MINI PROJECT REPORT

JIGSAW PUZZLE GENERATOR: AN INTERACTIVE IMAGE TRANSFORMATION TOOL

Submitted by

HATLIN JOHEIT J A

(22PDS807)

In Partial Fulfillment of the Requirements for the award of the degree

of

MASTER OF SCIENCE

in

DATA SCIENCE

Under the Guidance of

Dr. I. PRIYA STELLA MARY, MCA, MPhil, NET, Ph.D.,

ST. JOSEPH'S COLLEGE (AUTONOMOUS)

ACCREDITED AT A++ GRADE (4th Cycle) BY NAAC

SPECIAL HERITAGE STATUS AWARDED BY UGC

COLLEGE WITH POTENTIAL FOR EXCELLENCE BY UGC

TIRUCHIRAPPALLI - 620 002

SEPTEMBER – 2023

DEPARTMENT OF DATA SCIENCE

ST. JOSEPH'S COLLEGE (AUTONOMOUS)

TIRUCHIRAPPALLI - 620 002

CERTIFICATE

This is to certify that this project report "**JIGSAW PUZZLE GENERATOR: AN INTERACTIVE IMAGE TRANSFORMATION TOOL"** is the bonafide work of **HATLIN JOHEIT J A 22PDS807,** who carried out the project work under my supervision. Certified further that to the best of my knowledge the work reported here in does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE	**SIGNATURE**
Dr. I. PRIYA STELLA MARY	**Dr. L. AROCKIAM**
SUPERVISOR	**HEAD OF THE DEPARTMENT**
Department of Data Science	Department of Data Science
St. Joseph's College (Autonomous)	St. Joseph's College (Autonomous)
Tiruchirappalli – 620 002	Tiruchirappalli – 620 002

Submitted for Semester Mini-Project viva-voce examination held on ________________

Internal Examiner	**External Examiner**
Date:	Date:

ACKNOWLEDGEMENT

First of all, I am indebted to the Almighty for his choice and blessing showered on me in the completing this endeavor.

I express my sincere gratitude to **Rev. Dr. LEONARD FERNANDO SJ,** Rector St. Joseph's College (Autonomous) Tiruchirappalli - 620 002, for providing me with this opportunity to complete my mini-project.

I extend my gratitude to **Rev. Dr. K. AMAL SJ,** Secretary St. Joseph's College (Autonomous) Tiruchirappalli - 620 002 for giving me this opportunity.

I proudly grateful to **Rev. Dr. M. AROCKIASAMY XAVIER, S.J.** Principal, St.Joseph's college, Tiruchirappalli - 620 002, for having given me an opportunity to pursue mystudy and use the facilities available in this institution.

I would like to acknowledge **Dr. L. AROCKIAM**, Head, Department of Data Science, St.Joseph's college, Tiruchirappalli - 620 002, for the moral support and encouragement he had rendered throughout the course.

I am highly indebted to the internal guide **Dr. I. PRIYA STELLA MARY,** Assistant Professor, Department of Data Science for her guidance and constant supervision as well as for providing necessary information regarding the project and also for her support in completing the project.

I would also like to express my sincere regards and thanks to all our department staffs, friends and family, whose support, inspiration and kind help have been the strong pillars of support throughout my completion of this mini-project.

HATLIN JOHEIT J A

22PDS807

ABSTRACT

The "Jigsaw Puzzle Generator" is a Python-based image manipulation tool designed for creating interactive jigsaw puzzles from digital images. Leveraging computer vision and image processing libraries such as OpenCV and NumPy, this tool provides users with the capability to transform ordinary images into engaging puzzles. Users can specify puzzle grid sizes and target dimensions, tailoring the puzzle-solving experience to their preferences and requirements.

This project presents an in-depth exploration of the Jigsaw Puzzle Generator's functionality; encompassing image resizing, puzzle piece shuffling, and the assembly of both solved and unsolved puzzle images. The tool's versatility is highlighted, showcasing its potential applications in art, education, and entertainment. With its intuitive user interface and customization options, the Jigsaw Puzzle Generator offers a unique blend of creativity and interactivity. Whether employed by art enthusiasts seeking to turn their favorite artwork into an enjoyable puzzle or educators searching for innovative image manipulation teaching tools, this utility opens doors to diverse and engaging possibilities.

This abstract provides a concise overview of the Jigsaw Puzzle Generator's capabilities and its potential to serve as a versatile and entertaining tool for image transformation and educational purposes.

TABLE OF CONTENT

1. INTRODUCTION

In an increasingly digital world, image manipulation and transformation have become integral aspects of creative expression, entertainment, and education. Jigsaw puzzles, a classic and universally recognized form of entertainment, challenge individuals to assemble a complete image from smaller, interlocking pieces. The process of solving a jigsaw puzzle can be both intellectually stimulating and enjoyable. In the context of digital media, creating and solving jigsaw puzzles can offer a unique and engaging experience.

This project introduces a Python-based tool, "Jigsaw Puzzle Generator," which leverages the power of computer vision and image processing libraries like OpenCV and NumPy to transform ordinary images into interactive jigsaw puzzles. This tool allows users to specify the size of the puzzle grid and the target dimensions of the puzzle, enabling customization to suit their preferences and requirements.

The Jigsaw Puzzle Generator not only showcases the capabilities of image processing but also provides a fun and interactive experience for users to enjoy. Whether you are an art enthusiast looking to turn your favorite painting into an engaging puzzle or an educator seeking an innovative way to teach image manipulation, this tool offers a versatile solution.

2. PROBLEM STATEMENT

The problem addressed by the "Jigsaw Puzzle Generator" project is to develop a Python-based tool that can transform digital images into interactive jigsaw puzzles. The primary objective is to create a user-friendly application that allows users to take any image as input and generate both solved and unsolved versions of a jigsaw puzzle from it.

The challenge lies in effectively splitting the input image into puzzle pieces that can be reassembled to recreate the original image, ensuring that the pieces align seamlessly. Additionally, the tool should offer customization options, allowing users to define parameters like grid size and target dimensions to tailor the puzzle to their preferences. It should provide an interactive display of the generated puzzles, with the ability to save the puzzle pieces and images to disk for sharing and enjoyment. The project aims to offer an engaging and creative experience, making it suitable for entertainment and artistic purposes.

3. TECHNOLOGY ADOPTED

3.1.1 Anaconda Navigator

Anaconda is an open-source distribution of Python and R programming languages, primarily used for data science and machine learning tasks. It offers a comprehensive package management system that simplifies the installation and management of libraries and dependencies. Anaconda includes a wide range of preinstalled data science tools, making it a powerful and convenient platform for data analysis and scientific computing. It also provides an integrated development environment called Anaconda Navigator, which offers a user-friendly interface for package management and project development. Anaconda is favored by data scientists, researchers, and analysts for its versatility, ease of use, and support for creating isolated environments to manage project dependencies effectively.

3.1.2 Jupyter Notebook

Jupyter Notebook is an open-source web application commonly used for interactive data analysis, scientific computing, and documentation. It allows users to create and share documents containing live code, equations, visualizations, and narrative text. Jupyter Notebook supports multiple programming languages, with Python being the most popular choice. Users can execute code cells interactively, making it a valuable tool for data exploration, research, and education. Its user-friendly interface and integration with libraries like Matplotlib and NumPy make it a preferred choice for data scientists and researchers for creating reproducible and interactive data-driven reports.

3.2 Libraries Used:

3.2.1 OpenCV (cv2)

OpenCV, or Open Source Computer Vision Library, is a crucial library for image processing and computer vision tasks. In this project, OpenCV is used for various image manipulation operations, including resizing images, splitting them into pieces, and displaying them.

3.2.2 NumPy

NumPy is a fundamental library for numerical computing in Python. It provides support for multidimensional arrays and mathematical functions. In this code, NumPy is employed for array operations and calculations.

4. METHODOLOGY

4.1 Importing Libraries:

- The project begins by importing the necessary Python libraries, specifically OpenCV (`cv2`) and NumPy. These libraries provide essential tools and functions for image processing and numerical computations.

4.2 Defining the resize_image Function:

- The code defines a function called resize_image that takes an input image and resizes it to a specified target size. This function is based on OpenCV's resize function.

4.3 Defining the create_jigsaw_puzzle Function:

- The core functionality of the project is encapsulated in the create_jigsaw_puzzle function. This function takes several parameters, including the input image, grid size, and target size (optional).

4.4 Resizing the Input Image:

- If a target size is provided, the input image is resized using the resize_image function. This step ensures that the input image conforms to the desired dimensions.

4.5 Splitting the Image into Puzzle Pieces:

- The input image is divided into a grid of puzzle pieces. The size of each puzzle piece is calculated based on the grid size and the dimensions of the resized image.

4.6 Creating a Shuffled Order:

- A list of indices representing a row-wise shuffle is generated. This list of indices determines the order in which the puzzle pieces will be arranged to create the unsolved puzzle.

4.7 Arranging Puzzle Pieces for the Solved Image:

- The code creates a solved image by arranging the puzzle pieces in their original order. It iterates through the grid and places each piece in its correct position.

4.8 Creating the Unsolved Image:

- To generate an unsolved jigsaw puzzle, the code shuffles the list of puzzle pieces. It then assembles the pieces on the canvas in a random order, creating a challenging puzzle for the user to solve.

4.9 Saving Images:

- The code allows users to save the generated puzzle pieces, the solved image, and the unsolved image to disk. This step facilitates the sharing and distribution of the created puzzles.

4.10 Displaying Images:

- The OpenCV library is used to display both the unsolved and solved jigsaw puzzle images to the user. This interactive display provides an engaging experience for users to interact with the generated puzzles.

4.11 User Interaction:

- The code waits for user interaction, typically a key press, before closing the OpenCV windows. This interaction allows users to view the puzzles and exit the program when they are finished.

5. WORKFLOW

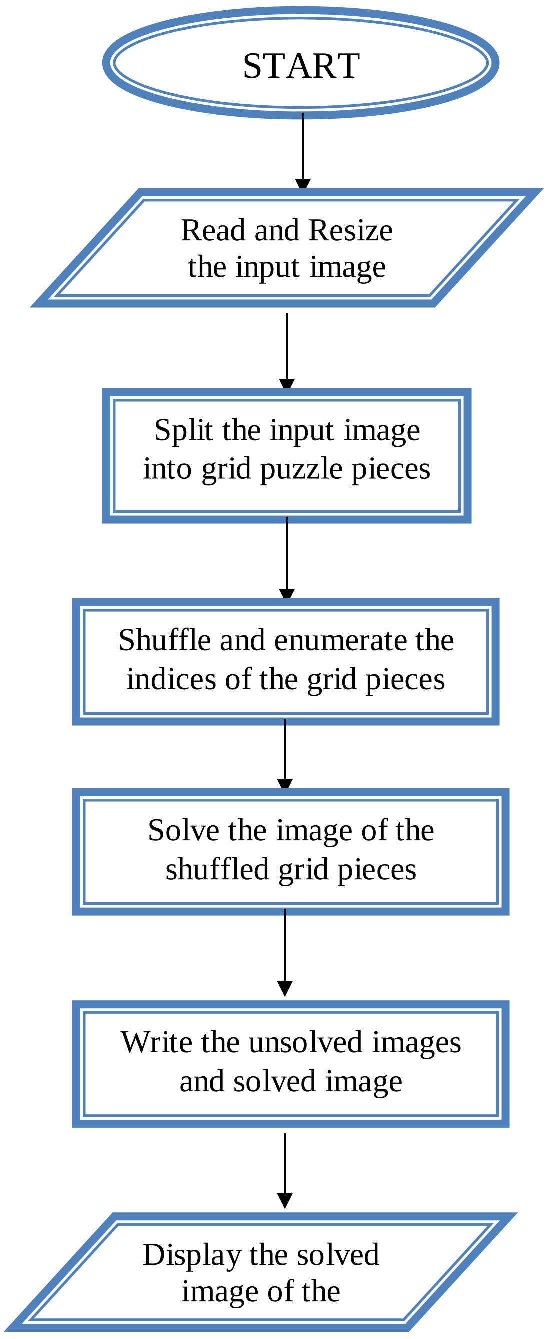

START
Read and Resize the input image
Split the input image into grid puzzle pieces
Shuffle and enumerate the indices of the grid pieces
Solve the image of the shuffled grid pieces
Write the unsolved images and solved image
Display the solved image of the

Fig5.1. Workflow

6. CONCLUSION

In conclusion, the "Jigsaw Puzzle Generator" project presents a creative and interactive application that transforms ordinary digital images into engaging jigsaw puzzles. This project leverages the power of Python, the OpenCV library, and NumPy for image processing and manipulation, allowing users to customize the puzzle's grid size and dimensions. The methodology involves splitting the input image into puzzle pieces, arranging them to form a solved image, and then shuffling them to create an unsolved version that challenges users.

By offering an intuitive user interface for image display and interaction, this project provides a unique blend of entertainment and education. Users can explore their creativity by generating custom puzzles or enjoy solving puzzles generated from their favorite images. Furthermore, the ability to save puzzle pieces and images to disk enhances the project's practicality for sharing and distribution.

This project showcases the versatility and utility of Python and its libraries in image processing and computer vision. It serves as a testament to the endless possibilities of technology and its potential to transform everyday experiences into fun and interactive endeavors. The "Jigsaw Puzzle Generator" offers a glimpse into the world of digital creativity, making it a valuable addition to both the hobbyist's toolkit and the educational landscape.

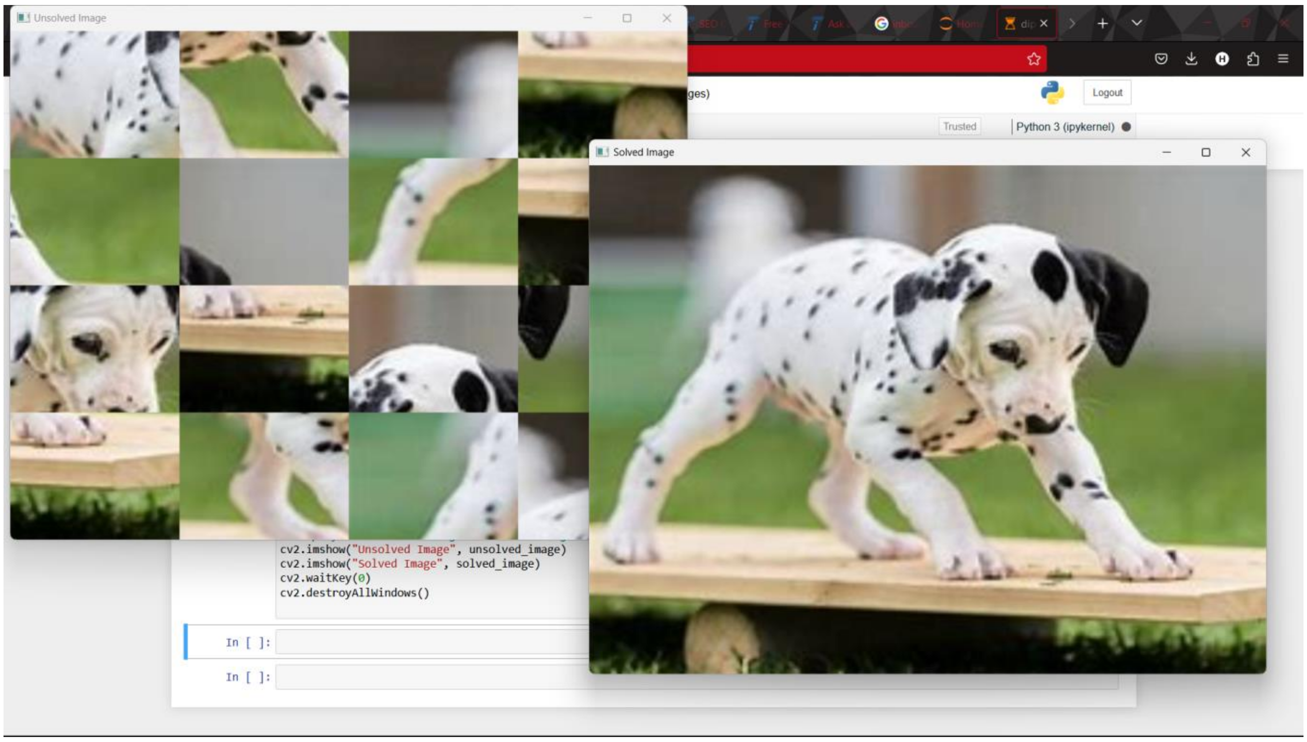
7. APPENDIX

```
import cv2
import numpy as np

def resize_image(input_image, target_size):
    """
    Resize the input image to the target size.
    """
    return cv2.resize(input_image, target_size)

def create_jigsaw_puzzle(input_image, grid_size=(4, 4), target_size=None):
    if target_size:
        input_image = resize_image(input_image, target_size)

    # Split the input image into a grid of puzzle pieces.
    puzzle_pieces = []
    puzzle_piece_size = input_image.shape[0] // grid_size[0], input_image.shape[1] // grid_size[1]

    # Create a list of indices representing a row-wise shuffle.
    indices = list(range(grid_size[0] * grid_size[1]))
    np.random.shuffle(indices)

    for i, idx in enumerate(indices):
        row, col = divmod(i, grid_size[1])
        x_start = col * puzzle_piece_size[1]
        x_end = (col + 1) * puzzle_piece_size[1]
        y_start = row * puzzle_piece_size[0]
        y_end = (row + 1) * puzzle_piece_size[0]
        puzzle_pieces.append(input_image[y_start:y_end, x_start:x_end])

    # Create a solved image of the puzzle.
    solved_image = np.zeros((input_image.shape[0], input_image.shape[1], input_image.shape[2]),
dtype=np.uint8)
```

```
    for row in range(grid_size[0]):
        for col in range(grid_size[1]):
            solved_image[row * puzzle_piece_size[0]:(row + 1) * puzzle_piece_size[0],
                          col * puzzle_piece_size[1]:(col + 1) * puzzle_piece_size[1]] = puzzle_pieces[row *
grid_size[1] + col]

    # Create an unsolved image by shuffling the pieces.
    unsolved_image = np.copy(solved_image)
    np.random.shuffle(puzzle_pieces)
    for row in range(grid_size[0]):
        for col in range(grid_size[1]):
            unsolved_image[row * puzzle_piece_size[0]:(row + 1) * puzzle_piece_size[0],
                            col * puzzle_piece_size[1]:(col + 1) * puzzle_piece_size[1]] = puzzle_pieces[row *
grid_size[1] + col]

    return puzzle_pieces, solved_image, unsolved_image

# Example usage:
input_image = cv2.imread("demohj.jpg")

# Resize the input image to a specific target size, e.g., (800, 600).
target_size = (800, 600)
puzzle_pieces, solved_image, unsolved_image = create_jigsaw_puzzle(input_image, grid_size=(4,
4), target_size=target_size)

# Save the puzzle pieces, solved image, and unsolved image to disk.
for i in range(len(puzzle_pieces)):
    cv2.imwrite(f"puzzle_piece_{i}.jpg", puzzle_pieces[i])
cv2.imwrite("solved_image.jpg", solved_image)
cv2.imwrite("unsolved_image.jpg", unsolved_image)

# Display the unsolved image and solved image.
cv2.imshow("Unsolved Image", unsolved_image)
cv2.imshow("Solved Image", solved_image)
```

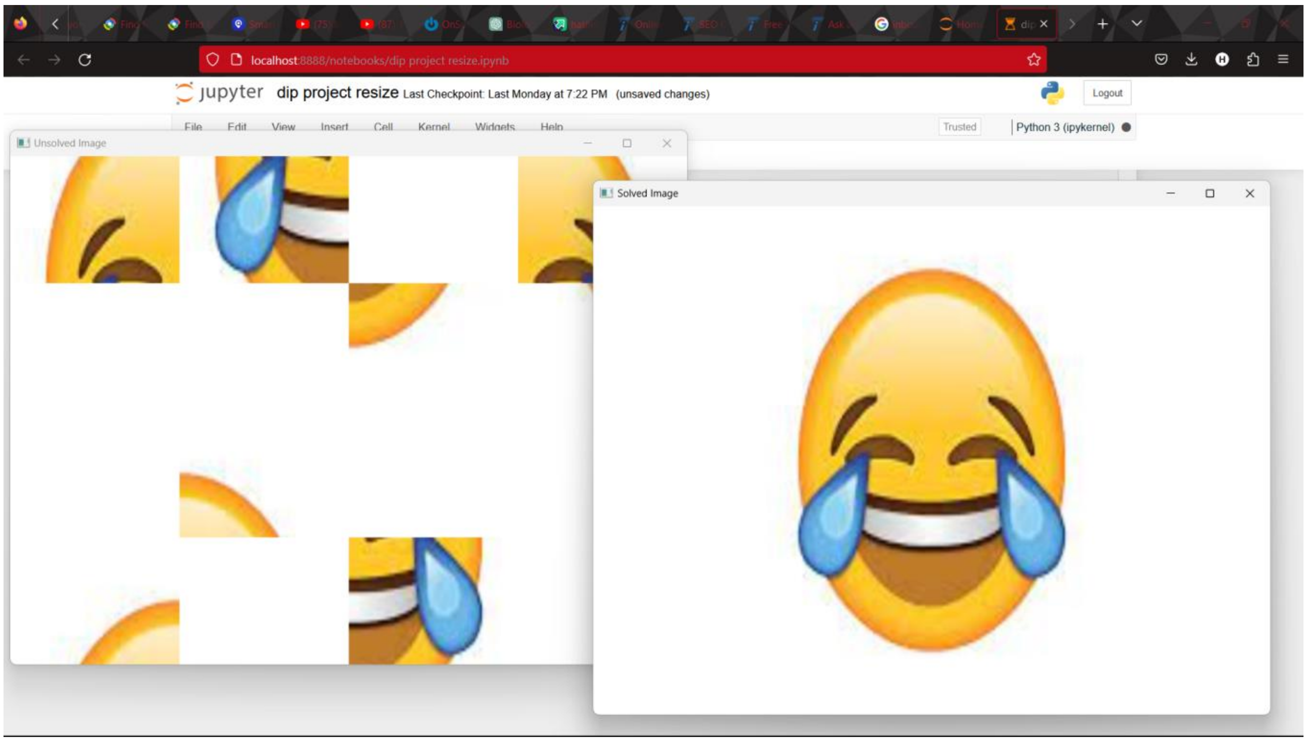
```
cv2.waitKey(0)
cv2.destroyAllWindows()
```

RESULTS:



```
cv2.imshow("Unsolved Image", unsolved_image)
cv2.imshow("Solved Image", solved_image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

```
In [ ]:

In [ ]:
```


localhost:8888/notebooks/dip project resize.ipynb
jupyter dip project resize Last Checkpoint: Last Monday at 7:22 PM (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Unsolved Image
Solved Image

REFERENCES

1. https://towardsdatascience.com/solving-jigsaw-puzzles-with-python-and-opencv-d775ba730660?gi=955ed94b5f20
2. https://www.abtosoftware.com/blog/computer-vision-powers-automatic-jigsaw-puzzle-solver
3. https://medium.com/cornell-tech/jigsolved-computer-vision-to-solve-jigsaw-puzzles-70b8ad8099e5
4. https://www.cs.rochester.edu/users/faculty/nelson/courses/vision/assignments/assg_jigsaw.html