Small Scale Manipulation with the Calliope Robot

Owen Watson[1] and **David S. Touretzky**[2]

[1]Department of Computer Science and Engineering
University of South Florida
Tampa, Fl 33620
opw@mail.usf.edu

[2]Computer Science Department
Carnegie Mellon University
Pittsburgh, PA 15213
dst@cs.cmu.edu

Abstract

Calliope is an open source mobile robot designed in the Tekkotsu Lab at Carnegie Mellon University in collaboration with RoPro Design, Inc. The Calliope5SP model features an iRobot Create base, an ASUS netbook, a 5-degree of freedom arm with a gripper with two independently controllable fingers, and a Sony PlayStation Eye camera and Robotis AX-S1 IR rangefinder on a pan/tilt mount. We use chess as a test of Calliope's abilities. Since Calliope is a mobile platform we consider how problems in vision and localization directly impact the performance of manipulation. Calliope's arm is too short to reach across the entire chessboard. The robot must therefore navigate to a location that provides the best position to access the pieces it wants to move. The robot proved capable of performing small-scale manipulation tasks that require careful positioning.

Introduction

Chess has become a popular task for robotic manipulation (Anderson et al., 2011; Chernova et al., in press). Our solution to chess focuses on maximizing the use of a mobile platform. The Calliope robot (Touretzky et al., 2010) is a mobile manipulator using an iRobot Create base, an ASUS netbook, and a webcam on a pan/tilt mount. The Calliope5SP model includes a five degree of freedom arm with a gripper with two independently controllable fingers. Calliope's limited reach requires it to maneuver around the board to manipulate pieces.

We use the Tekkotsu open source software framework to control the Calliope (Tira-Thompson and Touretzky, 2011). Chess was previously addressed using Tekkotsu on the Chiara hexapod robot (Coens, 2010; see Figure 1), and that code is the foundation for the solution on Calliope. Calliope uses a 3D arm, while the Chiara had a simpler arm that was essentially planar, with translation along the *z*-axis provided by the legs.

The implementation of chess on the Chiara focused on manipulating pieces reachable from the robot's side of the board. Chiara was able to move laterally to manipulate pieces, but due to lack of time, Chiara was not programmed to move around the sides of the board to manipulate pieces that were beyond its forward reach. In contrast, the solution implemented on Calliope provides the functionality to manipulate pieces at any location on the board.

The Tekkotsu Framework

The Tekkotsu framework includes a set of four interacting software modules called the Crew (Touretzky and Tira-Thompson, 2010). Each Crew member performs a set of low level tasks that allow programmers to design their solutions at a higher level of abstraction.

The Lookout

The Lookout is the crew member responsible for controlling the robot's sensor package. On the Calliope, the Lookout takes pictures after moving the pan-tilt to center the camera on points of interest.

The MapBuilder

The MapBuilder is the member of the crew responsible for building symbolic representations of the world. The MapBuilder invokes the Lookout in order to receive images from the camera. For chess, the MapBuilder was used to generate representations of the state of the game.

MapBuilder representations can be described in three coordinate systems, each useful for different tasks. Allocentric representations of the world were used as a reference point for localization of the robot and the game pieces. Egocentric representations were used for proximity estimates for desired pieces and for acquiring intermediary information necessary for localization. Camera-centric representations were used to identify square occupancy, and for line extraction.

The Pilot

The Pilot is the crew member responsible for moving the body. It provides functionality for autonomous navigation and collision detection. Due to the fact that the arm on Calliope is too short to reach across the entire board, the robot must navigate between playable zones to manipulate pieces on the board.

The Pilot constructs a navigation plan and moves Calliope between designated zones of the board. It uses a combination of odometry and landmark-based localization with a particle filter to estimate the robot's location. Whenever visual evidence is acquired that suggests that Calliope is positioned incorrectly, the Pilot calculates and executes a path to the correct location.

The Grasper

The Grasper is responsible for controlling the arm. At the time the work reported here was done, the Grasper did not yet support the version of the arm we were using, so the inverse kinematic solver built in to Tekkotsu was invoked directly to solve arm trajectories. Calliope is the first robot that attempts to use the Tekkotsu framework for manipulation in 3D.

Playing Chess with the Chiara

Chess playing in the Tekkotsu framework was first implemented on the Chiara robot, as shown in Figure 1. The solution was then adapted for Calliope. A green and white board with blue and yellow pieces was used to make visual segmentation easier.

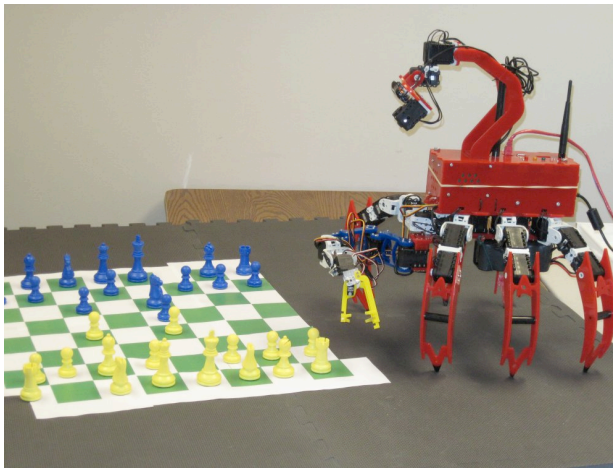

Figure 1: A Chiara hexapod robot with a special-purpose gripper designed for chess.

Localization

To localize, Chiara computed the average position of all yellow, green, and blue objects to the right, left, and in front of it. From this, an estimate of the board position relative to Chiara could be derived. The lower left corner of the board was then located and used as a reference point to determine the position and orientation of the robot.

Identifying Game State

The limited field of view and low height of the camera on Chiara prevented recovering the entire board state from one camera image. A routine that considered multiple partial boards constructed from multiple camera images was created to resolve this issue. Tekkotsu's dual coding vision system (Touretzky et al., 2007) assumes a 2D planar world, which is violated by the 3D nature of the pieces. For this reason the emphasis of the approach is on the occupancy of squares. Chess pieces are known to be tall, roughly cylindrical objects. The bottoms of the cylinders are of particular importance because it is the bottoms that will lie in the squares in the camera image. Occlusions between aligned pieces inspired the use of Chiara's ability to shift its body to achieve different viewing angles without taking a step. After this, line extraction took place to approximate the location of the board squares. A Hough transform grouped the line segments into continuous lines to deduce squares (Figures 2 and 3).

The current game state was then inferred considering the previous state, the set of legal moves, and the information obtained from integrating partial boards. If the state of the game remained ambiguous, the current approximate board was discarded and the process started over again.

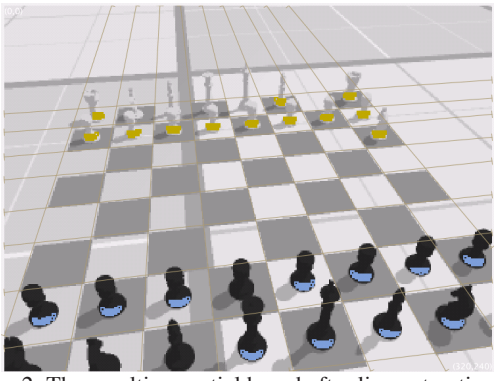

Figure 2: The resulting partial board after line extraction. Occlusions prevent some pieces from having an identified bottom.

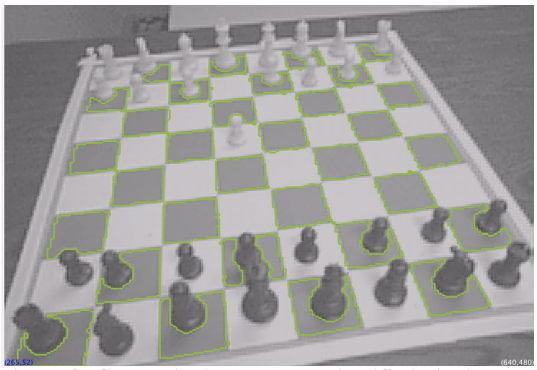

Figure 3: Green pixels next to unclassified pixels were assumed to identify the lines of the board.

Planning and Executing a Move

The GNUChess open source chess engine was used for move generation. Chiara used careful leg placement and body posturing to place itself as close to the playable area as possible. Calculations were made to find the optimal position of the body prior to grasping a piece, by finding solutions that would reduce the number of body movements needed to complete the move. Chiara had to lift its body to manipulate in 3D. First it raised itself high enough to reach over the pieces. Then the solution for the planar arm trajectory was calculated and executed. The gripper would next pitch down over the desired piece, grasp it, and pitch back up to align with the horizontal plane. Then the arm could move to its next position. A similar process was performed to place the piece in a new square. Multi-step moves, such as a capture, were accomplished by repeating this sequence. When the desired effect was achieved Chiara returned to a default position and waited for its opponent to move.

Playing Chess with the Calliope

Playing chess with the Calliope required tailoring the Chiara's approach to account for the differences between the two platforms. A new localization technique was needed due to the possibility of being on any of three sides of the board, with varying orientations.

Localization

Due to the inconsistency in the extraction of lines, a general localization routine could not be built solely from the results of line extraction. Adding heuristics to fill in missing lines could help but was not a sufficient solution, because if the visible lines were equally spaced, there was no way to tell whether the missing lines were on the left or the right. As the game progresses and pieces become distributed across the board, the occlusions to any line may be enough to disrupt the line extractor in one or more zones. To compensate for these problems with line extraction, the border of the board was marked with pink tape so that localization could be performed simply by reconstructing the border. Since pieces were constrained to lie inside the board, the border lines rarely suffered occlusions.

After the border is extracted (Figure 4), we use the locations of the corners of the board to refine our estimate of Calliope's location. This is possible because Tekkotsu's Pilot keeps a running estimate of the robot's displacement and orientation in the allocentric world, which is our starting point for localization. A least squares fit is calculated between the detected board location and the board that is in the allocentric map. We use this result as our estimate of Calliope's position and orientation.

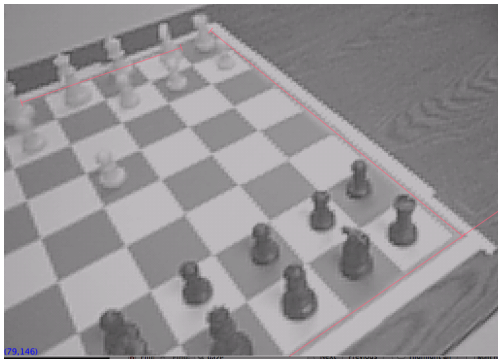

Figure 4: The extracted border from one viewing angle.

Identifying Game State

The approach to identifying the game state implemented on the Calliope is very similar to that of the Chiara. The first difference is that the anatomy of the Calliope allowed for recognition of the board state without moving the entire body. The height of the camera on the Calliope gave it a larger viewing area of the board. Unlike the Chiara, the Calliope cannot walk sideways or make small adjustments to the camera viewing angle by swaying its body. But Calliope could move its camera to look at different areas of the board, and in combination with its larger field of view, this proved sufficient.

The same logic of looking at the bottom of pieces was applied, as it is effective at handling occlusions given the height and angle of the camera relative to the board. If the derived board state was found to be ambiguous it was discarded and the board examination process repeated until a consistent state was reached.

Planning and Executing a Move

Calliope's arm is too short to reach across the entire board, so the board was broken up into four playable zones to allow Calliope to manipulate pieces on any side of the board efficiently. Once Calliope has successfully navigated to the zone of choice, and localized to refine its position estimate, the location of the piece of interest can be calculated from the position of Calliope using the information gained from the previous process of identifying the game state.

The arm is extended over the piece of interest. Since the Calliope arm is a 3D arm and can maneuver in the vertical plane, the inverse kinematics solver can directly solve for an arm trajectory that places the arm in position to grasp a piece. If the piece needs to be placed in a different zone, the Pilot is invoked to navigate to that zone and localization is done again. Afterward the piece is placed in the desired location (Figures 5, 6 and 7). If a multi-step move is needed, as with a capture or castle, the process is repeated for each step until the desired effect is achieved.

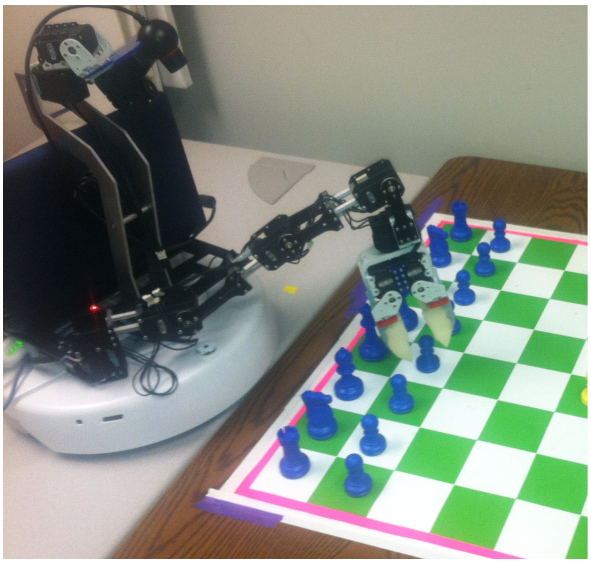

Figure 5: Calliope positioning its arm above a piece.

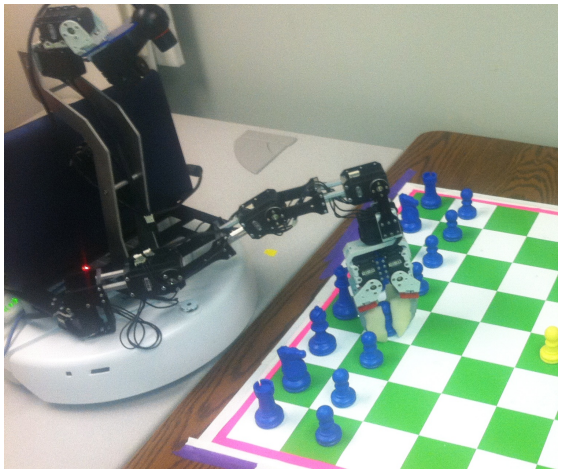

Figure 6: Calliope lowering its arm to place a piece.

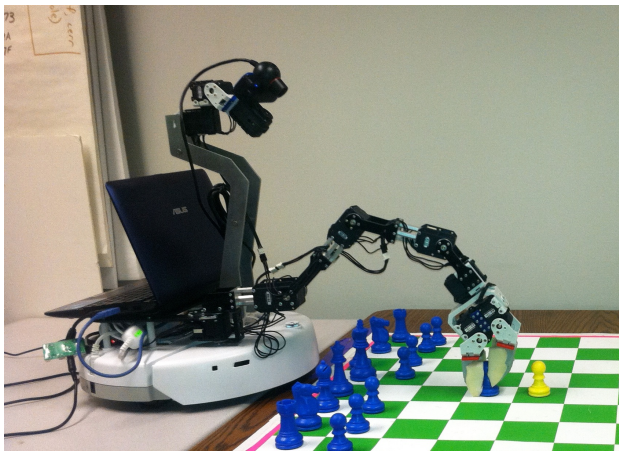

Figure 7: Piece successfully moved to the target square.

Discussion

Playing chess with a tabletop-scale robot such as Calliope is challenging because the robot must maneuver around the board to achieve its goals. The fact that the position of the robot changes with every move puts extra importance on the accuracy of localization. Other entrants in the 2011 AAAI small-scale manipulation challenge had sufficient reach to cover the entire board, and could thus play from a fixed position without ever needing to localize once initial calibration was complete (Chernova et al., in press).

The techniques developed for playing chess with Calliope should be applicable to many similar mobile manipulation tasks. The appeal of the Calliope platform is that it offers relatively low cost entry into the mobile manipulation space. Working at tabletop scale permits multiple robots to operate comfortably in a lab or classroom environment. And the design is open source; details are available at Chiara-Robot.org/Calliope.

Work on Calliope is continuing. A new model, Calliope5KP, has been developed that incorporates a Microsoft Kinect sensor to provide depth information in addition to RGB images. And a 3D arm path planner is almost ready for release. The next step will be to extend the Grasper to provide a variety of manipulation strategies that make full use of the arm's capabilities.

Acknowledgements

Jonathan Coens wrote the original chess playing application in Tekkotsu for the Chiara (Coens, 2010). Ashwin Iyengar assisted with the implementation of the inverse kinematics solver. RoPro Design is co-developer of the Calliope and provided several of its key components.

This work was supported by the National Science Foundation's Broadening Participation in Computing program under award 1042322 (ARTSI Alliance).

References

Anderson, M. D., Chernova, S., Dodds, Z., Thomaz, A. L., and Touretzky, D. S. (2011) Report on the AAAI 2010 robot exhibition. AI Magazine, 32(3):109-118.

Chernova, S., Dodds, Z., Stilman M., Touretzky, D. S., and Thomaz, A. (in press) Report on the AAAI 2011 robot exhibition. AI Magazine, to appear.

Coens, J. 2010. Taking Tekkotsu Out of the Plane. Masters thesis, Computer Science Department, Carnegie Mellon University, Pittsburgh, PA. Available online at http://Chiara- Robot.org/Chess.

Kuffner, J. J. Jr., and LaValle, S. M. 2000. RRT-Connect: An efficient approach to single-query path planning. Proc. IEEE International Conference on Robotics and Automation (ICRA- 2000).

Touretzky, D. S. 2010. Preparing computer science students for the robotics revolution. Comm. ACM, 53(8):27-29.

Touretzky, D. S., Halelamian, N. S., Tira-Thompson, E. J., Wales, J. J., and Usui, K. 2007. Dual-coding representations for robot vision in Tekkotsu. Autonomous Robots, 22(4):425-435.

Touretzky, D. S., and Tira-Thompson, E. J. 2010. The Tekkotsu "Crew": Teaching robot programming at a higher level. First AAAI Symposium on Educational Advances in Artificial Intelligence. Menlo Park, CA: Association for the Advancement of Artificial Intelligence.

Touretzky, D. S., Watson, O., Allen, C. S., and Russell, R. 2010. Calliope: Mobile manipulation from commodity components. Technical report WS-10-09: Papers from the 2010 AAAI Robot Workshop. Menlo Park, CA: Association for the Advancement of Artificial Intelligence.

Watson, O., and Touretzky, D. S., 2010. Navigating with the Tekkotsu Pilot. Proc. 24th Florida Artificial Intelligence Research Society Conference.