Final RPM Project

Michael Lukacsko
mlukacsko3@gatech.edu

1 AGENT DESIGN

The agent addresses 2x2 and 3x3 Raven's Progressive Matrices (RPM) problems by systematically identifying patterns and relationships among the images in the problem matrix. As illustrated in Figure 1 below, for 2x2 problems, the agent employs an affine algorithm along rows and columns to predict the missing image. For 3x3 problems, depending on specific problem types, the agent analyzes relationships between images across rows, columns, and sometimes diagonals.

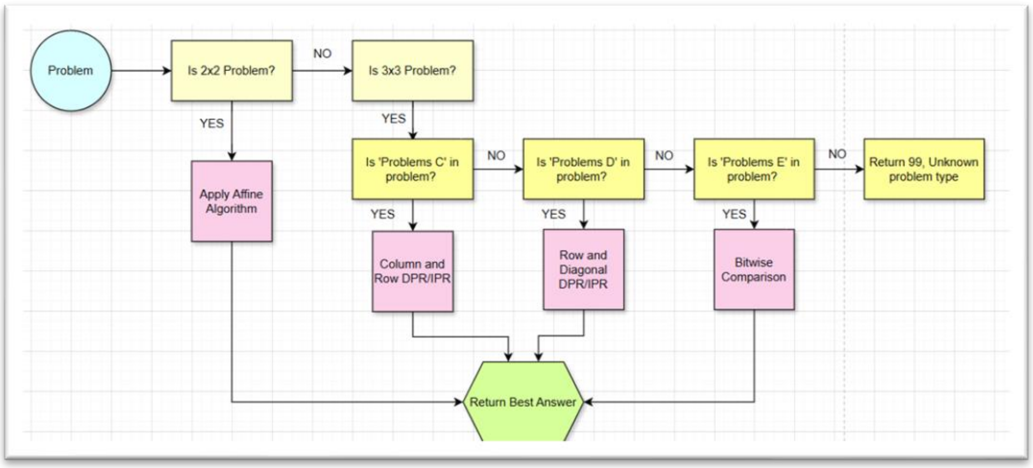

Figure 1 — RPM Agent Problem Solving Flowchart.

1.1 2x2 Problem Solving

The agent solves 2x2 Raven's Progressive Matrices (RPM) problems through a series of detailed steps designed to analyze relationships between images, generate predictions, and select the most probable answer.

1.1.1 *Load Images and Analyzing Relationships*

The agent begins by loading images A, B, and C for the problem using OpenCV's cv2.imread, as well as the six possible answer options. After converting these images into grayscale, using cv2.IMREAD_GRAYSCALE, for consistency, it analyzes the relationships between images in the given matrix. More specifically, it examines the transformations that relate image A to B (row relationship) and

A to C (column relationship). This is achieved by identifying the best-fit affine transformations that describe these relationships. This step considers various transformations, including rotations and reflections, to determine the optimal transformation matrices using mean squared error.

1.1.2 *Generate Probable Answers Using Affine Transformations*

With the identified transformations, the agent generates a predicted answer image using the relationships derived from the row and column. First, it applies the transformation derived from A to B to image C, producing a row-based predicted image. Similarly, it applies the transformation derived from A to C to image B, generating a column-based predicted image. These transformations allow the agent to infer what the missing image should look like.

1.1.3 *Combining Predicted Answers*

To refine the prediction, the agent combines the two predicted images, by averaging their pixel values. This is accomplished using the OpenCV cv2.addWeighted function, which blends the two images with equal weights, ensuring both predicted images contribute equally to the result. This process produces a single combined predicted image which represents the agent's best guess for the missing image. By combining predictions from both row and column relationships, the agent ensures its solution captures consistent patterns across the matrix.

1.1.4 *Generate a Second Probable Answer Using XOR Transformations*

In parallel, the agent generates a second predicted answer using logical XOR transformations. It first computes the XOR of images A and B, capturing the differences between these two images, and then applies the result to image C. This produces a second predicted image which captures potential logical patterns in the matrix.

1.1.5 *Determine the Best Match*

Finally, the agent compares both predicted images with each of the six answer options. It calculates the Mean Squared Error (MSE) for each comparison, where a lower MSE indicates greater similarity. The agent identifies the best match from each approach, affine and XOR, by selecting the answer option with the lowest MSE. To finalize its solution, the agent selects the overall best match by comparing the MSE scores from both approaches and returning the option with the smallest value.

1.2 3x3 Problem Solving

For 3x3 problems, the agent employs a systematic approach to solve each challenge by analyzing relationships between images in rows, columns, and sometimes diagonals. Like the process for 2x2 problems, the agent first loads and preprocesses the images using OpenCV, converting them into binary formats for computational efficiency. Depending on the problem type, it utilizes methods like Dark Pixel Ratio (DPR), Image Pixel Ratio (IPR), and bitwise operations to identify patterns and transformations that define the relationships among images. These methods are effective for 3x3 problems because they capture both pixel-level and geometric similarities, enabling the agent to evaluate relationships across more complex matrices. Each problem set (C, D, and E) is addressed using a tailored methodology to ensure accurate predictions.

1.2.1 *Problem Set C*

For problems in Problem Set C, the agent focuses on analyzing relationships in rows and columns using Dark Pixel Ratio (DPR) and Image Pixel Ratio (IPR) metrics, as these provide robust ways to capture pixel density changes and structural overlap between images. These metrics are particularly effective in Problem Set C because the relationships between adjacent images in the rows and columns often involve transformations, such as changes in shape size, pixel density, or geometric patterns.

The agent begins by calculating the DPR and IPR values for adjacent images in rows (e.g., A to B, D to E, G to H) and stores these as reference values. DPR captures global pixel density changes, allowing the agent to detect additive or subtractive transformations, while IPR focuses on structural overlaps, capturing geometric consistencies such as rotations or flips. Together, these metrics ensure the agent considers both quantitative and qualitative aspects of the relationship.

Next, for each answer option, the agent computes the DPR and IPR values between the last image in the row (H) and the candidate answer option. It filters the candidates by checking if the DPR values are within a set threshold of the reference DPR values for the row. This step ensures that only answer options that match the pixel density transformations observed in the row are considered.

If no candidates meet this threshold, the agent selects the answer with the closest DPR value, maintaining consistency with the pixel density pattern.

For candidates that pass the DPR threshold, the agent evaluates their IPR values and selects the option with the closest IPR match to the reference IPR value for the row. This logic ensures that the selected answer not only adheres to pixel density changes but also maintains structural and geometric relationships observed in the problem matrix. By combining DPR for density and IPR for structure, the agent effectively captures the relationships in Problem Set C, providing accurate predictions for the missing image.

1.2.2 *Problem Set D*

For problems in Problem Set D, the agent extends its analysis to include horizontal, diagonal, and inverse diagonal relationships, leveraging both DPR and IPR metrics to capture pixel density changes and structural overlaps across these orientations. This approach is particularly effective in Problem Set D, as the relationships between images often go in multiple directions and are more complex transformations such as diagonal progressions or inverse patterns.

The agent begins by calculating DPR and IPR values for image pairs in horizontal (e.g., A to B), diagonal (e.g., A to E), and inverse diagonal (e.g., F to A) orientations. These reference metrics provide the foundation for identifying transformations specific to the row, diagonal, or inverse diagonal patterns in the matrix. Again, the DPR metric captures pixel-level density changes, which are useful for detecting additive or subtractive pixel alterations, while the IPR metric identifies geometric or structural consistencies, such as rotational or reflective transformations.

Next, for each candidate option, the agent computes the DPR and IPR values relative to the final image in each orientation (e.g., H for horizontal, E for diagonal, B for inverse diagonal) and filters candidate options based on whether their DPR values fall within a specified threshold range of the reference DPR values for each orientation. Reusing some of the logic from problem set C, the agent evaluates the filtered candidates IPR values and selects those that best match the reference IPR values for the respective orientations.

To determine the best overall answer, the agent evaluates the cumulative differences between the reference metrics (both DPR and IPR) and the candidate metrics across all three orientations (horizontal, diagonal, and inverse diagonal). The agent then selects the option with the lowest cumulative difference, ensuring that the chosen answer aligns closely with all directional relationships in the 3x3 matrix. This multi-directional approach ensures the agent captures the more complex patterns and transformations seen with Problem Set D questions.

1.2.3 *Problem Set E*

For problems in Problem Set E, the agent uses a bit-wise analysis approach to identify the missing image by employing logical operations. This method is particularly effective for detecting pixel-wise patterns and transformations that are frequent in Problem Set E, where the logical operations XOR, AND, OR, and NOT often define the relationships between images in the matrix.

The process begins by applying bit-wise operations to pairs of images in the matrix. For example, the agent might compute the XOR operation between images A and B to generate an intermediate image. This intermediate result is then combined with a third image (e.g., image C) using a bit-wise operation to produce a predicted image. Similarly, the agent applies other operations such as AND, OR, and NOT, generating multiple predicted images for comparison.

To evaluate the validity of these predictions, the agent compares each predicted image with the target image (H) and all candidate answer options using the Tversky similarity metric. This metric provides a measure of similarity by quantifying pixel overlap between two images while accounting for false positives and false negatives. By calculating Tversky scores for each answer option, the agent determines which candidate aligns most closely with the logical patterns identified in the problem matrix.

The final step involves selecting the answer option with the highest similarity score based on the Tversky metric. This makes certain that the chosen option aligns with the observed logical operations and transformations. By focusing on bitwise logic and pixel-level relationships, the agent effectively captures the distinctive patterns in Problem Set E and provides robust solutions for these challenges.

2 AGENT PERFORMANCE

The performance of this agent was evaluated using both Gradescope and local testing. Specific to Gradescope, the agent passed 28 out of 48 Basic problems (58.3%), 31 out of 48 Test problems (64.6%), 13 out of 48 Challenge problems (27.1%), and 26 out of 48 Raven's problems (54.2%). This demonstrates strong performance on simpler problem sets like Basic and Test problems, where patterns are more structured and easier to identify, while performance dropped significantly on Challenge problems involving complex relationships and transformations. Local testing results corroborated these findings, with the agent passing 28 Basic problems (58.3%) and 13 Challenge problems (27.1%).

3 AGENT SUCCESS

3.1 2x2 Success - Basic Problem B-04

Referencing appendix section 7.1.1 figure 2 (Basic Problem B-04), the agent successfully identifies the correct answer by analyzing the transformation across rows in the 2x2 grid. For the row containing images A and B, the agent detects a rotation transformation. In particular, the black wedge in image A rotates clockwise by 90 degrees to form image B. Applying the same transformation to the column containing image C, the agent predicts that the missing image should be a clockwise rotation of 90 degrees from image C.

The agent achieves this prediction by:

1. Finding the Best Transformation: It computes the best affine transformation between image A and image B, identifying the clockwise rotation.
2. Applying the Transformation: The agent applies this transformation to image C to generate the predicted image.
3. Comparing Predictions with Options: The predicted image is compared with each of the six options using Mean Squared Error (MSE) to find the best match.

As a result, the agent selects Option 3 as the correct answer, as it aligns with the rotational transformation identified in the first row.

3.2 3x3 Success Using DPR/IPR - Basic Problem D-07

In this problem, found in appendix section 7.1.2, the agent successfully determines the correct answer by analyzing relationships along rows, columns, and diagonals. For this problem, the agent identifies that each row maintains a specific pattern based on the shapes within the images. In the first row (A, B, C), each image contains a distinct symbol enclosed in a unique frame, with no repetition of shapes or frames. Similarly, the second row (D, E, F) adheres to the same logic, with each image having a different shape and frame combination.

To predict the missing image in the third row, the agent examines the relationship between G and H. It identifies that the pattern involves unique combinations of shapes and frames, and the missing image must include a symbol and frame combination not already present in row three.

The agent processes this prediction as follows:

1. DPR and IPR Calculations: It calculates the DPR and IPR values for each pair of images in the row and compares them with the candidate answer options.
2. Threshold Filtering: The agent filters candidate options based on DPR values within a specified threshold to narrow down potential matches. Here, options like 2, 6, and 7 are dropped as possible answers.
3. Best Match Selection: The agent evaluates the remaining candidates using IPR values and selects the one that best fits the observed pattern of unique shapes and frames.

The agent selects Option 1 as the correct answer because it completes the row with a unique combination of a cross symbol and a distinct frame, adhering to the logic observed in the other rows.

3.3 3x3 Success using Bitwise Operations - Basic Problem E-02

Basic Problem E-02, found in the appendix section 7.1.3, is an example of the agent leveraging bitwise operations to identify the correct answer by analyzing relationships across the rows of the 3x3 matrix. Each row in this problem shows a progression of combining geometric shapes using logical operations. For instance, in the first row (A, B, C), image A and image B combine to form image C through the logical inclusion of overlapping features. Similarly, in the second row (D, E, F), the agent identifies that image D and image E combine to produce the overlapping features in image F.

To predict the missing image in the third row, the agent applies the same logical approach using bitwise operations:

1. Bitwise XOR Operations: The agent performs a bitwise XOR operation between images G and H to generate a predicted image that highlights the non-overlapping features of these two images.
2. Bitwise AND/OR Operations: The agent further evaluates other logical bitwise operations (AND, OR, and NOT) to refine the prediction and account for potential transformations.
3. Tversky Similarity Metric: The agent compares the predicted image with each of the candidate options using the Tversky similarity metric, ensuring a precise match based on pixel-level relationships.
4. Selection of Best Match: Among the candidates, the agent selects the image that achieves the highest similarity score with the predicted image.

The agent correctly identifies Option 7 as the answer because it is the logical combination of the features in images G and H, completing the row's pattern with overlapping and non-overlapping geometric features.

4 AGENT STRUGGLES

4.1 2x2 Agent Struggle - Basic Problem B-09

In Basic Problem B-09, pictured in appendix section 7.2.1, the agent struggles due to its reliance on transformations that fail to account for color inversion. In this problem, image A is an outlined octagon, and image B is a filled black octagon, representing a clear color inversion. Similarly, image C is an outlined square, and the missing image should be a filled black square to follow the same pattern. However, the agent's approach primarily focuses on affine transformations and Mean Squared Error (MSE) comparisons, which were not designed to detect changes in color or filling. Consequently, the agent incorrectly predicts the answer as Option 1, matching the geometric shape but failing to identify the necessary inversion, leading to an incorrect solution.

4.2 3x3 Agent Struggle - Challenge Problem E-12

Pictured in appendix section 7.2.2, Challenge Problem E-12, the agent encounters difficulties due to its inability to account for progressive changes in the number of objects. In this example, the problem presents a sequence where the number of birds on a perch increases across each row, with an incremental addition in every subsequent image. Because the agent relies on bitwise operations and similarity metrics to predict the missing image in this problem set, it fails to recognize and quantify this numerical progression effectively. As a result, the agent incorrectly identifies an option that matches the overall pixel structure but does not satisfy the progression pattern, leading to an erroneous solution.

5 OVERALL AGENT APPROACH

5.1 Agent Characterization

The design of this agent is best characterized as an ever-expanding suite of heuristics, tailored to address specific patterns and problem types. Instead of relying on a unified, formal AI model or a generalized machine learning approach, the agent employs a collection of distinct strategies such as affine transformations, bitwise operations, and metrics like Dark Pixel Ratio (DPR) and Image Pixel Ratio (IPR). Each heuristic is designed to handle specific visual or geometric relationships observed in the problem sets. As new challenges arise, additional heuristics are integrated into the agent's framework to address the specific nuances of those problems. This heuristic-driven design makes the agent less of a formal AI system and more of an iterative problem-solving toolkit. A formal AI agent typically operates under a unified decision-making framework, such as a neural network or a rule-based reasoning system, that applies broadly to a range of problems. In contrast, this agent's heuristic approach relies on a growing list of hand-engineered methods that solve problems in a piecemeal manner. While this allows for flexibility and targeted problem-solving, it also limits the agent's ability to generalize across unseen or highly varied problem types, as each heuristic must be manually designed and integrated. This approach reflects a tradeoff between adaptability to specific challenges and the broader, more autonomous capabilities of formal AI agents.

5.2 Iterative Refinement

In the development of the agent, one approach that was attempted but ultimately discarded was the Additive Deduction Convergence (ADC) Method for solving all of the problems in the 3x3 problem set. The ADC method aimed to analyze image relationships by combining additive and deductive transformations, generating a predicted image for comparison with candidate answers. However, during testing, this approach yielded suboptimal results, failing to consistently produce accurate predictions for the diverse patterns in the 3x3 problems. As a result, the ADC method was dropped in favor of the more effective Dark Pixel Ratio (DPR) and Image Pixel Ratio (IPR) metrics, which provided a more robust framework for identifying pixel-level and geometric relationships in rows, columns, and diagonals.

6 HUMAN VS AGENT

For 2x2 problems, a human often relies on intuitive recognition of visual patterns or transformations, such as rotations, reflections, or object manipulations, by observing the relationships between images in rows and columns. The agent, on the other hand, relies on mathematical operations like affine transformations and Mean Squared Error (MSE) comparisons to identify these relationships, which is more mechanical and devoid of the abstract reasoning humans employ.

For 3x3 problems, humans might analyze relationships across rows, columns, and sometimes diagonals by looking for repeating visual or logical patterns. Humans can abstract relationships like addition or subtraction of shapes, movement, or scaling without needing pixel-level calculations. The agent, however, systematically uses pixel-based metrics such as Dark Pixel Ratio (DPR) and Image Pixel Ratio (IPR), as well as logical bitwise operations, to compute the relationships and predict the missing image. While effective, this approach lacks the higher-level reasoning and generalization capabilities of humans.

In terms of speed, the agent processes problems much faster than a human could. The agent can solve problems in milliseconds, whereas it would take a human significantly longer. This computational efficiency highlights the advantage of an algorithmic approach, even though it lacks the flexibility and intuition that humans bring to problem-solving. However, the trade-off is that the agent may fail in scenarios requiring abstract understanding, where a human may excel.

7 APPENDICES

7.1 Agent Success Images

7.1.1 *Example 1 – 2x2 Success*

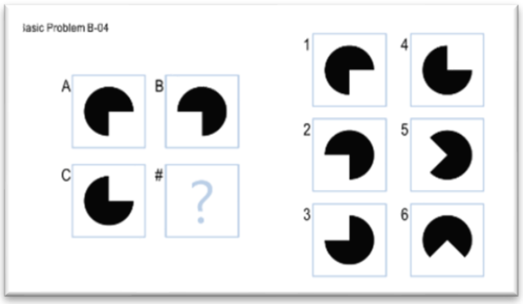

Figure 2— Basic Problem B-04 From 2x2 Problem Set.

7.1.2 *Example 2 – 3x3 Success Using DPR/IPR*

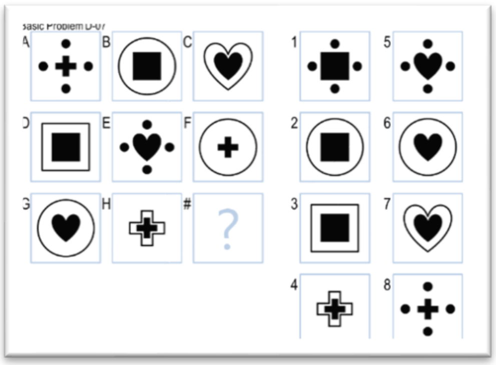

Figure 3— Basic Problem D-07 From 3x3 Problems Solved Using DPR/IPR.

7.1.3 *Example 3 – 3x3 Success Using Bitwise Operations*

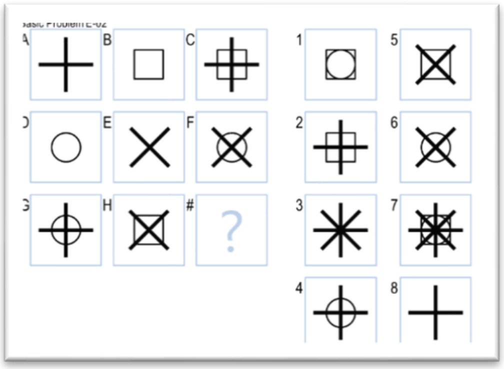

Figure 4— Basic Problem E-02 From 3x3 Problems Solved Using Bitwise Operations.

7.2 Agent Struggles Images

7.2.1 *Example 1 - 2x2 Agent Struggle*

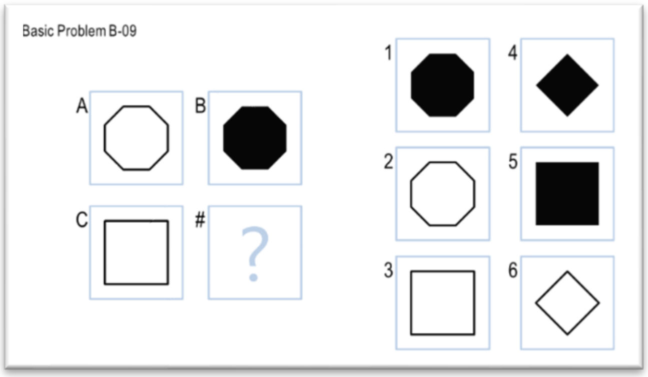

Figure 5 — Basic Problem B-09 From 2x2 Problem Set.

7.2.2 *Example 2 - 3x3 Agent Struggle*

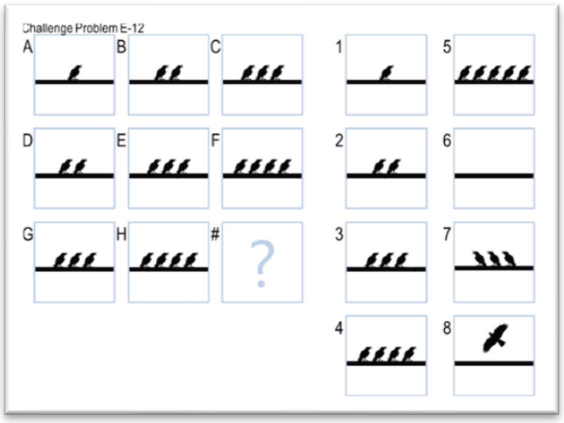

Figure 6 — Challenge Problem E-12 From 3x3 Problem Set.