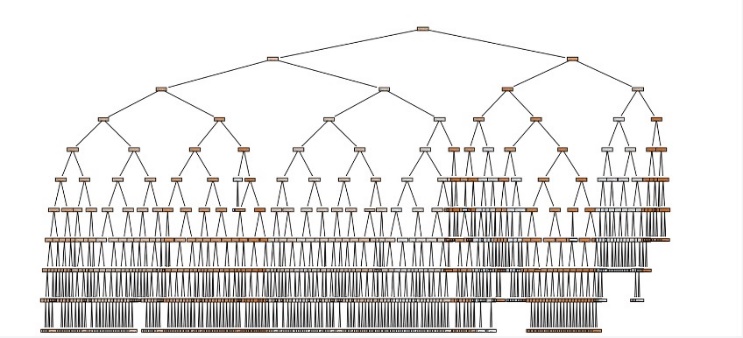
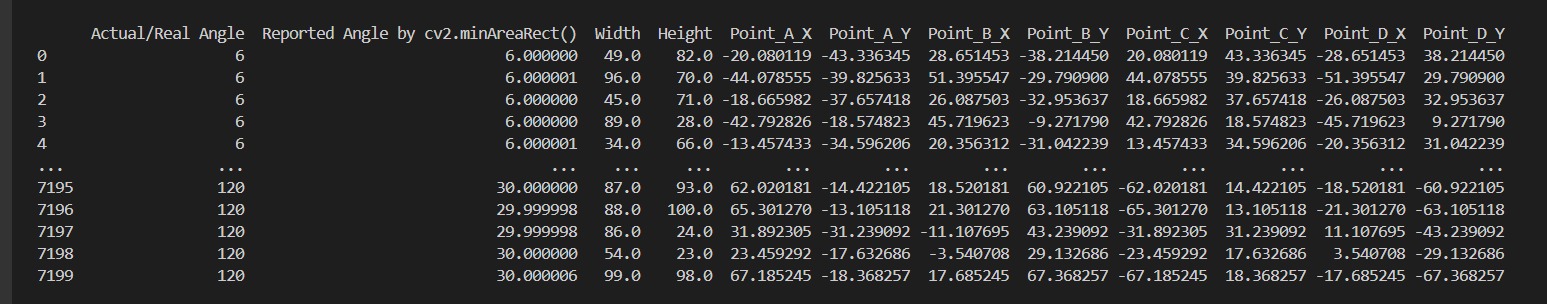
**Methodology:**

The approach taken involves the following steps:

We decided on generating a dataset, due to the specific nature of the problem and the lack of availability of datasets, our data set was generated by running iterations using OpenCV’s, Cv2.minareaRect() function, and recording the coordinates of the vertices of the image, the width and height of the image, the angle reported by minAreaRect() and the actual input angle. using these iteration we were able to generate a usable data set to train our model on.

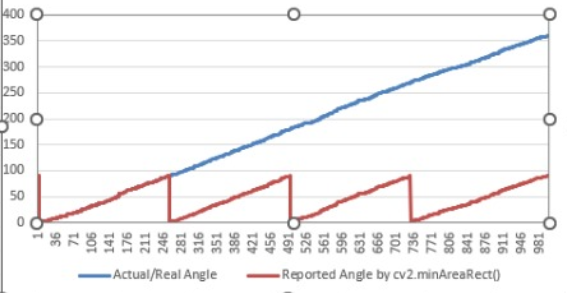
**Generation of Synthetic Data**: Rectangles of varying sizes are generated, and each rectangle is rotated randomly from 0 to 360 degrees. For each rotated rectangle, the angle reported by the minAreaRect() function is captured, along with the coordinates of the other three vertices relative to a fixed vertex around which the rectangle is rotated.

**Data Collection**: Synthetic data consisting of the reported angles and corresponding object orientations are collected for a range of rotated rectangles.

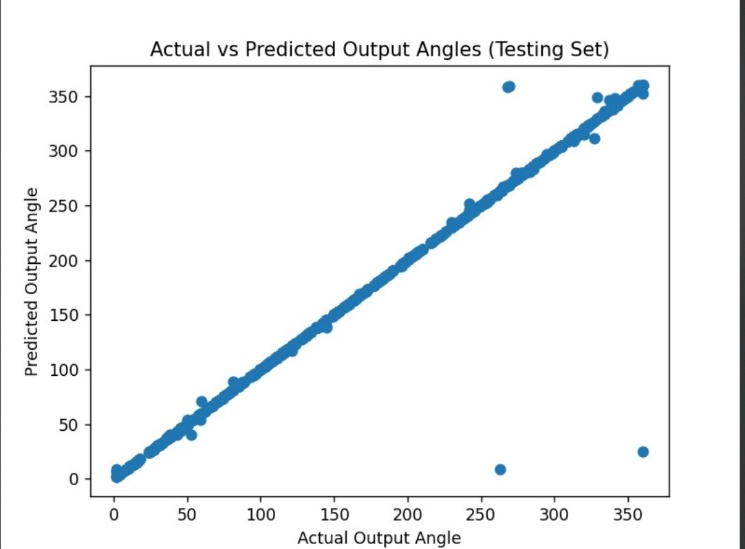
**Analysis**: Statistical analysis is performed on the collected data to identify patterns and relationships between the reported angles and the actual object orientations. The Dataset was generated by creating a type of lookup table for corresponding values of the Actual angle and the angle reported by the OpenCV command Cv2.MinAreaRect(). The angle returned by the command is between (-90,0), which makes it difficult to understand what the actual angle of the image is on the 2d plane and which quadrant it lies in. The data set has 3 columns, Actual angle, reported angle and Pixel Position. The pixel position column is a set of observations, constituted of the pixel position of the top left corner of an image for every angle or degree of change in the Actual Angle.

**Mapping Function Development**: Based on the analysis, Decision tree regression is employed as mapping function to predict the actual angle of an object based on reported values and vertex coordinates. Grid search is utilized to determine the optimal maximum depth of the decision tree.

**Results and Discussion:**



Preliminary analysis of the synthetic data reveals a complex relationship between the reported angles and the actual object orientations. While the reported angles exhibit a limited range (-90 to 0 degrees), the true orientations span the entire 360-degree range.



The decision tree regression model achieves promising results in predicting the actual angle of objects based on reported values and vertex coordinates. Evaluation metrics including mean absolute error, mean squared error, root mean squared error, and coefficient of determination demonstrate the effectiveness of the proposed approach in accurately mapping reported angles to actual object orientation. Visualizations of actual vs. predicted output angles and decision tree structure further illustrate the performance of the model.

**Discussion:**  
The proposed approach provides a viable solution to the challenge of mapping reported angles to actual object orientation in a 2D image plane. However, further experimentation and refinement may be required to enhance the accuracy and robustness of the model. Future research directions may include exploring alternative machine learning algorithms and incorporating additional features to improve prediction performance.

The comparatively higher MSE can be explained due to some predictions being extreme outliers which skews the calculation. Overall the accuracy of our model is quite high and whilst this usually points towards overfitting, we were able to determine that since the problem was simplistic in nature and also since the training and test data were extremely similar as well, the model was able to adapt well to both, to confirm that the model is not overfitted we also ran a K-folds validation, where the Mean Squared Error dropped to 75.811.

**Conclusion:**  
In conclusion, this report presents an approach to create a mapping/function between OpenCV's reported angle and actual object orientation in a 2D image plane. The use of decision tree regression on synthetic data demonstrates the feasibility of accurately predicting object orientation based on reported values and vertex coordinates. The findings of this study contribute to advancing the understanding and application of image processing techniques in object .

**References:**

[1] OpenCV Documentation: https://docs.opencv.org/  
[2] Python Documentation: <https://docs.python.org/>

[3]<https://forum.opencv.org/t/rotate-the-bounding-box-of-object-detection-and-crop-it/7949>

[4]<https://stackoverflow.com/questions/61718596/how-to-rotate-bounding-box-in-open-cv-and-crop-it-python>

[5]https://stackoverflow.com/questions/59651764/python-opencv-how-to-calculate-angle-of-object-wrt-a-sample