# DOCUMENTATION FOR ARK PERCEPTION TEAM TASK

## TASK-1 Gebe dich nie auf

## I.INTRODUCTION

The task at hand involves image analysis and password recovery. It presents a scenario where a famous π image is corrupted with noise. Using the corrupted pixel values, we need to design a 2x2 filter. Another distorted artwork is suspected to be a copy of a famous portrait. The goal is to analyze the images, remove noise, restore the original portrait, and generate a password based on the recovered image with the help of template matching and then we need to perform the RRT-Connect algorithm on a maze image obtained from a zip file using the password generated.

## II. PROBLEM STATEMENT

The problem consists of the following tasks:

- Removing noise from the corrupted π image by comparing it with digits of π and applying a specific conversion to the distorted digits.
- Restoring a distorted artwork image by applying the filter using bitwise operations.
- Scaling the restored artwork image to the appropriate dimensions for template matching.
- Performing template matching from scratch to find the coordinates of the matched template in the collage image.
- Generating a password by manipulating the coordinates and using π.
- Writing the RRT-connect algorithm for finding the path in the maze image (from start point to end point) provided.

## III. INITIAL ATTEMPTS

First it was required to find out the relationship with the image and π. On printing each pixel of the pi image in row wise order it was found that the pixels correspond to the digits of π, multiplied by 10. At only four places the pixel value was 255(white) and on dividing by 10, there is so such digit 25.5 in π. So, the distorted digits were found out at those places where pixel value was 255. Then the filter was formed.

Applying the filter on the image was difficult and required a lot of guesses. So, I rather chose applying the filter on the black and white image of the portrait and could obtain a black and white portrait which completely matched one of the images in the collage.png.

Then after generating the password, I could begin with RRT-Connect algorithm.

## IV. FINAL APPROACH

The final approach was to write the code for performing the necessary activities. The explanation of the code is as follows:

1. **template_Match()** Function: This function is defined to perform template matching. It takes two arguments: img (the image to be matched against) and template (the template image). The function calculates the dimensions and size of the template and the image, initializes variables, and applies template matching using a sliding window approach. It finds the best match by comparing pixel values and returns the coordinates of the top left corner of the matched template.
2. **compute_pi()** Function: This function is defined to compute the digits of π. It takes an argument n to determine the precision of π calculations. The function uses the Bailey-Borwein-Plouffe formula to calculate the digits of π and returns the computed value as a string. The compute_pi() function is called with a precision value of 2499 to compute the digits of π.
3. The π image (pi_img) is read using OpenCV's imread() function, and its height and width are extracted. An array a is initialized to store the digits of π. The code loops through the image pixels, divides the pixel value by 10, and stores the result as an integer in the array a. The first element of a is set to '.' to represent the decimal point.
4. Filter Generation: A filter array is initialized to store the filter values. The code compares the computed π digits with the corresponding digits from the image and checks for distortion. If distortion is detected, the filter value is calculated by multiplying the distorted digit by 10π and converting it to an integer. The filter values are sorted in descending order.
5. Artwork Restoration: The distorted artwork image (artwork_picasso.png) is read in grayscale mode using imread() with the flag IMREAD_GRAYSCALE. The dimensions of the image are extracted. The code applies bitwise operations (XOR, OR, AND) between the filter values and the image pixels, updating the pixel values. This process is performed in a nested loop over the image, applying the filter element-wise. The restored image is stored in img2.
6. Template Matching and Password Generation: The collage image (collage.png) is read in grayscale mode. The restored artwork template (img2) is matched against the collage image using the

template_Match() function. The top left corner coordinates of the matched template are obtained. The password is calculated by summing the x and y coordinates, multiplying by π, and converting the result to an integer. Finally, the password is printed.

7. RRT CONNECT algorithm:
    a. Image Loading and Preprocessing: The maze image (maze1.png) is read in grayscale mode using imread() with the flag 0. The image is then converted to a binary representation, where black pixels are set to 1 and white pixels to 0.
    b. treeNode Class: This class represents a node in the RRT tree. Each node has a location (locationX, locationY), a list of children (children), and a parent node (parent).
    c. rrtAlogo Class: This class represents the RRT algorithm. It takes the start and goal locations, the number of iterations, the grid (maze), and the step size as input parameters. It initializes the RRT tree with the start location as the root node (randomTree) and the goal location as a separate node (goal).
    d. addChild() Function: This function adds a child node to the nearest node in the RRT tree. If the child node's location matches the goal location, the nearest node is added as a parent of the goal node and goal node is added as the child of nearest node.
    e. samplePoint() Function: This function generates a random point within the maze grid.
    f. moveToPoint() Function: This function calculates the new point by moving a specified step size (rho) towards the destination point (locationEnd) from the current point (locationStart).
    g. isInObstacle() Function: This function checks if any point on the line segment between locationStart and locationEnd lies within an obstacle. It iterates over the line segment and checks the corresponding grid cell's value in the maze. If any cell is 1 (obstacle), it returns True; otherwise, it returns False.
    h. unitVector() Function: This function calculates the unit vector between locationStart and locationEnd.
    i. findNearest() Function: This function recursively searches the RRT tree for the nearest node to a given point. It updates nearestNode and nearestDist if a closer node is found.
    j. distance() Function: This function calculates the Euclidean distance between node1 and a given point.
    k. goalFound() Function: This function checks if the goal point is within a specified distance (rho) of the goal point.
    l. resetNearestValues() Function: This function resets the nearestNode and nearestDist variables to initial values.
    m. retraceRRTPath() Function: This function retraces the path from the goal node to the start node in the RRT tree. It appends the waypoints (locations) to the waypoints list and updates the path_distance and numWaypoints variables.
    n. Plotting the Maze: The maze grid is displayed using imshow() with a binary colormap. The start and goal locations are plotted as red and blue circles, respectively.
    o. Path Retracing and Waypoints: The path from the goal to the start is retracted using the retraceRRTPath() function. The waypoints are inserted into the waypoints list, and the path distance and number of waypoints are printed. The waypoints are plotted in the maze using plot() and pause() functions. The resulting path is saved as plot image_2.png

## V. RESULTS AND OBSERVATION

The code successfully addresses the problem statement by removing noise, restoring the artwork, performing template matching, and generating the password. The restored artwork image is displayed, and the generated password is printed as the output. The RRT connect algorithm then successfully finds a path for both the instances.

## VI. FUTURE WORK

Here are some potential improvements and future enhancements:

- A better algorithm could be used to extract the image in coloured form.
- A more efficient RRT Connect algorithm could be used which takes the shortest path.

## VII. CONCLUSION

The provided code demonstrates effective image analysis techniques and password recovery methods. It showcases the utilization of mathematical computations, bitwise operations, and template matching to recover valuable information from corrupted and distorted images. Finally, a path planning algorithm is applied to find an appropriate path from start point to end point avoiding the obstacles.

## VIII. REFERENCES

- https://levelup.gitconnected.com/generating-the-value-of-pi-to-a-known-number-of-decimals-places-in-python-e93986bb474d
- https://www.geeksforgeeks.org/opencv-python-tutorial/