Making Real Games Virtual: Tracking Board Game Pieces

Steven Scher
sscher@soe.ucsc.edu
www.soe.ucsc.edu/~sscher

Ryan Crabb
rcrabb@soe.ucsc.edu

James Davis
davis@soe.ucsc.edu

Department of Computer Science, University of California, Santa Cruz, CA

Introduction
Many board games are played in person and online. We bring the real world into the virtual world.

A player may record an in-person game by placing their camera on the table next to the game board, taking photos of the game. We automatically find the likeliest legal move sequence.

The game transcript may be studied afterwards, shared with friends and teachers, or added to online compilations, bringing the attendant benefits of online game play to an in-person game.

Experimental Results
Single Frame Mistakes: Wrong Color, False Negatives, False Positives

Inference Over Sequence: Detection Delayed
These images represent a time sequence of moves played. The top row shows the original image of the board. The middle row shows the likelihood of detected stones. The bottom row shows the best move sequence detected by the inference algorithm. Inference qualitatively changes the kinds of errors incurred: many fewer mistakes occur, with the most common mistake being stones detected in the wrong order.

The Most Likely Path in a Hidden Markov Model is a Shortest Path

Finding The Most Likely Move Sequence In an Exponentially Large HMM Efficiently with A*

For a Game of Go
Each time step corresponds to a photo.

Possible nodes (nonzero transition probability) at time step 1 are the empty board, or a single black stone in any position.

Possible nodes at any time step are an uncharged board, or the legal addition of a single stone. All legal transitions are equally likely.

Following stone capture, the only possible transitions are removal of possible stones.

Calculating \( P_j(t) \)
The Go board is a 19x19 grid of 361 possible stone locations. Each location is either empty, or occupied by a black stone or white stone. A Support Vector Machine is trained to estimate probabilities for the three possible values, and is applied independently to each pixel in each photo.

The board state \( j \) of the whole board assigns a value (black, white, empty) to each of the 361 locations. The likelihood of a state is the product of the likelihood of each assignment.

A Good Heuristic for \( A^* \)
All graphs that calculate \( P_j(t) \) as we do admit a very good heuristic for \( A^* \). The state that assigns the most likely value to each location (independently and without regard to history or game rules) gives an upper bound on \( P_j(t) \). In practice this bound is quite tight, since the stone detector typically makes no more than a few mistakes in any single photo.

A tight bound allows \( A^* \) to trim large portions of the search tree. Since the tree is constructed on the fly, and only those nodes in the fringes are kept in memory, a tree with \( |E(t)| \) nodes may be searched with only \( 2|E(t)| \) nodes actually evaluated, and only \( |E(t)| \) in memory at once.