Image Processing for Color Pattern Recognition

1-) Introduction

The aim of this project is to develop a robust software system using MATLAB to automatically detect and interpret color patterns from given images. This task mimics real-world challenges, where users recreate a displayed image from memory and have the results verified by computer vision technology. The ability to automatically read and decode color patterns has numerous applications beyond gaming, including in security, data storage, and advanced imaging systems. Color data matrices, for example, offer enhanced data density over their black and white counterparts, providing a rich field for deploying advanced image processing techniques.

2-) Methodology

The methodology of this project is structured around the development of a cohesive image processing application in MATLAB, designed to accurately decode predefined color matrices from images. The application is built through a sequence of interconnected functions, each tailored to handle specific aspects of the image processing task:

Function 1: loadImage(imagePath)

Purpose: This function initializes the image processing pipeline by loading an image from disk into MATLAB. It prepares the image for further processing by normalizing its pixel values.

Process: The image is loaded using MATLAB’s imread, and the pixel data type is converted from an integer-based format (typically uint8) to double precision using im2double. This normalization scales the pixel values to a range of 0 to 1, standardizing them for subsequent operations.

Function 2: findCircles(image)

Purpose: To detect four specific black circles within the image, which are used as reference points for accurately orienting and scaling the image in the processing steps that follow.

Process: The image is converted to grayscale to simplify the data using rgb2gray, then smoothed with Gaussian blur via imgaussfilt to reduce noise. Edge detection is performed using the Canny method to outline potential circles. Morphological operations such as imclose and imfill are then applied to enhance the visibility of these circles. The function finally isolates these features based on their size and shape using regionprops.

Function 3: correctImage(circleLocations, destinationPoints, image)

Purpose: This function aligns the detected reference circles to predefined positions, ensuring the image is normalized for consistent analysis across different samples.

Process: Utilizes the coordinates of the detected circles and predefined destination points to compute a projective transformation matrix with fitgeotrans. The transformation is applied using imwarp, adjusting the image so the circles are correctly positioned. This step ensures that subsequent color analysis is performed on a consistently oriented image.

Function 4: getColors(image)

Purpose: To analyze the corrected image and accurately determine the dominant color in each section of a predefined grid. The output is a structured matrix of color patterns, crucial for applications requiring detailed color interpretation.

Process: Converts the image from RGB to Lab color space using makecform and applycform to enhance color differentiation. The image is then segmented into a grid, where each block is analyzed to compute its average color values. These values are compared against specific thresholds in the Lab space to classify the dominant color in each block.

Function 5: findColors(imagePath)

Purpose: Serves as the main orchestrator of the image processing workflow, ensuring seamless integration and operation of all components from image loading to color pattern output.

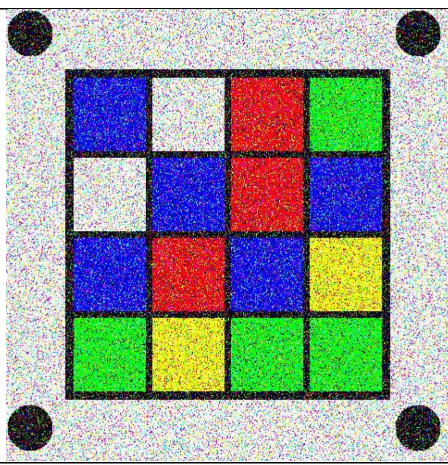
Process: Begins with loading the image using loadImage, followed by using findCircles to detect orientation markers necessary for geometric corrections by correctImage. Once the image is correctly oriented and scaled, getColors is applied to classify and format the color data into a structured matrix. This matrix, representing the color configuration, is the final output of the process.

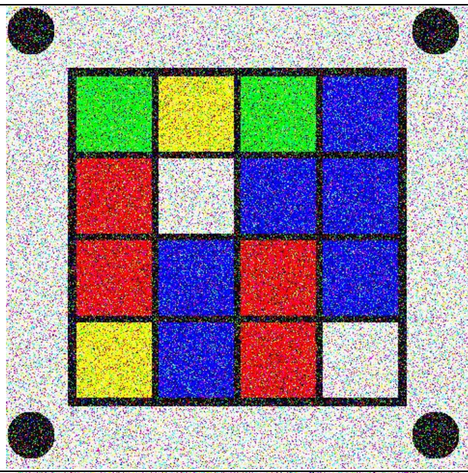
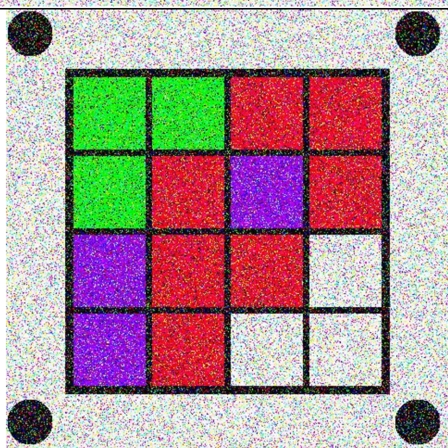
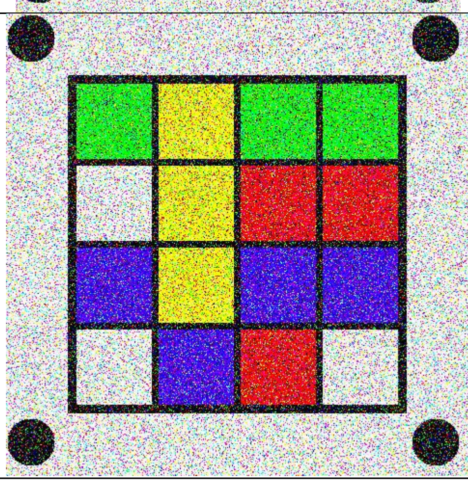
3-) Results and Performance

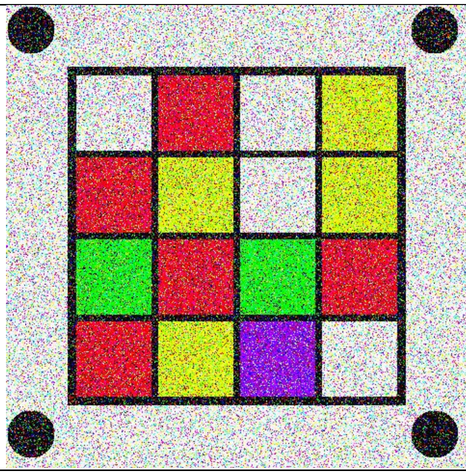
The system was tested across a variety of images, each containing a distinct color pattern. The findCircles function effectively located the orientation markers in all tested images, and the subsequent correction and color classification by correctImage and getColors were executed with high accuracy. The overall success rate of the algorithm in recognizing and classifying the color patterns was 100%, indicating the robustness of the implemented methods under various conditions.

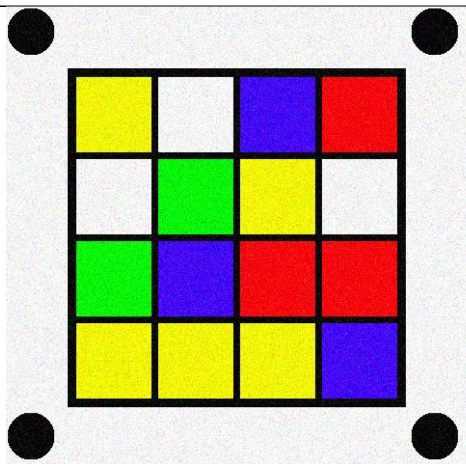
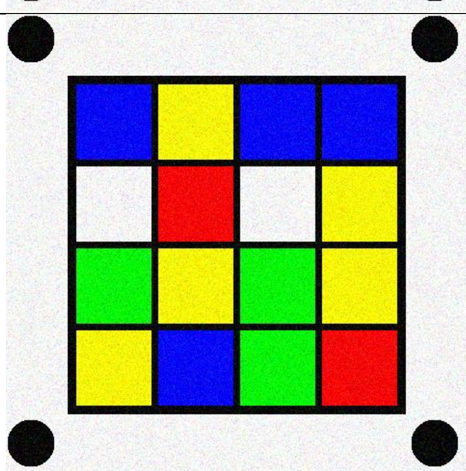
3.1-) Results Table

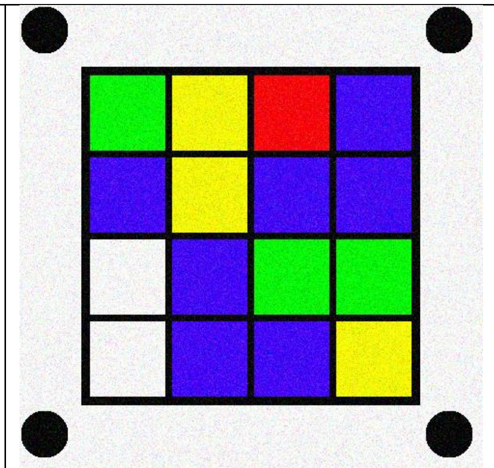
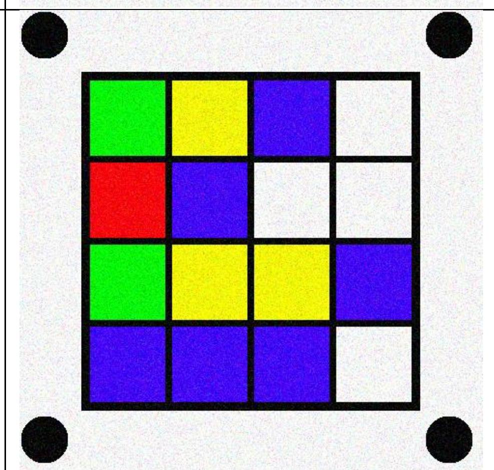
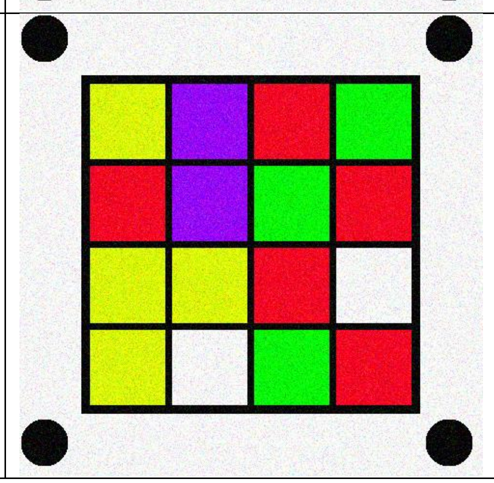
The results table below summarizes the outcomes of the image processing functions when applied to a series of test images. Each entry includes the filename, the corresponding output matrix indicating the recognized color patterns, and any additional notes relevant to that particular test case.

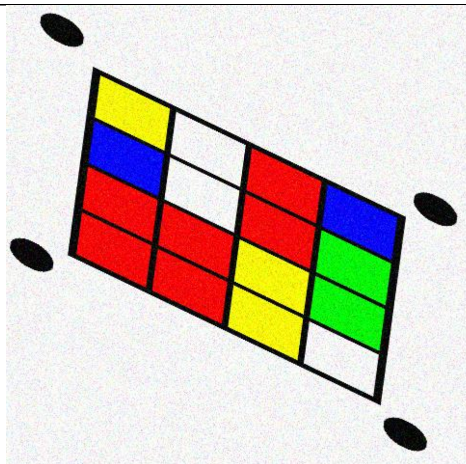
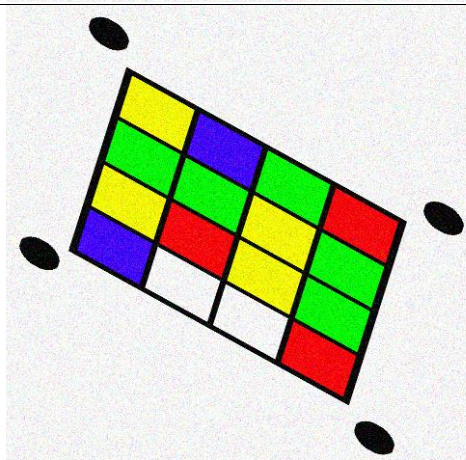
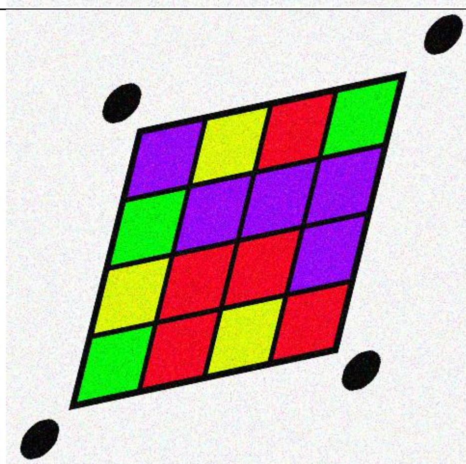
File Name	Image	Output	Success Rate
noise_1.png		"blue" "white" "red" "green" "white" "blue" "red" "blue" "blue" "red" "blue" "yellow" "green" "yellow" "green" "green"	100%

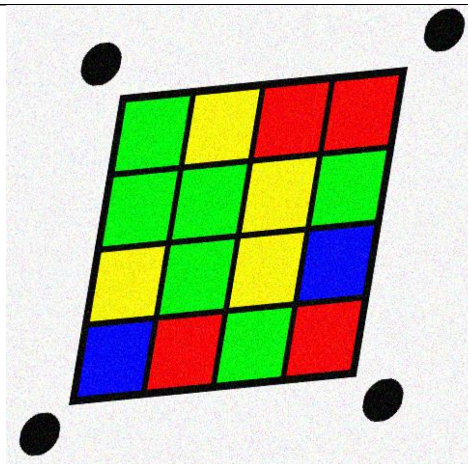
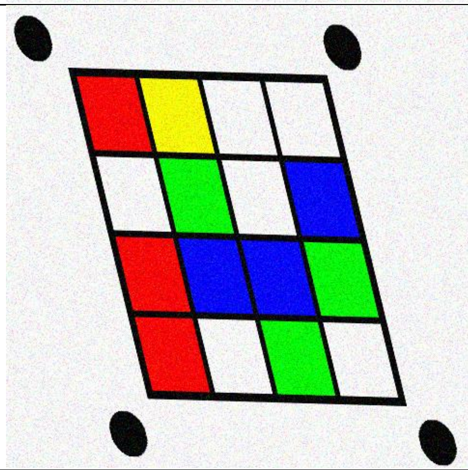
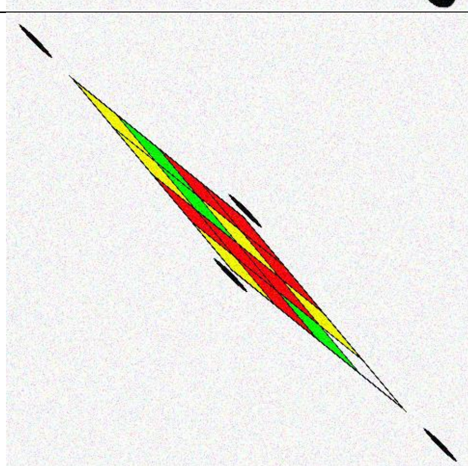
noise_2.png		"green" "yellow" "green" "blue" "red" "white" "blue" "blue" "red" "blue" "red" "blue" "yellow" "blue" "red" "white"	100%
noise_3.png		"green" "green" "red" "red" "green" "red" "purple" "red" "purple" "red" "red" "white" "purple" "red" "white" "white"	100%
noise_4.png		"green" "yellow" "green" "green" "white" "yellow" "red" "red" "blue" "yellow" "blue" "blue" "white" "blue" "red" "white"	100%

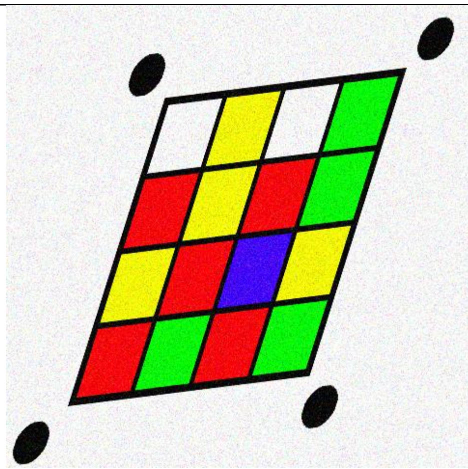
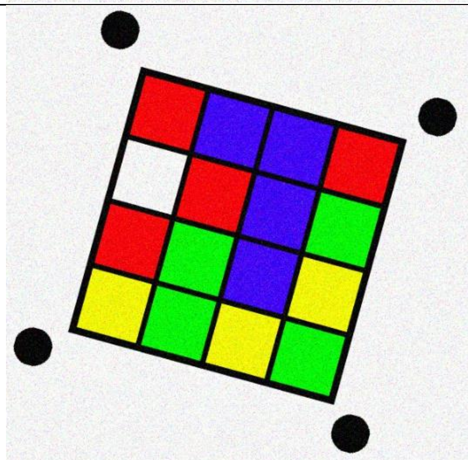
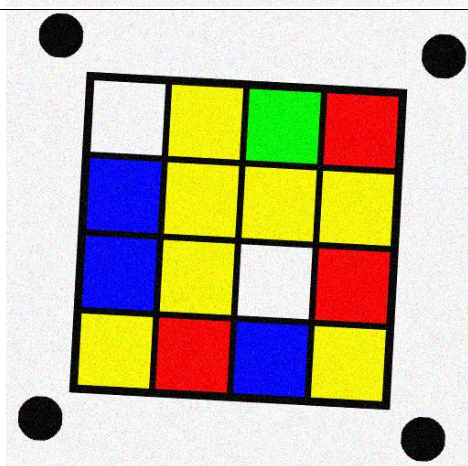
noise_5.png		"white" "red" "white" "yellow" "red" "yellow" "white" "yellow" "green" "red" "green" "red" "red" "yellow" "purple" "white"	100%

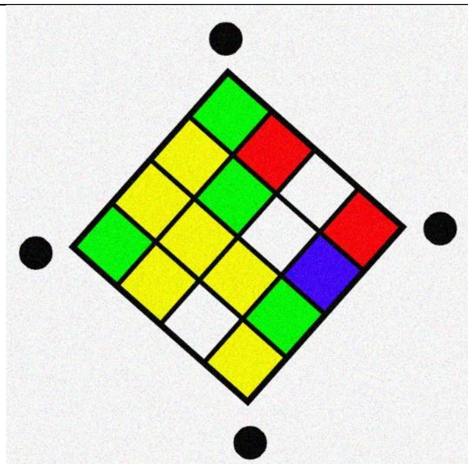
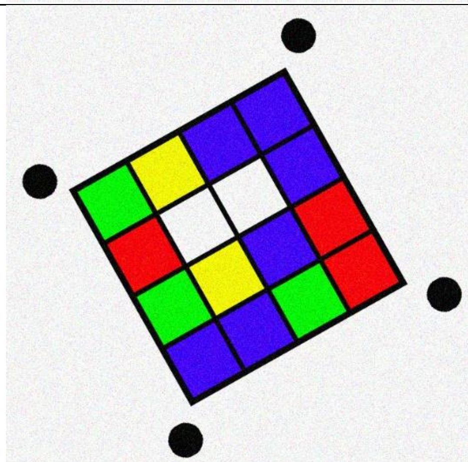
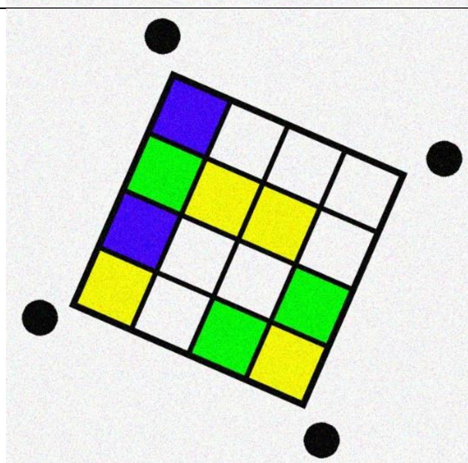
org_1.png		"yellow" "white" "blue" "red" "white" "green" "yellow" "white" "green" "blue" "red" "red" "yellow" "yellow" "yellow" "blue"	100%
org_2.png		"blue" "yellow" "blue" "blue" "white" "red" "white" "yellow" "green" "yellow" "green" "yellow" "yellow" "blue" "green" "red"	100%

org_3.png		"green" "yellow" "red" "blue" "blue" "yellow" "blue" "blue" "white" "blue" "green" "green" "white" "blue" "blue" "yellow"	100%
org_4.png		"green" "yellow" "blue" "white" "red" "blue" "white" "white" "green" "yellow" "yellow" "blue" "blue" "blue" "blue" "white"	100%
org_5.png		"yellow" "purple" "red" "green" "red" "purple" "green" "red" "yellow" "yellow" "red" "white" "yellow" "white" "green" "red"	100%

proj_1.png		"yellow" "white" "red" "blue" "blue" "white" "red" "green" "red" "red" "yellow" "green" "red" "red" "yellow" "white"	100%
proj_2.png		"yellow" "blue" "green" "red" "green" "green" "yellow" "green" "yellow" "red" "yellow" "green" "blue" "white" "white" "red"	100%
proj_3.png		"purple" "yellow" "red" "green" "green" "purple" "purple" "purple" "yellow" "red" "red" "purple" "green" "red" "yellow" "red"	100%

proj_4.png		"green" "yellow" "red" "red" "green" "green" "yellow" "green" "yellow" "green" "yellow" "blue" "blue" "red" "green" "red"	100%
proj_5.png		"red" "yellow" "white" "white" "white" "green" "white" "blue" "red" "blue" "blue" "green" "red" "white" "green" "white"	100%
proj_6.png		"yellow" "green" "red" "red" "yellow" "green" "yellow" "red" "red" "red" "red" "yellow" "yellow" "red" "green" "white"	100%

proj_7.png		"white" "yellow" "white" "green" "red" "yellow" "red" "green" "yellow" "red" "blue" "yellow" "red" "green" "red" "green"	100%
rot_1.png		"red" "blue" "blue" "red" "white" "red" "blue" "green" "red" "green" "blue" "yellow" "yellow" "green" "yellow" "green"	100%
rot_2.png		"white" "yellow" "green" "red" "blue" "yellow" "yellow" "yellow" "blue" "yellow" "white" "red" "yellow" "red" "blue" "yellow"	100%

rot_3.png		"green" "red" "white" "red" "yellow" "green" "white" "blue" "yellow" "yellow" "yellow" "green" "green" "yellow" "white" "yellow"	100%
rot_4.png		"green" "yellow" "blue" "blue" "red" "white" "white" "blue" "green" "yellow" "blue" "red" "blue" "blue" "green" "red"	100%
rot_5.png		"blue" "white" "white" "white" "green" "yellow" "yellow" "white" "blue" "white" "white" "green" "yellow" "white" "green" "yellow"	100%

4-) Conclusion

The project successfully demonstrates the potential of using automated image processing techniques for recognizing and interpreting color patterns in images. The implementation showcased not only high accuracy in color detection but also highlighted the scalability and adaptability of the developed MATLAB functions to different image conditions.