RPM Milestone 1:
CS7637

Josh Adams
jadams334@gatech.edu

1.1 How do you feel you, as a human, reason through the problems on the Raven's test?

As a human I am confident I can reason through the problems on the Raven's test. The problems I have come across deal with having multiple answers to the question, this makes the Ravens test more complicated.

1.1.1 *Raven's Problem (Low Complexity)*

Using problem 'Basic Problem B-01' to convey a low complexity example of the Ravens problem. When you compare image A and image C, it looks as though no changes occur. Looking at image B we can conclude the expected result would be image 2.

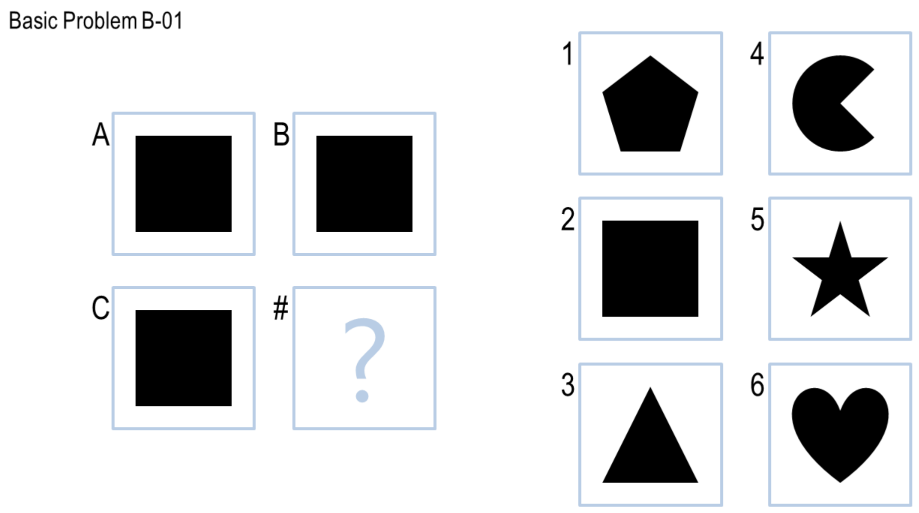

Figure 1—Basic Problem B-01, provided to CS7637.

1.1.2 *Raven's Problem (Moderate Complexity)*

For a moderate complexity problem, I chose 'Challenge Problem E-07'. For a human this does not seem to be terribly difficult.

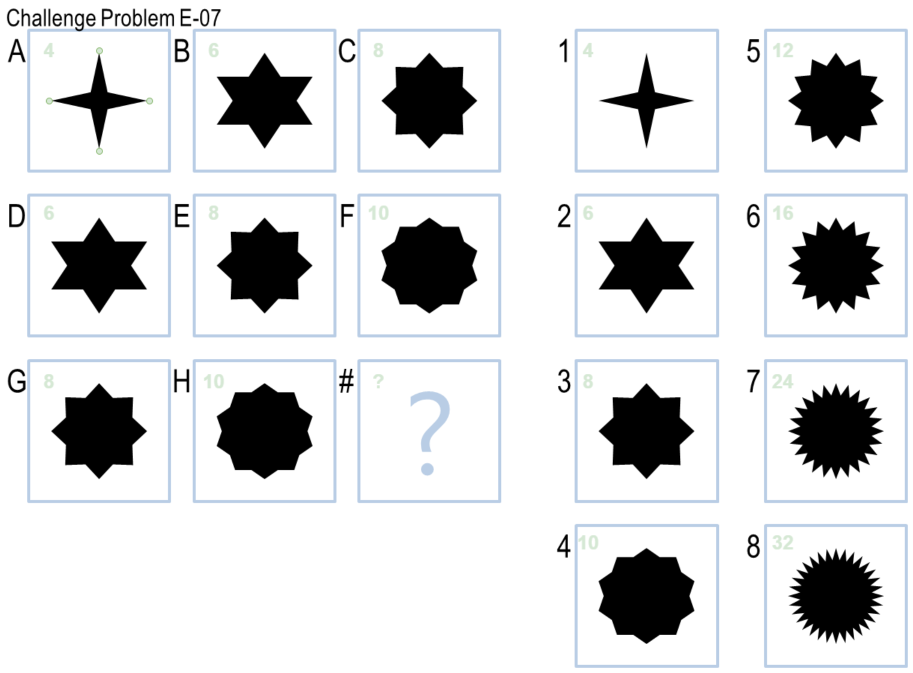

Figure 2 — Challenge Problem E-07, provided to CS7637.

I have added the number of points for the various shapes to the images above in green. Looking at image A, it has a green four, thus means it has four points associated with that image. When we do a horizontal scan and look at image A, B and C, the number of points starts at four, is incremented by two to produce six and then incremented by two to produce 8. Looking at image D, E and F, a similar pattern is found of incrementing the previous shapes points by two. A reasonable solution would be to apply the same pattern for G, H. Image H has 10 points, so it would be reasonable to expect the answer to have ten + two (twelve) points, image 5. Doing a vertical scan shows the same pattern of incrementing the previous shapes points by two. Diagonal shows a different pattern of incrementing by four, these three different scans all give the same answer of a shape with twelve points, image 5.

1.2 How do you expect you will design an agent to approach these problems?

I am not sure how I will incorporate learning because as it stands, I would not be able to submit a model and the auto-grader does not seem as though it would allow for accessing the images and learning from them. I will attempt at a later point to resolve this issue, currently my agent will employ multiple methods for discerning a possible answer. Each of these methods will be used to solve a problem (when applicable) and their results will be weighted by the confidence of that method to produce an overall answer. For all the problems I have already implemented methods to separate the problem parts. I will be able to access each individual image and process as needed.

1.2.1 *Method One (Low Complexity)*

Method one is a low complexity algorithm which can be entirely implemented using NumPy. It works on the most elementary of problems. It works differently when the problem is a 2x2 or 3x3 matrix. When the problem is 2x2, it calculates the root mean squared error between the A image and the C image. It then compares the B image with all available answers and returns the one closest to the error between images A and C. For a 3x3 matrix it does not perform well but is processed similarly to that of a 2x2 problem. Starts by calculating the error between A and B, E, D; those errors are then averaged. The next step is to calculate the error between E and H, F, and each of the possible answers. Then average those errors and find the one closest to the average error produced by A and B, E, D.

1.2.2 *Method Two (Moderate Complexity)*

Method two is of moderate complexity and can be implemented using OpenCv. This method performs better than the low complexity method on more difficult problems but has problems on very simple problems. First a feature detector is used to find features in the images A and C. Then a brute force matcher is used to link features that are shared in the two images. A transformation matrix is calculated using OpenCv findHomography and then we generated a new image using one of OpenCv transformation functions. The transformation matrix is a mathematical representation of how the image was moved from image A to image C. Then compare the newly generated image to image C and calculate the root mean squared error, this will be our new threshold for possible answers. To find the answer to take image B and go over these same steps with each of the answers and find one which produces a similar error as the A to C images. One of the problems with this method is that in very rudimentary problems, the features are not able to be detected which does not allow for further examination of the problem.

1.2.3 *Method Three (High Complexity)*

Method three is of (relatively) higher complexity and can be implemented using OpenCv. This method combines a few different sub methods such as feature detection, edge detection, object detection and pattern/template matching. To clarify feature detection may find singular points as features while object finds a collection of features. Feature detection will be used in a similar fashion as method two. Using edge detection to find corresponding edges within the images. Using the features and edges, calculate the changes applied to the images. Using object detection to find shapes, such as triangles or squares, then keeping track of the objects and determining if object is modified and how it is modified. If image A has a triangle in it and image C does not have a triangle in it then we can infer if there is a triangle in image B, then there should not be a triangle in the answer. This becomes more complicated as the shapes are varied. Combining the object detection with the template matching should allow for pattern discovery. Combining the results from these various methods should help to converge to a relatively accurate answer. These methods have defined results, meaning, no matter the question the produced answer can be determined. By the end of the course, I expect my agent will be able to learn from past experiences and make more educated predictions.

1.3 What do you anticipate your biggest challenges in designing your agent will be?

I expect the most challenging part in designing my agent will be allowing my agent to learn. Currently, I am unable to think of a method to allow my agent to learn because the correct answer is not provided. While my methods may work, given new problems, my agent would not be able to generalize well and will always perform relatively the same. My hope is to be able to provide functionality which would allow the agent to get better as more examples have been processed.

2 REFERENCES

1. All images provided by the course CS7637, per https://edstem.org/us/courses/3783/discussion/211828 we do not need to cite these.
2. https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_template_matching/py_template_matching.html