Raven's Progressive Matrices Project: Final Project

Jorge Salvador Aguilar Moreno
jmoreno@gatech.edu

Abstract—The final project report outlines the final implementation of a knowledge-based artificial intelligence (KBAI) agent that solves 2x2 and 3x3 Raven's Progressive Matrix tests. The implementation uses pixel comparisons for the solution and solves 86 out of 96 tests correctly.

1 INTRODUCTION

The KBAI agent presented in this final report extends and finalizes the implementation from previous Milestones to solve Raven's Progressive Matrix (RPM) tests for 2x2 and 3x3 matrices. The solution relies on pixel comparisons as heuristics to solve the RPM tests. The agent primarily uses Dark Pixel Ratio (DPR) and Intersection Pixel ratio (IPR) as baseline metrics to find test solutions, as described for 'agent 3' in the paper: *Using Human Computation to Acquire Novel Methods for Addressing Visual Analogy Problems on Intelligence Tests* (Joyner et al., 2015).

The AI agent also implements other pixel comparison techniques to improve the confidence to select correct solutions – these techniques include the following:

- Symmetry patterns in frames along vertical, horizontal, or diagonal axis.
- Relationships of addition or subtraction for frames in the same rows and columns, e.g., 'A' + 'B' = 'C' or 'A' - 'B' = 'C', respectively.
- Representation of problems using Total Pixels Matrix (TPM) to leverage other techniques, such as the Pearson correlation coefficient (PCC) and 'learning by recording cases.'
- Combination of all the frames in the same row or column using the Sum of Total Pixels Matrix (STPM) to identify when overlying the frames in the rows and columns coincide, e.g., is 'A' + 'B' + 'C' == 'D' + 'E' + 'F'?

After multiple *Gradescope* submissions to improve the agent performance, the KBAI agent solves 86/96 RPM tests correctly for the combination of 'Basic' and 'Test' sets, as shown in Figure 1. This is equivalent to a success rate of 89.58%.

```
Message

Thank you for submission 20!


Software:

        Python version: 3.8.0
        NumPy version: 1.19.5
        Pillow version: 8.1.0
        OpenCV version: 4.2.0


Results:

        Basic passed: 48, failed: 0
        Test passed: 38, failed: 10
        Challenge passed: 22, failed: 26
        Raven's passed: 28, failed: 20
        Runtime: 11.898631811141968 seconds
```

Figure 1 — Submission results in Gradescope

2 IMPLEMENTATION

This section presents a detailed description of the KBAI agent implementation to solve RPM tests. This report follows the same name convention as several papers in the reference section – individual squares in a RPM test are referred to as 'frames', and shapes within each frame are named 'objects'.

The knowledge-based artificial intelligence (KBAI) agent mimics human intelligence to solve problems – the cognitive system that describes both of them functions as follows: the agent 1) receives an input; 2) interconnects metacognition and reaction with a deliberation process, which leverages reasoning, learning, and memory, and finally, 3) it returns an output.

In a high-level description for the KBAI agent implementation that solves RPM tests, 1) the input consists of a set of Figures (or Frames) constituting the RPM problem and a group of potential solutions that are read using the PIL Python library, 2) the input is then converted to pixel arrays of ones and zeros, mapped to multiple strategies to be implemented, and after this processing, 3) the best solution frame is selected for the output. The implementation described in this section is focused on the solution strategies for the agent.

Firstly, in order to optimize memory use in image processing, the figures are converted to 1-bit grayscale images using the function `.convert('1')` in the PIL library. After the conversion, NumPy binary arrays save the pixel information for each figure – 1 for white, 0 for black. These arrays are the basis of the comparison between frames.

Several pixel comparison techniques are implemented with the NumPy binary arrays: Dark Pixel Ratio (DPR) and Intersection Pixel Ratio (IPR), symmetry relationships, addition and subtraction relationships, Total Pixel Matrix (TPM) representations, and Sum of Total Pixel Matrix (STPM) representation. The implementation of these metrics is described below:

2.1 Dark Pixel Ratio (DPR) and Intersection Pixel Ratio (IPR)

The implementation of pixel ratios is the main strategy of this KBAI agent due to its effectiveness– it solves approximately half the tests by performing multiple pixel comparisons between two frames. Moreover, this method is simple because it is not required to recognize shapes in objects nor find relationships in more than two frames. The Dark Pixel Ratio (DPR) is defined as the difference of number of black pixels between frames over the total pixel number, whereas the Intersection Pixel Ratio (IPR) is the difference of numbers of black pixels at the same coordinates between frames[1].

Figures 4 and 5 in the Appendix illustrate the basic implementation of direct pixel comparisons using DPR and IPR in 3x3 RPM tests. By inspection, this methodology can be reduced for 2x2 RPM tests, which are simpler problems. Taking as an example the "horizontal relationship 1", it is considered that the following pairs of direct pixel comparisons could lead to similar values: A & C, D & F, and G & #, where # is the frame number that leads to the correct answer (in this case, 4). For the first two pairs, the DPR and IPR are computed; then, the pixel ratios are calculated for G & #, i.e. G & 1, G & 2, G & 3, etc. Once all the pixel ratios are computed, the KBAI agent calculates how good a match is for the calculated values of DPR and IPR by using the following formulas:

$$DPR\ Match = \sqrt{|(DPR\ of\ Frame\ with\ letter\ and\ \#) - (Goal\ DPR)|}$$

$$IPR\ Match = \sqrt{|(IPR\ of\ Frame\ with\ letter\ and\ \#) - (Goal\ IPR)|}$$

For "horizontal relationship 1", there are 4 comparisons – there is a potential solution for each DPR and IPR, and for each frame combination A & C, and D & F. Similarly, there are 4 potential solutions for "horizontal relationship 2", 4 for "vertical relationship 1", 4 for "vertical relationship 2", and 10 for the "diagonal relationships". Thus, the selection using pixel ratios is based on a pool of multiple potential solutions after all the 26 comparisons.

2.2 Symmetry Relationships

The KBAI agent identifies when there is strong evidence of symmetry (or reflection) about axes. The following conditions apply for 3x3 RPM tests only:

- Symmetry about vertical axis: When 'A' is the reflection of 'C', and 'D' is the reflection of 'F'; then # is likely to be the reflection of 'G'.
- Symmetry about horizontal axis: When 'A' is the reflection of 'G', and 'B' is the reflection of 'H'; then # is likely to be the reflection of 'C'.
- Symmetry about diagonal axis: When 'C' is the reflection of 'G'; then # is likely to be the reflection of 'A'.

These symmetries are found with the NumPy functions `fliplr()` and `flipud()`.

2.3 Addition and Subtraction Relationships

This technique is only applicable to 3x3 PRM tests. This method recognizes when any of the following conditions are met:

- Horizontal addition: 'A'+'B'='C' &'D+'E'='F'. Thus, it's likely that 'G'+'H'="#"
- Vertical addition: 'A'+'D'='G' & 'B'+'E'='H'. Thus, it's likely that 'C'+'F'="#"
- Horizontal subtraction: 'A'-'B'='C' &'D-'E'='F'. Thus, it's likely that 'G'-'H'="#"
- Vertical subtraction: 'A'-'D'='G' & 'B'-'E'='H'. Thus, it's likely that 'C'-'F'="#"

The implementation is straightforward – it is either the sum or subtraction of black pixels between the first two frames, and then a comparison of the result and the last frame.

2.4 Total Pixel Matrix (TPM)

This method reduced the PRM tests to either 2x2 or 3x3 NumPy arrays depending on the Problem Type. Each element of the array contains the sum of black pixels. For instance, the Basic Problem C-03 is represented as follows:

```
np.array([[330, 658, 996], [ 658, 1284, 1975], [981, 1985, NaN]])
```

This representation is effective to solve RPM tests for two reasons:

- It is the main input to identify highly correlated rows or columns by using the Pearson correlation coefficient (PCC). This is only applicable to 3x3 tests.

It is derived using the NumPy function `corrcoef()`. To select a frame using this criterion, all the absolute value of the coefficients must exceed 0.999.

- This number identifies challenging solutions that the KBAI agent can't solve by using 'learning by recording cases.'

2.5 Sum of Total Pixel Matrix (STPM)

This method is applied to both 2x2 and 3x3 RPM tests. The KBAI agent identifies patterns when all the black pixels in either columns or rows are overlaid. For example, the following relationships can be drawn using 3x3 matrices:

- Row combination: when 'A' + 'B' + 'C' = 'D' + 'E' + 'F', then 'G' + 'H' + '#' is likely to have the same black pixel pattern.
- Column combination: when 'A' + 'D' + 'G' = 'B' + 'E' + 'H', then 'C' + 'F' + '#' is likely to have the same black pixel pattern.

2.6 Combination of Pixel Comparison Techniques

The different techniques are combined by the KBAI agent in a hierarchical manner, in the following order:

1. If problem matches TPM using 'learning by recording cases,' select solution.
2. If the problem is symmetric horizontally, vertically, and diagonally, select solution that keeps this symmetry.
3. If the problem has an addition or subtraction pattern, select solution that keeps this relationship.
4. If the problem has a potential solution that has a high PCC using the TPM for frames in the same column or the same row, then select this solution.
5. If overlying the dark pixels for rows and columns using STPM leads to the same result, select solution that keeps this relationship.
6. If none of these methods are conclusive, use DPR and IPR to find frame with the closest match by using the formulas shown in Section 2.1.

This order is followed for each one of the 3x3 RPM tests. Items 3, 4, and 5 are skipped for the 2x2 solutions, as a relationship between 3 frames is required.

3 AGENT PERFORMANCE

The agent has an acceptable performance, as it successfully solves 86 out of 96 problems from the 'Basic' and 'Test' RPM sets in *Gradescope* (Refer to Figure 1).

A summary of the agent's performance on all the problems is presented in Table 1. Even though the score for this final project is high, the KBAI agent still has significant room for improvement for more challenging conditions, as shown in the results for the 'Challenge' and 'Raven's' sets, where only approximately half of the problems are solved correctly.

Table 1 — Summary of agent's performance to solve RPM tests

Problem Type	**Set B**	**Set C**	**Set D**	**Set E**	**Total**
Basic	12 / 12	12 / 12	12 / 12	12 / 12	48 / 48
Challenge	6 / 12	8 / 12	5 / 12	3 / 12	22 / 48
Raven's	9 / 12	8 / 12	6 / 12	5 / 12	28 / 48
Test	11 / 12	10 / 12	11 / 12	6 / 12	38 / 48

The KBAI agent's efficiency is adequate – the processing to 1-bit grayscale figures simplifies the operations with frames. However, it is acknowledged that the algorithm could be faster. The code contains a nested loop to calculate the multiple strategies. Thus, it is expected to have a quadratic order big O, i.e., $O(n^2)$. There are other single 'for' loops that do not affect the order of the big O. As shown in Figure 1, the agent ran the 192 RPM tests in 11.8986 seconds using the *Gradescope* environment, which is an acceptable running time.

4 AGENT SUCCESSES

Generally, the agent is effective at problems that pixel comparisons lead to a correct solution. Figure 2 illustrates four problems that the agent solves successfully, in accordance with the implementation described in Section 2. Each problem solution is explained in more detail in this Section.

For the Basic Problem C-04 (upper left corner in Figure 2), the agent identifies there is a strong correlation by identifying $|PCC|>0.999$ when comparing the frames' total pixels in the first two rows and first two columns (See Section 2.4) – there is always approximately double the initial circles (and pixels) when looking at adjacent frame to the right or down. Out of all the potential solutions, #8 has the number of black pixels (or TPM) that continues this trend. Per the hierarchy shown in Section 2.6, this solution is found in item number 4.

For the Basic Problem C-06 (upper right corner in Figure 2), none of the specific methods finds the solution, and the baseline comparisons using DPR and IPR is triggered to select the correct answer (See Section 2.1 and Section 2.6, item 6). For this condition, the frame that has the DPR and IPR best match was #7, which is expected to have an increasing number of black pixels and intercept all the black pixels. It is worth nothing that the PCC correlation used in the previous problem was not conclusive for this solution due to the specific pixel count.

The solution for the Basic Problem D-02 was easily identified when the agent found a good match of the final frame when combining the black pixels of the same rows and columns (lower left corner in Figure 2). The only object that could lead to the same solution is the circle in Frame #1. Per the hierarchy shown in Section 2.6, this solution is found in item number 5.

The agent identifies strong symmetry patterns for Basic Problem C-07 (lower right corner in Figure 2). Frames are the mirror image of each other along the horizontal, vertical, and diagonal axes. Frame '2' is the only one that continues this trend – it is the reflection of 'A', C' and 'G'. Per the hierarchy shown in Section 2.6, this solution is found in item number 2.

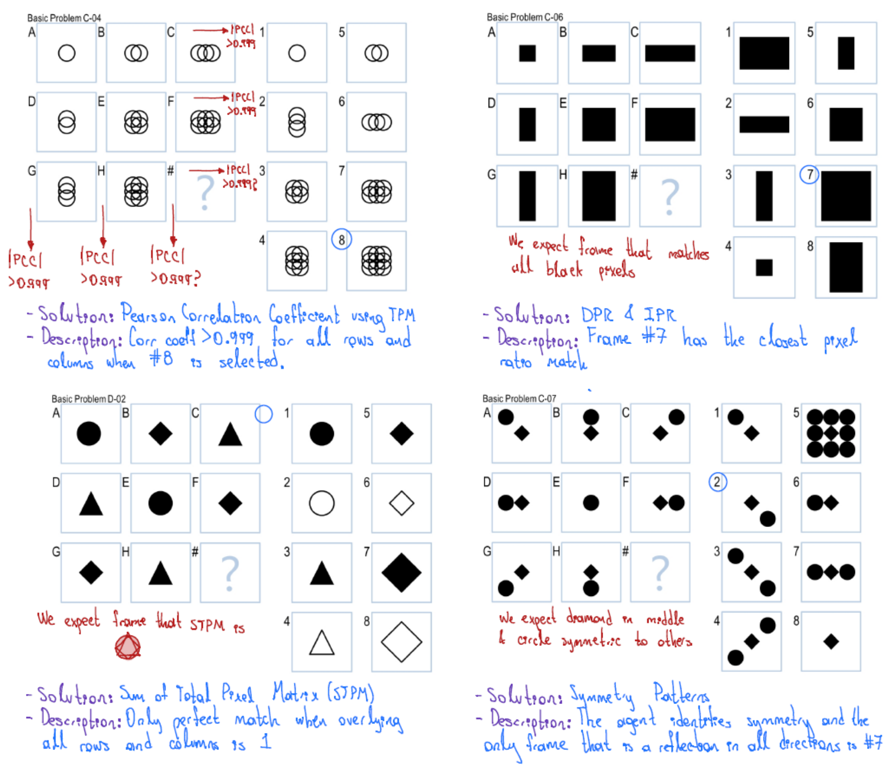

Figure 2 — Examples of agent success cases

5 AGENT STRUGGLES

The main struggle of the KBAI agent is its inability to detect specific objects, as the solution heavily relies on pixel comparison. The problems in Figure 3 evidence this limitation. ‘Learning by recording cases’ was used for the known problems, but the same struggles were expected to be present in the ‘Test’ set.

For the Basic Problem C-03 (upper left corner in Figure 3), the agent has an excellent correlation of $|PCC|>0.999$ when using the total pixel count in the TPM. However, the black pixel count is small and it selects #4 and #6 as equally likely solutions – the current agent narrowed down the solution from 8 to 2 frames.

Another struggle is the loss of information when overlying black pixels in the same row or column. Basic Problem D-06 (upper right corner in Figure 3) show an example of this limitation. Even though the agent knows that overlying the frames in the columns would lead to 3 exterior squares and a combination of a cross, circle, and triangle in the middle, the agent cannot choose between frames #1, #7, and #8 as it leads to the same combined frame.

The agent cannot even narrow down the solution to a significantly smaller set of frames for Basic Problem D-07 (lower left corner in Figure 3). The agent struggles to identify that the solution because there is no clear pixel comparison pattern between frames. Moreover, it fails to recognize that the solution must have 4 dots around the figure.

Another problem that requires shape recognition is Basic Problem E-12 (lower right corner in Figure 3). The KBAI agent does not find any good correlation using all the umbrella of pixel comparison techniques that is described in Section 2.6. Thus, it performs a DPR and IPR comparison that leads to an incorrect answer. The agent would work if it could recognize shapes of a triangle representing +1, and an inverted triangle representing -1.

6 AGENT DEVELOPMENT

The agent is better described as a suite of pixel comparison heuristics, from which a technique is selected to find the solution when there is an excellent match. For example, when there is evidence of symmetry along the horizontal, vertical, and diagonal axes, a heuristic is selected to find the frame that “makes sense” to keep the strong trends in the last row, column, and diagonal.

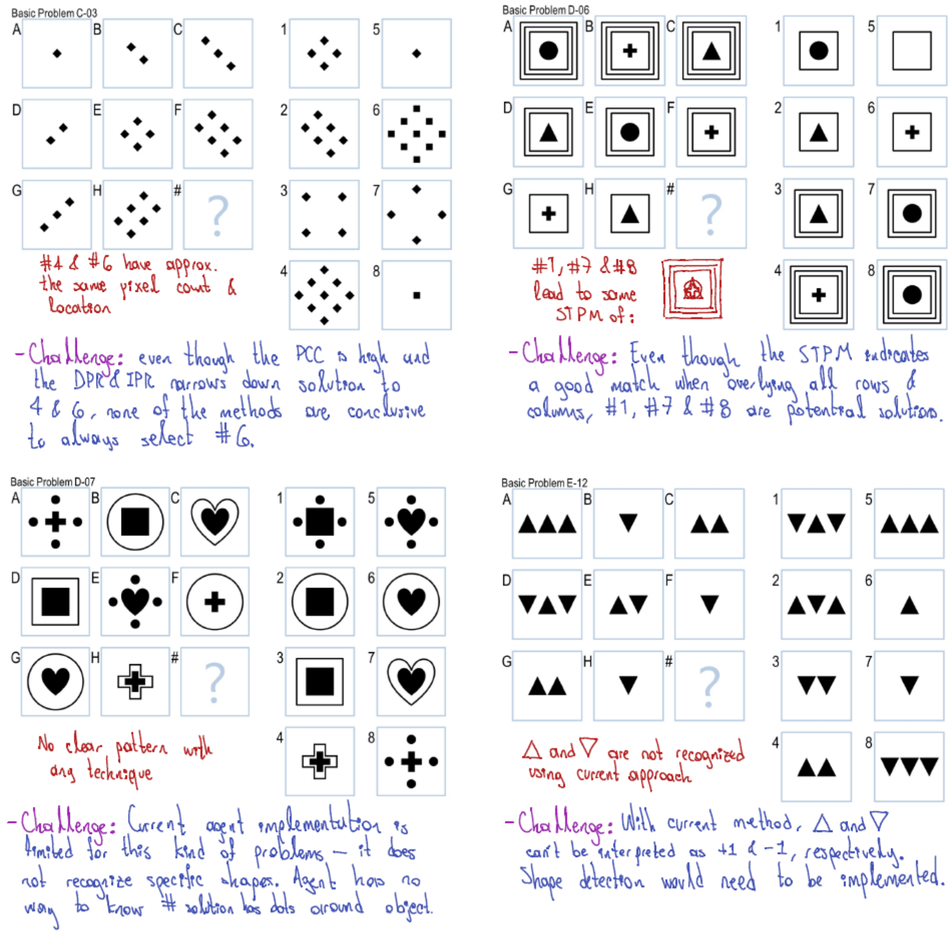

Figure 3 — Examples of agent struggle cases

The set of heuristics was gradually expanded for each project milestone. For instance, at the beginning, the agent could only solve 2x2 RPM tests using DPR and IPR comparisons; now, the final implementation has several techniques that could address specific problem types based on observable patterns.

The KBAI agent approaches each problem by trying a single technique in a specific order: 1) perfect match using 'learning by recording cases', 2) symmetry patterns, 3) addition or subtraction patterns, 4) high correlation using pixel count, 5) pixel combination in columns and rows in STPM, and 6) DPR and IPR comparison. This hierarchy described in Section 2.6. If the technique being tested does not have a good match, it saves the potential results, but moves on to the next one. When none of the techniques is conclusive, a frame is selected using a holistic evaluation of the saved results, where the methods are weighted depending on how good the match is.

7 AGENT HUMAN COMPARISON

The KBAI agent solves the RPM tests in a similar manner to a human. The way I – a (presumably) intelligent human – would solve these problems would be as follows: first, I would associate the problem to others I have seen in the past. If there is no resemblance in my memory, I would move on to find simple patterns and I would ask myself: is it the same figure? is this symmetric? what if I combine the figures, do I get the same result? is the object becoming larger? Are the objects shifting in a direction? It is possible that any of these questions could be answer with a 'Yes' and I could find the right solution. If none of these questions is conclusive, or the potential solutions are contradicting each other, then I would look at the problem more carefully to identify patterns of where the black pixels are located, how are the frames interacting with each other (horizontally, vertically, and diagonally), and how the black pixels coincide.

The KBAI agent follows the same logic as my personal approach to solve the problem. It uses 'learning by recording cases' to match a solution with a problem that was already studied in the 'Basic Set'. Then, it would find if any of the simple questions in the previous paragraph could be answered to find a straightforward solution – the KBAI agent would not move forward to explore other alternatives when there is an excellent match that could solve the problem. Finally, when there is no straightforward solution, a detailed exploration using DPR and IPR is performed to find the best match.

A major difference of my human intelligence approach and the KBAI agent is that I would leverage my ability to recognize objects to solve the problems – For example, I know what a circle, square, and triangle are, but my KBAI agent do not. Nonetheless, this approach will be explored is it is intended to be implemented for my personal learning. I am curious how far this agent can go using my implementation. Another significant difference is that it was easy for me (as human) to extend my solution to larger matrices, similarly, I could have extended my approach to different matrix shapes, additional colors, or even words. The KBAI agent cannot do this easily, nonetheless, once implemented, the agent can solve specific problems significantly faster.

When developing my KBAI agent, I always had the question in mind: "how would I solve this problem?". This is reflected in the KBAI implementation. This final project made me realize that there is no Artificial Intelligence without Human Intelligence. We need to understand how we reason, learn, and memorize to be able to implement this in our Computer Science tools.

8 REFERENCES

- Joyner, D., Bedwell, D., Graham, C., Lemmon, W., Martinez, O., & Goel, A. (2015). Using Human Computation to Acquire Novel Methods for Addressing Visual Analogy Problems on Intelligence Tests. In *Proceedings of the Sixth International Conference on Computational Creativity*. Provo, Utah.
- Goel, A. (2015). Geometry, Drawings, Visual Thinking, and Imagery: Towards a Visual Turing Test of Machine Intelligence. In *Proceedings of the 29th Association for the Advancement of Artificial Intelligence Conference Workshop on Beyond the Turing Test*. Austin, Texas.
- McGreggor, K., & Goel, A. (2014). Confident Reasoning on Raven's Progressive Matrices Tests. In *Proceedings of the 28th Association for the Advancement of Artificial Intelligence Conference*. Québec City, Québec.
- numpy.corrcoef — NumPy v1.24 Manual. (n.d.). https://numpy.org/doc/stable/reference/generated/numpy.corrcoef.html#numpy.corrcoef

9 APPENDIX

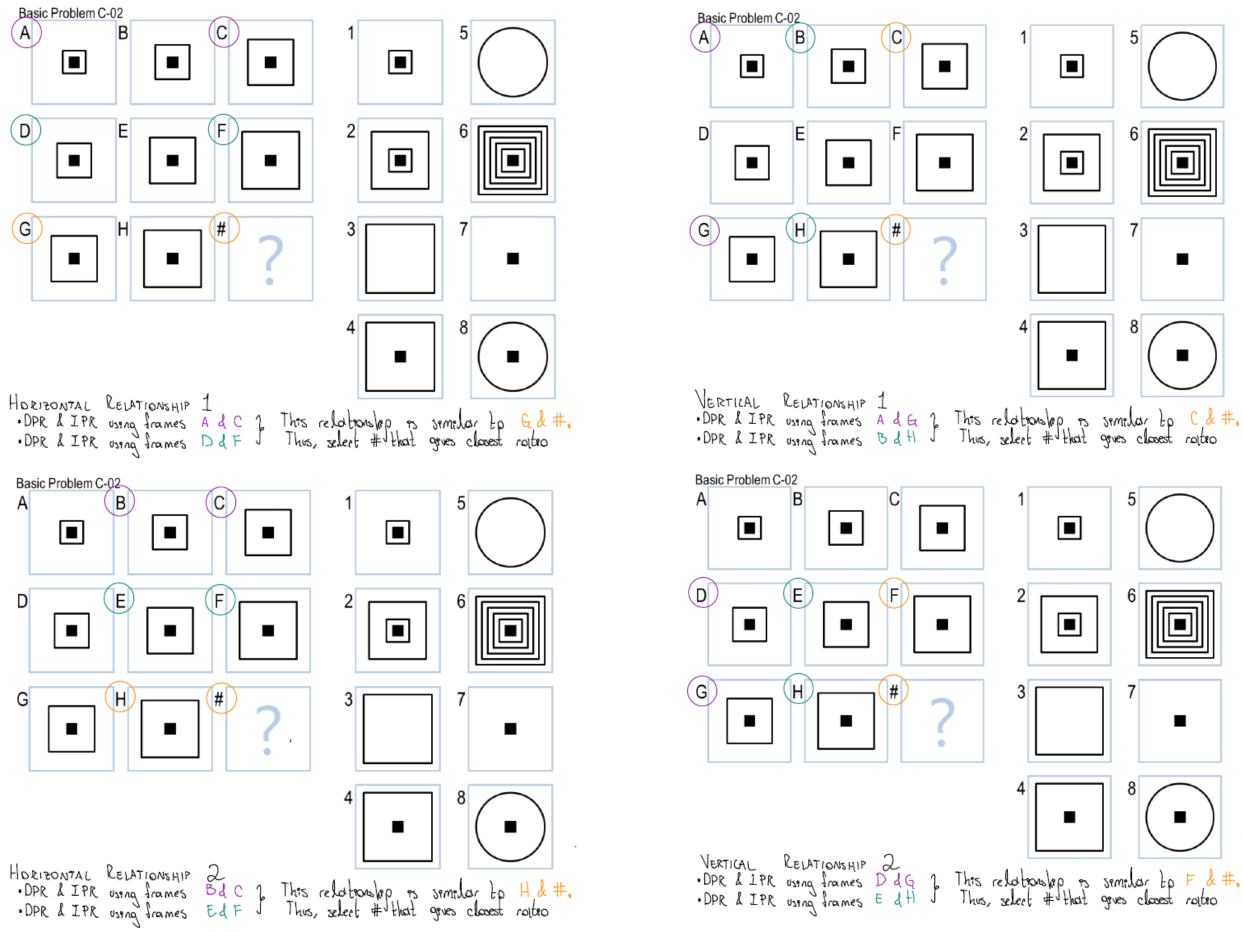

Figure 4 — Baseline horizontal and vertical comparisons using DPR and IPR for 3x3 RPM tests

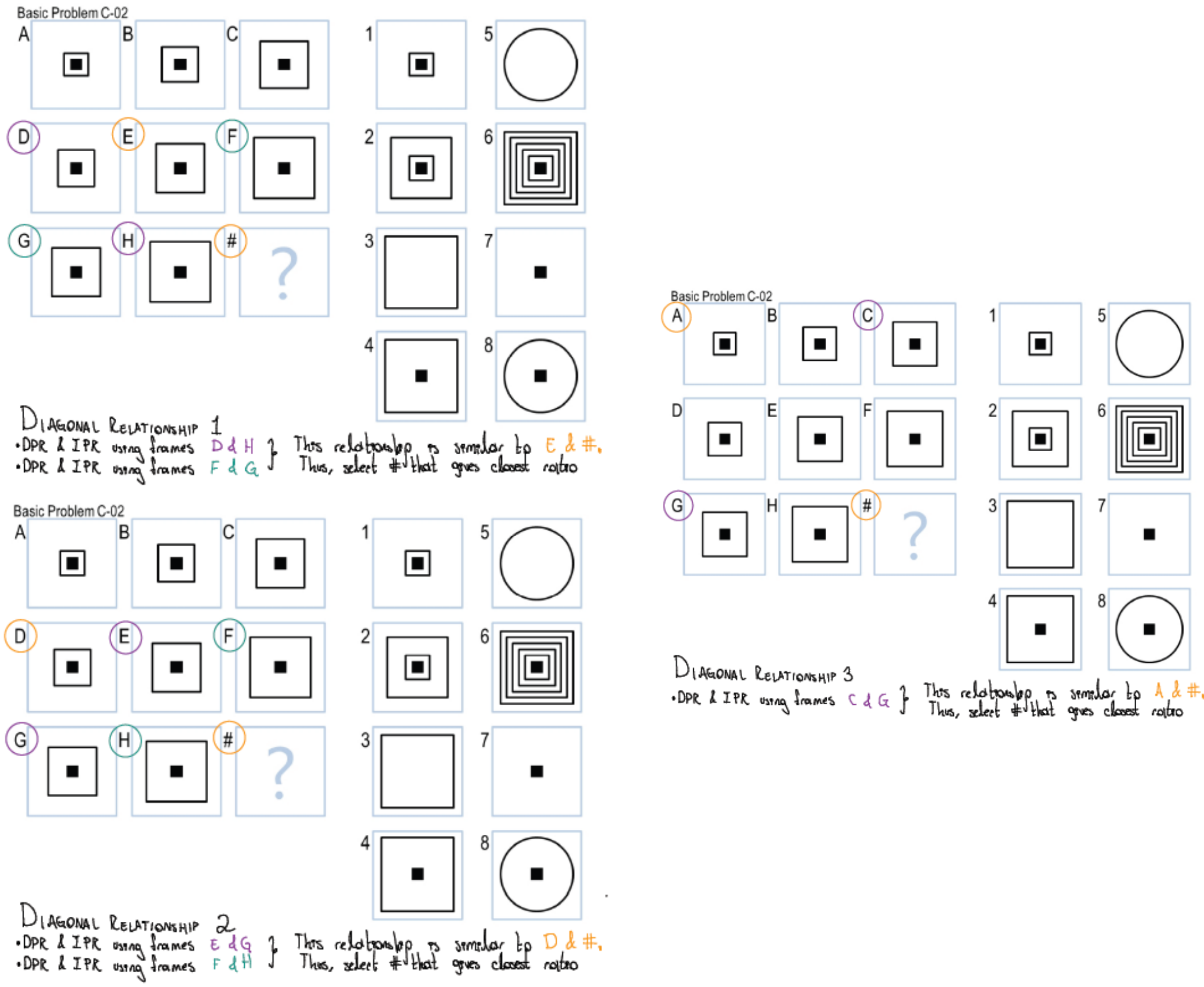

Figure 5 — Baseline diagonal comparisons using DPR and IPR for 3x3 RPM tests