IQ Car

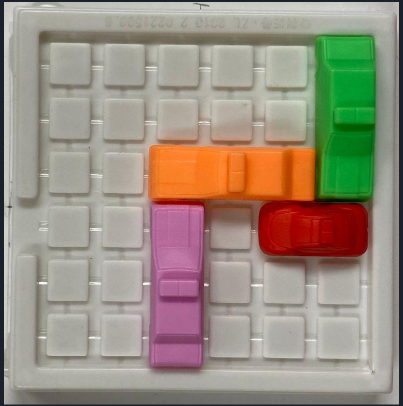
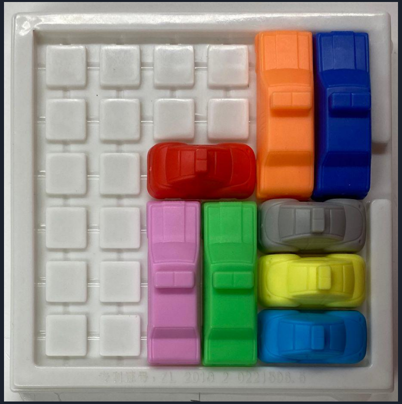
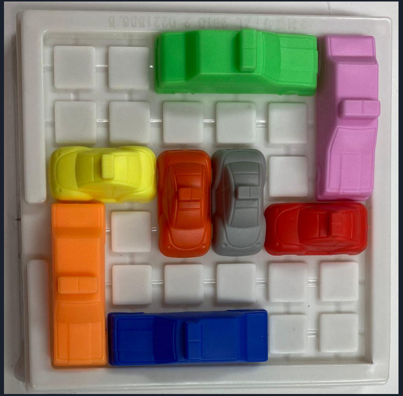
Alexander Peseckis, Casey Ford, Jack Stanek

Background and Problem Statement

What is IQ Car (A.K.A. Rush Hour)?

- N x N game board with rectangular cars, either 2 or 3 squares long
- Each car can move horizontally or vertically, but cannot turn
- Cars can block other cars
- Moving a car out of the way can block other cars further down the board
- Goal: move red car out of the board

Example game boards

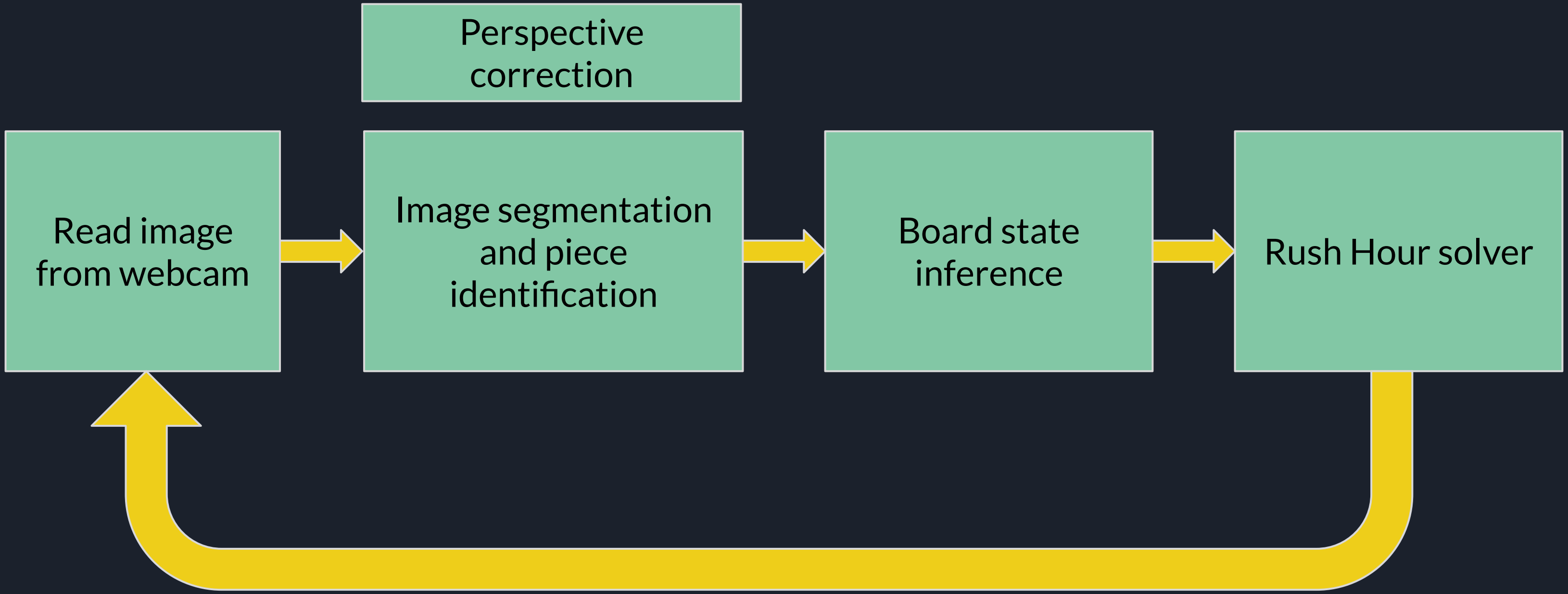




Problem Statement

- Rush Hour can be difficult and users might get stuck while playing it
- Our project allows such players to upload an image of their game board and instantly know how to solve it when they get stuck
- Given an image of a game board, interpret the state of the game and search for a solution. Display the suggested move on top of the original image.

Methodology

Image Analysis Pipeline

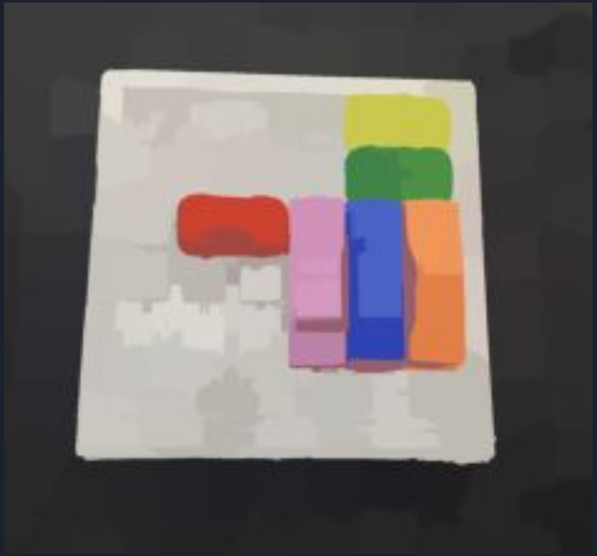


Corner Detection

- We found poor performance with standard Corner detection (lots of erroneous corners!)
- Solution: Find the Center of mass - split the board into 4 sections
- Find the min/max of the binary image in the 4 quadrants after cleaning it up with erosion/dilation.

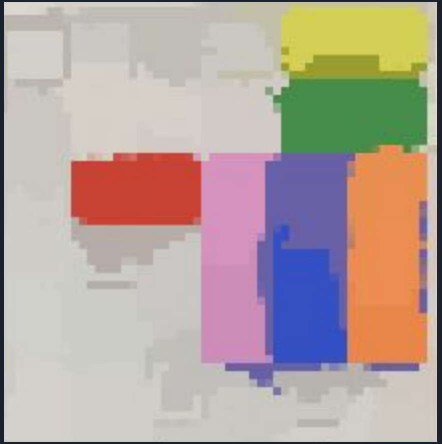
Image Segmentation

- Used SLIC for our image segmentation
- Breaks board up into colors
- Allows us to better identify the
  - Board location
  - Car locations



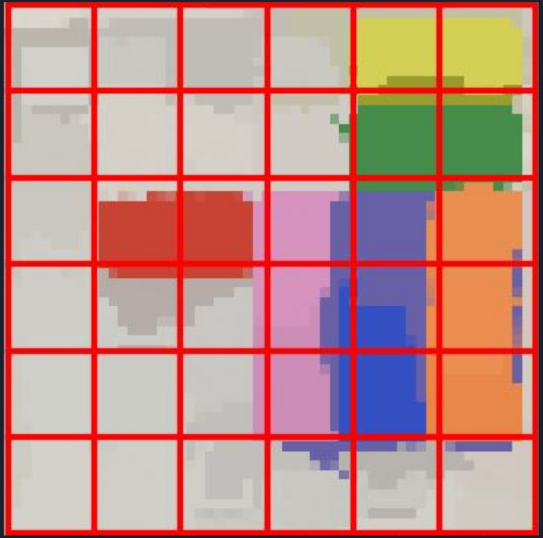
Perspective correction

- After locating the corners of the game board, compute a projective transformation to warp the board into a square shape



Board state inference

- Split the perspective corrected and segmented image into 6x6 chunks and find the modal color of each chunk
- Output is a 6x6 array of pixels
- Interpret that into a boardstate



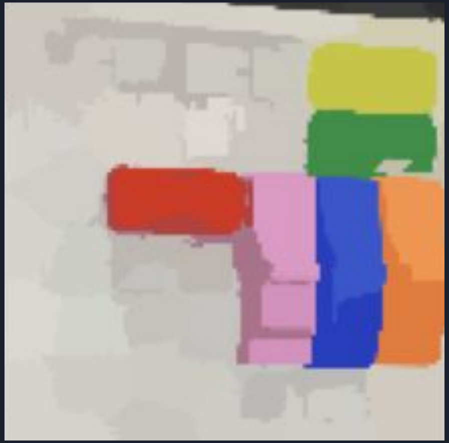
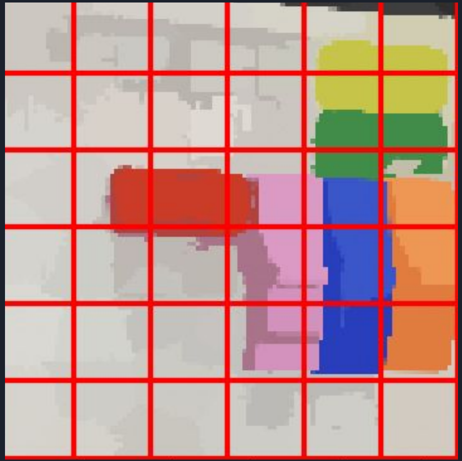
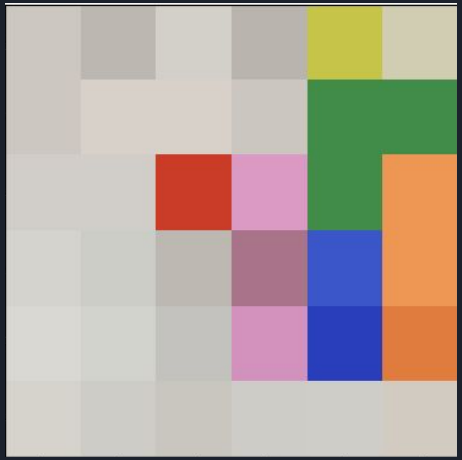
Solving

- Generalized Rush Hour is PSPACE-complete; no deterministic polynomial-time algorithm is known to solve it
- Game board is small enough that a brute-force breadth first search is practical
- For some given game state, recursively enumerate all valid single-move perturbations of the game state

Results

Complications

- Capable of coping with tilt to some extent, fast enough for real time applications, however…
- Struggles with robustness to orientation and lighting
- Making assumptions what the background and board look like and where the exit is relative in the image

Conclusions

- Determining board state from a picture is not as easy as it may sound
- Classical methods can be used but require many limitations
- Using a neural network to infer board state would likely be more robust than interpreting it by classical methods
- Simplifying the image should make training a network faster and more robust by default since it has to learn less complexity

Thank You!