A Maze Solver for Android
Rohan Paranjpe, Armon Saied
Department of Electrical Engineering, Stanford University

Motivation
- We wanted to use image processing techniques to detect and solve a maze
- Maze solving and shortest path algorithms are also prevalent within image processing (e.g. routing problems, feature extraction and labeling, topographical mapping problems)
- We restrict our problem space to simply connected mazes whose start and end points are on the edge of the maze
- Using filtering, region extraction, morphological thinning, and other techniques, we are able to obtain the solution and overlay it onto the original image

Algorithm Design
A Summary of the Methods Used
- Local Adaptive Binarization via Otsu's method for thresholding
- Median Filtering for small noise removal
- Region labeling and small region extraction
- Erosion of thin outer borders
- Morphological thinning using the Zhang-Suen algorithm and a template-based extraction
- Start/End detection using sums of weighted exponentials
- Deadend pruning
- Dilation of the solution space and overlay onto the original image

Images from Run
Original Image → Binarization and Filtering → Morphological Thinning and Junction Extraction* → Solution and Overlay onto Original Image

Conclusions
- Overall our image-processing maze-solving algorithm was successful
- Results met expectations for scope of primarily rectangular, simply connected mazes
- Future work could include:
  - optimizing algorithms and computation to allow for "real-time" solving
  - expanding scope to include more complex mazes and mazes of varying shape
  - extending project to work on topographical maps

Acknowledgements:
We would like to thank David Chen and Peter Vajda for their help and guidance throughout the quarter.