RPM Final Journal

Elias Izmirlian
eizmirlian3@gatech.edu

Abstract- Raven's Progressive Matrices Test is an aptitude test where each problem uses a series of figures to imply some kind of pattern which the test taker tries to identify by picking the missing final figure. This test requires some level of visuospatial reasoning, pattern recognition, and logical problem solving from humans, and is widely regarded as a fairly effective measure of intelligence. The goal of this paper is to document a Knowledge Based AI agent designed to solve these problems and analyze what the agent's approach and performance say about its intelligence compared to a human. I opted for an approach which was meant to mimic human visuospatial reasoning to produce an agent more closely resembling human intelligence but ended up being limited by the pitfalls of programmed shape recognition.

1 MY AGENT'S APPROACH

My agent functions by using two generate and test techniques by generating a possible state where each answer choice is the solution and testing the state by calculating a score for how well this solution fits in with the rest of the problem. The two aforementioned techniques focus on different aspects of the problem. One will be hereon referred to as Shape Analysis, and the other will be referred to as Bit Analysis.

1.1 Shape analysis

Shape Analysis uses the OpenCV library in python to recognize shapes in each figure. Information about each shape is stored in a frame-based knowledge representation, with slots for center, size, number of corners, orientation, fill, and related shapes. This is done for the shapes in each figure, and the shape frames are stored in a frame-based knowledge representation for the Figure. Then, the agent compares any two adjacent figures in the problem by comparing the shapes in the of the two figures. A shape from the first Figure is assigned to a single shape in the second Figure, or if no match can be found it is assumed to have been deleted. The shapes which are picked to be related across figures are determined by a heuristic based on the similarities in the two shape's information or features. The more similar two shapes are, the more likely they are to be related.

Once related shapes are decided, the features or information which differs between two related shapes are used to infer what kind of transformations are occurring between the two. In previous journals I described my method of inferring transformations as a Thematic Role System since the kinds of information stored about the transformation and the shapes change based on what transformation it is, but since then I've modified the reasoning to be more of a Diagnosis. The features or slots which change are the signals, and the available transformations are the possible diagnoses. This portion of the agent aims to account for the changes in shapes with as few as transformations as possible, so I believe using a Diagnosis approach makes more sense. Once a list of transformations (and quantities used to quantify how much the shape was transformed) is produced, it is stored in a larger list of transformations occurring the problem (transformations between vertically adjacent figures are stored separate from transformations between horizontally adjacent figures). The agent then compares the number of shapes in each figure to try to identify any multiplicity-based transformations.

Then comes the Generate and Test portion of this sub-approach: the agent generates a state by substituting an answer choice in for the missing final figure, and then repeats the above process to come up with transformations occurring between this answer choice and the figures directly preceding it. A score is calculated for this state based on how many transformations found in the problem's figures are also found when this answer choice is substituted in. This is done for each answer choice, and the answer choice with the highest score is selected as the answer.

1.2 Bit analysis

Bit Analysis is a lower level of abstraction problem solving technique the agent employs to address certain transformations in 3x3 RPM problems which the above method could not detect. Bit Analysis takes whole rows into account at once in order to detect binary operations like AND, OR, XOR, etc.This problem-solving technique is used first, and if no answer is returned, then the shape analysis is run. It compresses the figure images into binary black and white images, and then tests the binary operations in question on the first two images of a row and compares the output with the third image. It does this for each row. If there is a match (so the operation produces an image similar to the final image in that row) then each answer choice is compared to the output from this operation on the two images of the final row. The full list of operations the agent checks is AND, OR, XOR, subtraction (so figure 2 minus figure 1) and area addition. The similarity between the output of an operation and an answer choice or figure is determined by counting the number of non-zero pixels in the difference between these two images (so figure minus output and output minus figure). The answer choice with the lowest difference (below a threshold) is chosen as an answer. If no answer choice is similar enough to the output, even if there is a match for that operation, then no answer is returned. If none of the operations produce a match or an answer choice, then no answer is returned from Bit Analysis so Shape Analysis is run.

2 PERFORMANCE

2.1 Overview

I would define performance with respect to AI as the ability of an agent to accurately complete the task it was designed for in a variety of different problem scenarios. With respect to this definition, my agent performs decently, but not incredibly. Below I have included a table of the performance of the agent for each problem category:

Table 1 — Reports the accuracy of the agent in solving RPM problems for each of the listed Problem Sets

Problem Set	Correct	Incorrect
Basic B	7	5
Basic C	7	5
Basic D	4	8
Basic E	8	4

Problem Set	Correct	Incorrect
Challenge B	3	9
Challenge C	4	8
Challenge D	1	11
Challenge E	3	9

The above table demonstrates that my agent performs pretty well with problem sets B, C, and E, but not very well with problem set D. It performs better on the basic problems than the challenge problems, but the same trend is still noticeable there. Due to my agent failing to perform well in all problem scenarios, I would describe it as adequate, or promising if certain improvements are made.

2.2 Problems with good performance

My agent performed well with problems that had solutions which were easily recognized by the bit analysis – that is to say problems with these binary operations were almost always answered correctly. For example, below I have included one such problem:

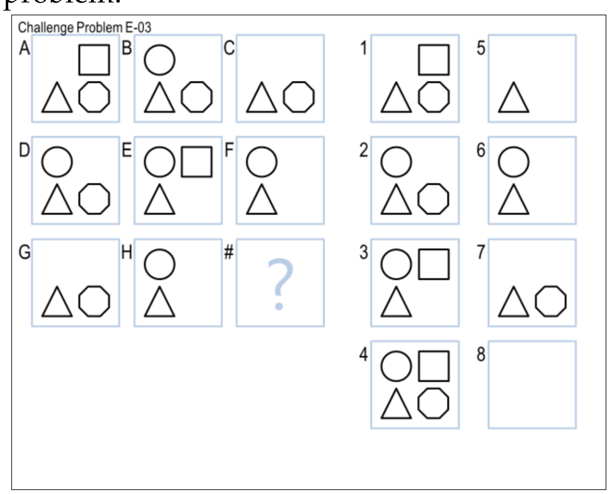

Figure 1 — Challenge Problem E-03 where the first and second elements of a row have multiple shapes, and the third is the AND of the first two

My agent would run Bit Analysis on this problem first, taking the binary pixel data of the first two images in each of the first two rows, performing a binary operation on them, then comparing the result with the final figure in the row. If they match up, meaning the difference is within a given threshold, then the agent starts comparing the result for the final row with answer choices. The only binary operation which would have matches for the first two rows is AND. The agent would then AND figure G and figure H to create an expected final figure. This expected final figure has only a triangle in its lower left corner. The agent then calculated the difference (in pixel data) between this expected image and the answer choices. The answer choice with the smallest difference, 5, is returned from the bit analysis method with the difference, which in this case is very low. Since the difference is less than the threshold, and is the lowest difference among the operations tested, 5 is returned as the answer, which is correct.

The agent performs very well with these types of problems because the possible operations are easily checked and verified with the answer choices. The agent's problem solving for these kinds of problems also requires no shape recognition, which is good as shape recognition is often a source of uncertainty and risk.

Another type of problem my agent performs well with is problems with shapes and transformations which are easily detectable. The methods my agent uses from the OpenCV library are the best at detecting shapes with solid black fill, and thus these are the shapes my agent is the best at identifying. Having accurate information about the shape makes the agent significantly better at diagnosing what kinds of transformations are occurring. I have included such a problem as an example in Figure 2.

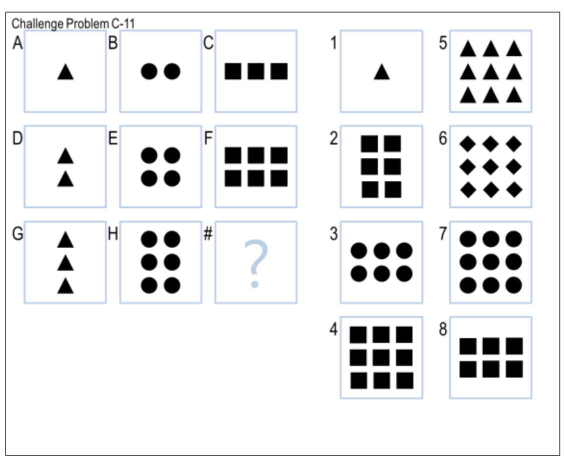

Figure 2 — Challenge Problem C-11 where the figures contain increasing amounts of triangles, circles, and squares.

For figure 2, my agent would first run Bit Analysis, but none of the operations it would try would produce compelling results (area addition would be the closest, but the differences in shapes keep it from being used here), so then it would run Shape Analysis. The agent would be able to accurately identify the shapes in each figure as they are nicely spaced apart and are high contrasting with the background. The agent would then run a "Multiplicity Analysis" function which just compares the change in number of shapes per figure. This method identifies consistent increments in the rows and columns of the problem, so the agent awards points to the scores of answer choices which maintain these increments (so answer choices 4,5,6, 7). The agent then compares shapes between figures and identifies transformations. The shapes between two figures are pretty easy to match up based on location and/or shape. It recognizes that some shape transformation is going on in figures that are horizontally adjacent, and that shapes stay the same vertically. Each answer choice is then tested, with the most points being awarded to the answer choices which involve shape transformation from Figure H and keep shapes the same from figure F. In other words, answer choices with squares are awarded the most points, so answer choices 2, 4, and 8. Thus, 4 has the most points, and is selected as the answer, which is correct.

My agent performs well on problems like these as there are multiple opportunities for the correct answer choice to get additional points which would make it the most favored answer choice. When the shapes are easily separable and identified, the analysis on them is much simpler for the agent, and the agent is able to make more accurate inferences, which contributes to the correct answer choice being the one to get the most points.

A third type of problem my agent performs well on is a bit more unexpected: problems which involve operations with the number of sides. The reason for this was unintentional – the shape recognition methods my agent uses are not very good at detecting shapes that have a white fill matching the background, as the library identifies each side as a very thin polygon. Normally this is an issue, but with problems that benefit from looking at the sides of the shape separately, this is actually a big strength. I have included such a problem below in Figure 3.

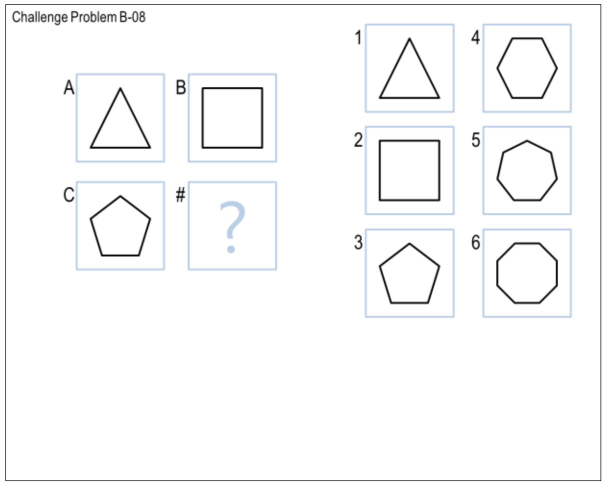

Figure 3 — Challenge Problem B-8 where the figures contain a single shape with an increasing number of sides

For this problem, my agent would run Bit Analysis but not get any conclusive results, then run Shape Analysis and identify each side as a shape as I mentioned above. Then, the agent would run the multiplicity analysis method on these figure frames and notice that there is an increment of 1 between horizontally adjacent figures and an increment of 2 between vertically adjacent figures in the number of shapes (or sides) the figure has, so it would award points to the answer choice which maintains this increment – 4, as answer choice 4 has 6 sides/shapes. Then, the agent would try to establish transformations between the shapes but would not be able to get anything particularly useful, except for maybe some translation of the shapes/sides. Thus, answer choices would be awarded minimal points otherwise, so 4 would be selected as the answer.

My agent performs well on problems like these because it correctly analyzes the sides individually; however, for a lot of other problems this is a weakness.

2.3 Problems with poor performance

As I mentioned in the section above, the methods my agent uses to recognize shapes runs into issues with shapes that have a white fill, as the agent identifies each side as an individual shape. It took me a long time to realize that was what was happening, and unfortunately was not able to come up with an effective solution to this issue in time. Since the shape recognition is such a vital component of Shape Analysis (which is used for every problem Bit Analysis is not able to solve), this causes a lot of issues in the subsequent inferences my agent makes. My agent faces this issue with complicated empty shapes and concentric shapes, which there are a lot of in the Basic Problem D set, explaining its poor performance in that category. I have included such a problem which my agent struggles with in Figure 4 (not from Problem Set D though).

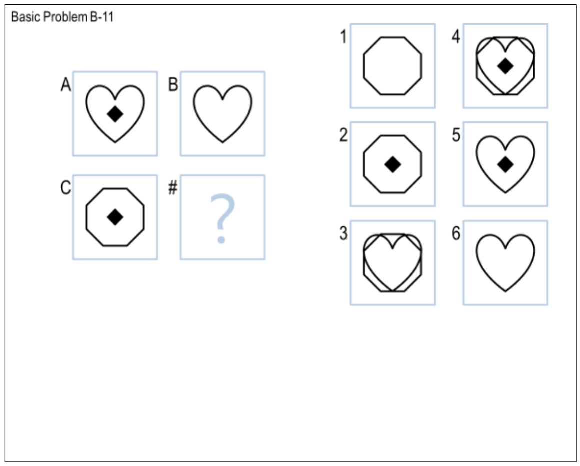

Figure 4 — Basic Problem B-11 where the figures contain an empty heart or octagon with or without a black diamond in the middle.

In this problem, my agent recognizes the black diamonds easily, and is able to infer that the black diamond is deleted from figure A to figure B, but kept from figure A to figure C, so it awards points to answer choices with no black diamond in the middle – answer choices 1, 3, and 6. However, the agent has a much tougher time recognizing the empty hearts and octagons, and instead interprets it as a bunch of small shapes. Thus, the agent is unable to infer the shape transformation occurring vertically, and instead awards points to answer choices which keep the same tiny shapes it picked up on. In other words, the agent sees that the black diamond stays the same, so it looks for answer choices with this "Same" transformation in relation to the heart it misidentified as an amalgamation of smaller shapes. Thus, it awards points to answer choice 6, selecting it as the answer instead of the correct answer 1.

Another kind of problem my agent struggles with is problems that involve recognizing the attributes present in rows and columns and maintaining these attributes. The reason is that I did not encode any human knowledge dealing with this to my agent. I failed to equip my agent with any reasoning that could handle these sudoku-like problems which require the agent to consider the features of a row or column as a whole outside the context of binary operations. One such problem is shown below in Figure 5.

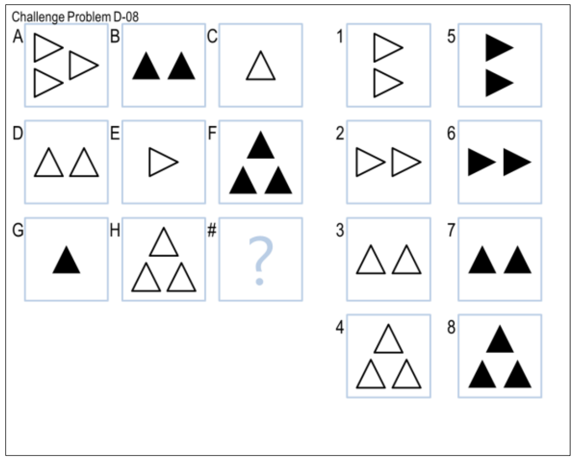

***Figure* 5** — Challenge Problem D-08 where the figures contain 3,2, or 1 triangle(s) which are tilted, empty, or filled.

For this problem, my agent tries Bit Analysis, but is unable to come up with any binary operations which create matches. It then tries Shape Analysis, but the transformations vary so much from shape to shape that no single answer choice is awarded many points. It tries multiplicity analysis as well, but the increment is not consistent, so this fails as well. The agent ends up selecting 5 as the answer since it involves rotation from figure F, which is a transformation it recognized. This is similar to answer choice 1, which is the correct answer, but because the agent is unable to consider attributes of the row or column as a whole, it is unable to effectively solve this problem.

3 OVERALL APPROACH AND HUMAN COMPARISON

3.1 Overall approach

Though I have a couple of approaches I use in the agent, my goal was to mimic human reasoning in the agent, so I focused the most on the Shape Analysis component. However, due to the issues I faced with shape recognition, I eventually added Bit Analysis as an additional support to Shape Analysis. The number of heuristics I employed increased somewhat over time, but the score for the answer choices was always the primary heuristic. Thus, I would say that the tools my agent employed increased over time, but the main tool was always meant to be

Shape Analysis as it most closely resembles human reasoning for shapes like these.

3.2 Human comparison

As I mentioned previously, I believe that the Shape Analysis component of my agent's reasoning is very similar to human reasoning. I believe we humans also recognize shapes, establish transformations, and compare this with answer choices. This human reasoning was the guiding methodology when I was designing this component. Unlike humans, however, my agent is much more methodical and robotic in its problem-solving strategy. For instance, a human might imagine what they expect the answer to look like for a problem and then look for this in the answer choices. This creationary aspect of making an expected answer is something the Bit Analysis has (in a way) but the Shape Analysis does not.

While I would say the concepts my agent uses to guide its answer selections are pretty similar to human visuospatial reasoning, its inconsistent shape recognition is a major difference. Another large difference is that my agent is very rigid in its problem solving: if the problem is outside the capabilities of the methods of analysis I've programmed, the agent often fails to come up with a clear answer, and the answer choice selected is picked with a very slim margin. While humans might also be unable to decide between multiple answer choices, they have more flexibility and adaptability in their reasoning which my agent is lacking. One major reason for this is I while I put in a limited conceptual knowledge base on certain kinds of transformations, I did not implement any conceptual learning such as learning by correcting mistakes or incremental concept learning through version spaces. If my agent had a growing knowledge base of transformations or problem-solving strategies, I think it would have the flexibility and adaptability it is lacking in comparison to human reasoning and been much closer to mimicking human intelligence with respect to RPM problems