In Proceedings: IEEE International Conference on Intelligent Robots and Systems (IROS), 2009.

Robot Jenga: Autonomous and Strategic Block Extraction

Jiuguang Wang, Philip Rogers, Lonnie Parker, Douglas Brooks, and Mike Stilman

***Abstract*— This paper describes our successful implementation of a robot that autonomously and strategically removes multiple blocks from an unstable Jenga tower. We present an integrated strategy for perception, planning and control that achieves repeatable performance in this challenging physical domain. In contrast to previous implementations, we rely only on low-cost, readily available system components and use strategic algorithms to resolve system uncertainty. We present a three-stage planner for block extraction which considers block selection, extraction order, and physics-based simulation that evaluates removability. Existing vision techniques are combined in a novel sequence for the identification and tracking of blocks within the tower. Discussion of our approach is presented following experimental results on a 5-DOF robot manipulator.**

***Index Terms*— Jenga, Entertainment Robot, Block Extraction, Game Strategy, Movable Obstacles, Pose Estimation**

I. Introduction

JENGA [1] is a popular game marketed by Hasbro, consisting of 54 wooden blocks constructed into a tower of three blocks per level (Fig. 1). Players take turns identifying, removing loose blocks, and placing them on top of the tower. As the game progresses, the tower becomes increasingly unstable. The objective of the game is to build the tallest tower possible (in a single-player setting) or avoid being the player that collapses the tower (in a multi-player setting). Tower instability, uncertainty in pressure distribution, and imprecise force control are all known to be difficulties for human players, where the world record is 40 and 2/3 levels. A robot manipulator that plays Jenga faces an even greater challenge, where the robot must also recognize and localize Jenga pieces as well as track their unexpected motion.

Due to these complex physical requirements, the design of a robot Jenga system has received significant attention in the robotics community. South [2] created a spring-based physics simulation model to identify a playing strategy, but simulated a portion of the Jenga tower and not the full 18-level structure. Ziglar [3] approached the block extraction from a mechanics point of view and analyzed the translational and rotational moves for the likelihood of disturbing other blocks in the tower assuming a piecewise planar model; the results are used to determine moves that do not disturb the configuration tower, forming a play strategy. Barnum [4] constructed a simulator based on the commercial NVIDIA PhysX engine which is capable of simulating the full tower, and analyzed its stability with particular considerations to the center of gravity and the forces of friction. Kroger, et al [5] created a robotic Jenga platform to demonstrate multi-sensor integration using a Staubli RX 60 industrial manipulator, four PCs, a 6D force/torque sensor, a 6D acceleration sensor, a laser triangulation distance sensor, and two CCD cameras for visual feedback. The system was able to construct ten additional levels (29 blocks) on top of an 18-level tower, using no strategy beyond the random selection of blocks.

The authors are with the Center for Robotics and Intelligent Machines (RIM) at the Georgia Institute of Technology, Atlanta, Georgia, 30332, USA. Emails: {j.w, progers7, lonnie, douglas.brooks}@gatech.edu, mstilman@cc.gatech.edu

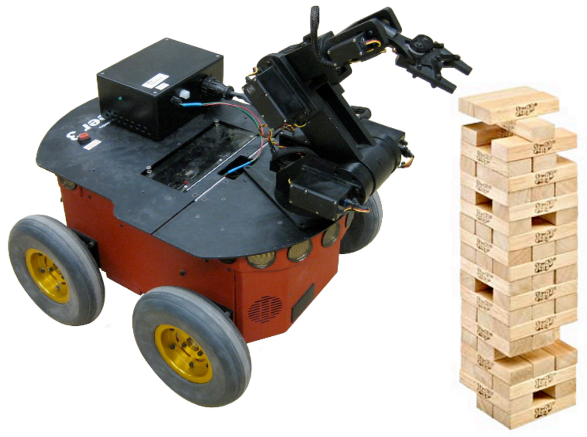


Fig. 1. A Pioneer 3-AT robot and the associated 5-DOF servo manipulator, with a standard Hasbro Jenga tower. Image courtesy of Hasbro, Inc.

The main contribution of the paper is the development and evaluation of an autonomous and *strategic* Jenga system, with particular consideration given to the planning of block extraction. In this work, we use a three-stage planner for block extraction which explicitly tracks the extraction history and simulates tower stability using a physics engine. Using active contour-based block pose estimation and optical flow-based tower movement detection, we do not assume a structured environment such as the painted blocks used in the Kroger platform. We demonstrate our approach using low-cost, off-the-shelf components including a five degrees of freedom servo manipulator (shown in Fig. 1), two CMOS cameras, and a single desktop for all sensing and control tasks. While the performance obtained was not comparable to the Kroger platform, the results are promising given the drastic reduction in cost and resources.

The remainder of the paper is organized as follows. Section II formulates the Jenga game as a planning problem and proposes several strategies for use with a robotic manipulator. Section III describes the hardware and software architecture for the robotic Jenga system. Section IV describes the results obtained by applying our system with a simple robot on a generic Jenga tower. Finally, Section V provides concluding remarks and directions for future work.

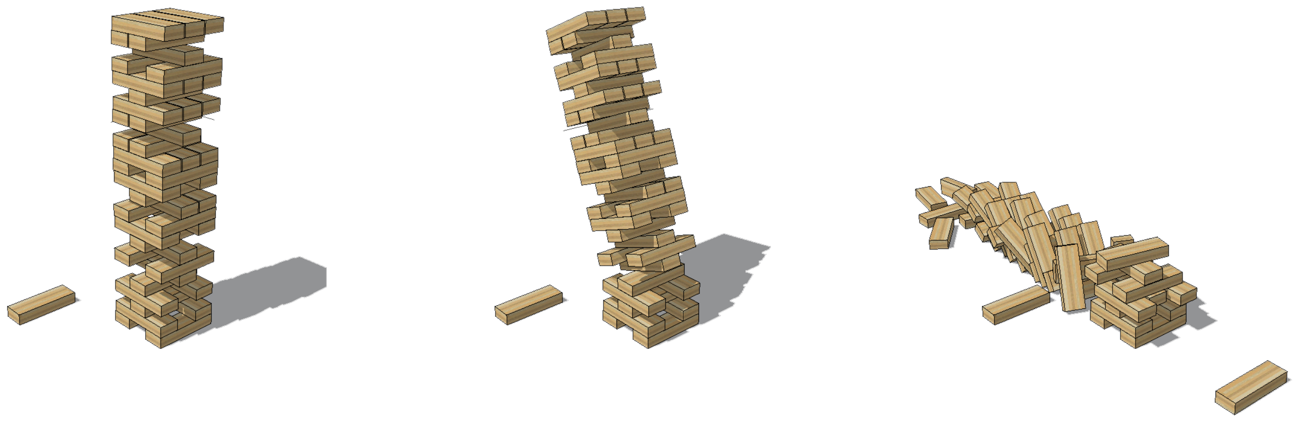


Fig. 2. A physics-based simulation showing that a particular removal leads to a tower collapse.

II. Strategy

A. Overview

We present three-stage strategy, coupled with a vision-based error detection scheme, for autonomous extraction of blocks within a Jenga tower. First, we limit our block selection to side blocks to maximize the number of removable pieces. Second, we select a particular block from a list of potential choices generated in the first stage and identify a single block from a decision table. Third, a physics-based simulation is conducted for the final block selection to identify the feasibility of removing that particular block. If the removal is deemed feasible, the robot proceeds to extract the block using vision-based error detection to monitor tower stability and abort if appropriate.

B. Removal strategy

On a tower with three blocks per level, there are two types of blocks available for removal, a single center block and two side blocks. There are advantages and disadvantages to both types: removing the center block ensures the stability of the tower, since there are still two side blocks on that level to support the pieces on top; however, there are no longer any available blocks for removal on that level. Removing the side blocks pushes the tower towards instability, but maximizes the number of removable blocks.

While our algorithmic strategy has no constraints on which blocks to remove, our platform adds some restrictions. The five-axis Pioneer arm cannot simultaneously control all six degrees of freedom for the end-effector. In the general case, translation of a block implies its rotation about some axis. For the middle block this necessarily implies a disturbance to the neighboring pieces. For the side blocks, we select motions that couple lateral translation with rotation about the vertical axis. The robot grips the small rectangular end of the Jenga block, and rotates the piece to the side. Aside from servo error, our approach causes no additional forces on neighboring blocks.

C. Block Removal Planner

Based on the removal strategy, we generate a list of candidate blocks for removal at every step. The Block Removal Planner (BRP) selects a block in this list based on heuristic evaluation, and passes the information to the Removal Feasibility Planner for further analysis.

The BRP tracks the history of removal attempts in a decision table which is used as the basis for new removal choices. The decision table couples each removable block with a tower movement threshold. The movement threshold is initialized to a pre-defined value and updated as the game progresses. The threshold keeps track of how much the tower moved when the robot last attempted to remove the particular block. By tracking earlier block motion, BRP can rank blocks according to threshold values. Hence, it identifies the block that is *least likely to affect tower stability*.

For every successful block removal, an additional validation step is used to consider previously failed extractions. As a particular block is removed, the forces due to weight and friction acting on its neighboring blocks change. If one of the neighboring blocks was previously marked unremovable, then this change justifies another attempt. Consequently, with every successful removal, we iteratively mark the neighboring blocks for a new attempt.

D. Removal Feasibility Planner

Once the Block Removal Planner returns a potential choice, the Removal Feasibility Planner (RFP) conducts a physics-based simulation to determine if this removal choice could potentially lead to tower collapse. If simulation results in a collapse as shown in Fig. 2, then the BRP is asked to provide an alternate choice and the process is repeated. We chose the Newton Physics Engine, which has previously been shown to produce accurate approximations of static friction, outperforming all other engines compared in [6]. Depending on the size and structure of the tower, the RFP typically requires several seconds of computation on a standard desktop computer to complete a simulation.

Once the RFP verifies that the block can be removed in simulation, the robot proceeds with the manipulation as described in Section III. During a block removal, the motion of the tower is continuously monitored, and a stop signal is issued in the event that the tower movement exceeds the threshold. If a stop occurs, then the final movement

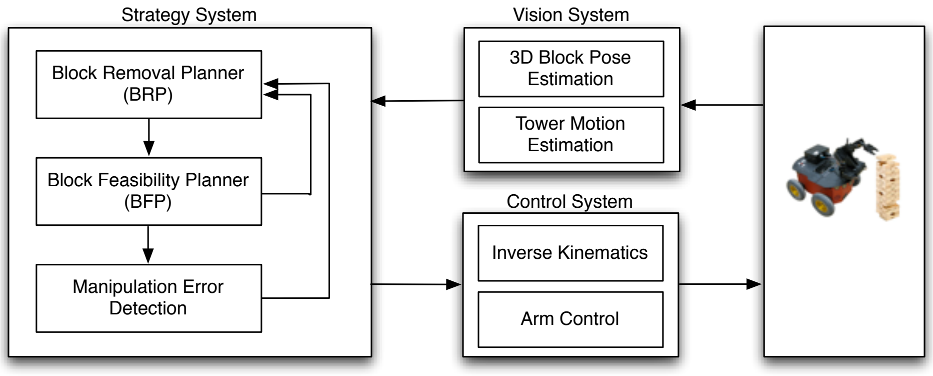


Fig. 3. The Jenga architecture with strategy, vision, and control subsystems.

value is recorded in the decision table, and the BRP is asked to provide an alternate choice. The tower movement is determined using a vision-based technique, also described in Section III.

III. Architecture

A. Overview

Our robot Jenga player consists of inexpensive, off-the-shelf components. Compared to the Kroger platform, which uses a Staubli manipulator, we use a low-cost five-DOF Pioneer servo arm. Instead of multiple PCs, we use a single desktop computer (2.0 GHz, 2 GB of RAM) to make decisions about motion, control the manipulator and process all sensor data. Our system only employs two CMOS cameras for visual feedback and no additional sensors.

Despite our limited equipment, the system operates successfully with standard Jenga blocks and minimal control over lighting. In contrast to the structured environments described in previous work we do not assume painted/marked blocks and instead use monocular pose estimation to identify, localize, and track blocks with respect to the robot arm.

The software architecture for our Jenga system consists of two subsystems, each with two components. First, the vision subsystem uses active contours for pose estimation and optical flow for motion estimation. Second, the control subsystem applies closed form inverse kinematics and controls arm motion. The structure of this architecture is given in Fig. 3

B. Block Pose Estimation

The pose estimation subsystem uses a CMOS camera facing the front side of the Jenga tower from an angle. It is tasked with determining the 2D pose of all candidate blocks on this plane with the assumption that the distance from the robot to the tower is known. During setup, a calibrated camera image is transformed using planar homography so that one tower side appears from the manipulator's reference frame. The pose estimation problem can then be separated into 3 steps: (1) locating candidate blocks, (2) tracking blocks across time, and (3) calculating pose. This is accomplished using existing techniques applied in a novel sequence, overcoming variations in block color, texture, and other uncertain variables.

1) Locating candidate blocks: First, the image is processed using four filters with intermediate results shown in Fig. 4). The image is converted to grayscale, then subjected to Gaussian blur (to reduce woodgrain), Canny edge detector [7], and dilation using a square kernel. The resulting binary image is searched for connected regions of appropriate area using the topological method outlined in [8]. Any regions found are used to initialize the block tracker.

2) Tracking blocks: The tracker is based on an active contour model [9]. Because the Canny edge detector provides reasonable contour initializations, it is again used as the external energy measure of the active contours. This is implemented as a minimization of the negative Canny edge value along the block's contour, and has proven effective in our experiments. We apply a heuristic method to infer the location of missing blocks from the even-odd level structure of the tower and the presence of other tracked blocks on a given level.

Block tracking is suspended when a block is occluded by the manipulator if either of the following conditions are met:

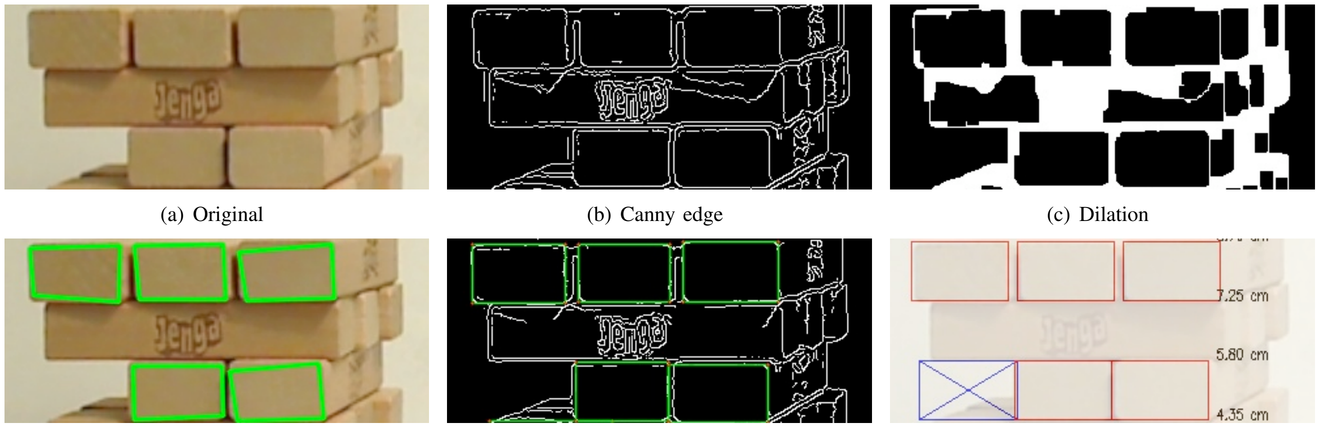
- Area: Given block area a_b and expected area a_e, a block is removed if $|a_b - a_e| > \theta_a$
- Cross-ratio: given four corners $C_1 \ldots C_4$, the cross ratio is equal to

$$CR(C_1, C_2, C_3, C_4) = \frac{(C_1 - C_3)(C_2 - C_4)}{(C_1 - C_4)(C_2 - C_3)} \quad (1)$$

Any four points are invariant with respect to their cross ratio under a perspective transformation [10] so for known block corners $b_1 \ldots b_4$ and estimated block image corners $i_1 \ldots i_4$, a block is removed if

$$|CR(b_1, b_2, b_3, b_4) - CR(i_1, i_2, i_3, i_4)| > \theta_{CR} \quad (2)$$

3) Calculating pose: Due to the simplification of blocks being attached to one side of the tower, the pose calculations are reduced to calculating block position relative to the tower plane. Because of the planar homography calculated during setup, an image of the tower from the manipulator's



(a) Original (b) Canny edge (c) Dilation

(d) Contour (e) Snake (f) Pose

Fig. 4. Intermediate results from filtering, used to locate the candidate blocks.

perspective is known. Using a constant ratio of pixels to real units (cm), the x and y components are computed as follows:

$$world_x = image_x \times \frac{cm}{pixels} \tag{3}$$

$$world_y = image_y \times \frac{cm}{pixels} \tag{4}$$

The block tracking and pose estimation results are shown in Fig. 4.

C. Tower Motion Estimation

Vision is also used to estimate tower stability as it relates to the motion of the tower while the robot attempts to remove a block. If the movement exceeds a threshold, a stop command is issued, and the movement value is recorded. To accomplish this, we compute the optical flow field of the tower as seen by an overhead CMOS camera, and compute a movement value based on the magnitude of the field.

We divide the motion estimation problem into several stages. First, we select easily identifiable features that can be tracked across frames. Next, we run the Lucas-Kanade algorithm to compute the optical flow field with respect to the movement of the blocks. Finally, we calculate a movement value based on the magnitude of the field.

1) Feature Selection: In order to compute tower motion we must first select robust features that can be relied on to compute optical flow. Initially, we chose Harris corners [11] as features to track, but encountered significant errors in the motion estimation, as many of the corners are not readily identifiable between frames. Instead, we settled on the feature selection criterion described in [12]. Based on the Shi and Tomasi definition, good features to track in motion estimation are features that minimize a dissimilarity function while using affine motion as the underlying image change model. This has been shown to be particularly compatible with the Lucas-Kanade algorithm discussed below.

In experimentation, we found that while the Shi and Tomasi definition returned fewer features, the quality of the keypoints was superior to that of Harris. We were able to produce consistent matching results with relatively low computation.

2) Optical Flow: The Lucas-Kanade algorithm [13] is one of the most widely used algorithms for estimating optical flow. A template with an extracted sub-region (a small window) of the image is locally displaced in an attempt to minimize the squared error between two images. The Lucas-Kanade algorithm assumes three basic conditions,

- Brightness constancy: a pixel does not change in appearance between frames. In grayscale the brightness should not change.
- Temporal persistence: the motion of a feature window changes slowly in time.
- Spatial coherence: neighboring points in a scene, belonging to the same surface, have similar motions, and project to nearby points on the image plane.

With controlled lighting, small movement, and standard tower shape, our system satisfies all three assumptions. To avoid the aperture problem that results from large motions and without using a large window (which breaks the small and coherent motions assumption), we run the Lucas-Kanade algorithm in a pyramidal fashion. In this approach, we construct a Gaussian image pyramid of several layers, with increasing resolution from top to bottom. The motion estimation from each preceding level is taken as a starting point for estimating motion at the next level. This provides a low-cost assurance of accurate motion estimation.

3) Movement Magnitude: Once the optical flow field is calculated via Lucas-Kanade, we simply sum the flow field to obtain a value representing the overall movement in the frame. This value is then used by the planner described in the previous section for error detection. The threshold is determined experimentally. If needed, the flow field can also provide the dominant direction of flow, but this was not used in our platform.

D. Arm Control

The motion of our servo-based robot arm was necessarily position controlled. We found a geometric closed form solution for the inverse kinematics of the Pioneer manipulator. Due to the limited range of motion required for removing individual blocks from the tower, we elected to keep the gripper orientation facing towards the blocks and only varied its position when approaching blocks.

The low-level arm control was implemented in the Player Project [14], a popular middleware for a variety of physical and simulated robots. The Player system follows a server-client architecture, in which a sever running the Pioneer driver continuously listens for motor commands. The control algorithm was implemented as a client to the Player server which sends the appropriate joint commands. In this architecture, the individual components can be separated to different processors or computer, in the case that a single process cannot reliably control the arm in conjunction with other computationally intensive vision components.

IV. Experiments & Results

In the experiments conducted, four adjustments were made to the game of Jenga to simplify the system:

- Due to the limited length of the manipulator, a nine-level tower was used instead of the full eighteen-level structure.
- Due to the limited width of the gripper, the side blocks are pre-pulled forward approximately 0.5 cm from the tower; if a wider gripper is used, the extraction could take place from the longer side of the blocks, making this step unnecessary.
- After a block is removed, instead of placing it on top of the tower, the block is simply dropped to the side to simplify the manipulation task (i.e., the tower is not reconstructed).
- If the tower is disturbed during the block extraction process, the out of place blocks are not corrected (where in the official game, this is required).

Fig. 7 shows an example run of the the block extraction process, demonstrating various aspects of the decision process discussed in the previous sections. There are six frames in the figure taken from the accompanying video: in the first frame, a tower is constructed with eight side blocks pre-pulled. In the second frame, the gripper is in place, ready to extract the block. In the third frame, block is successfully extracted. In the fourth frame, a new block is selected by the BRP, and the gripper is in place. In the fifth frame, the previous extraction attempt exceeded movement threshold and a new block is selected by the BRP. Finally, in the sixth frame, another block is successfully extracted from the tower.

Trials are set up to test the performance of the system. Out of 20 trials, we recorded the number of total attempts, successful extractions, and aborts due to tower movement; the statistics are shown in Fig. 5 and Fig. 6. Nine of the 20 trials successfully extracted three blocks, two trials failed to extract any blocks, and one trial extracted five blocks. We also calculated the average success percentage, i.e., in a single trial the number of blocks extracted divided by the total number of attempts (which includes aborts). Overall, approximately half of the trials have an average success percentage of 90% or above.

In the various components we used, such as history tracking, physics simulations, reactive stopping, etc, the vision-based pose estimation and movement detection were the most relevant and critical to the success of block extraction. The accuracy of the vision results directly affects the planning and manipulation tasks and when incorrect readings are obtained, the extraction is more likely to fail. We typically experienced two types of pose estimation failures: failure to locate a candidate block and failure to calculate the true pose of the block. While the former simply implies that a block will not be considered in the planners, the latter lead to more serious consequences, as an incorrect pose almost always resulted in manipulation failures.

While we believe the physics simulation is a valuable component in a successful Jenga strategy, we found that it did not play a significant role in our experiments and often did not catch instability in the reduced-size tower. The simulation can reliably capture simple errors, e.g., removing the only support block on a level; however, for other stability simulations, the inaccuracy in vision and in the simulation engine itself often prevents it from returning valuable results.

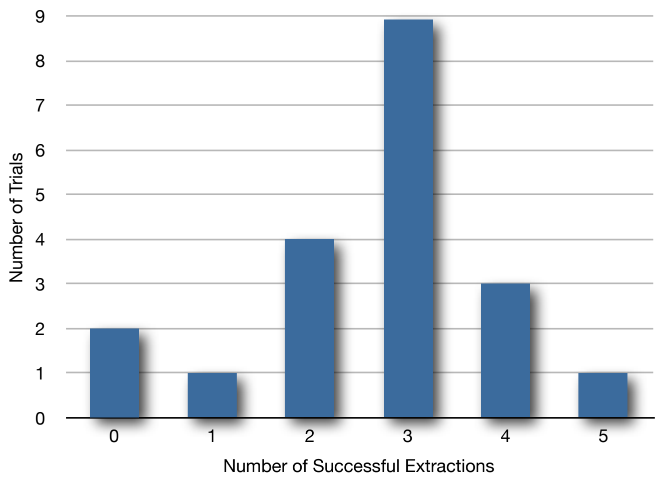


Fig. 5. Number of successful extractions for each of the 20 trials.

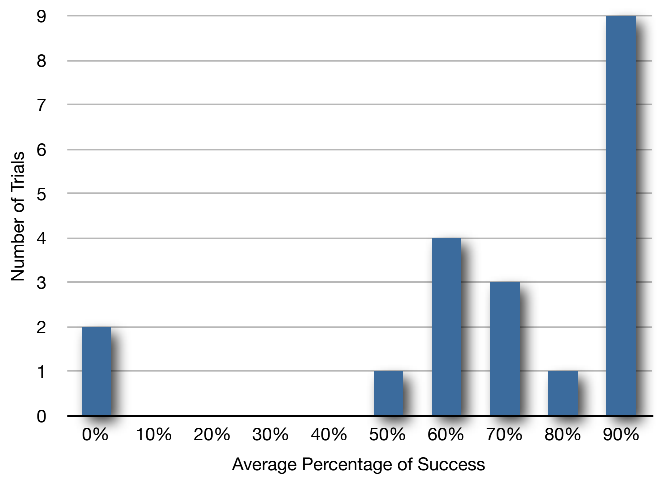
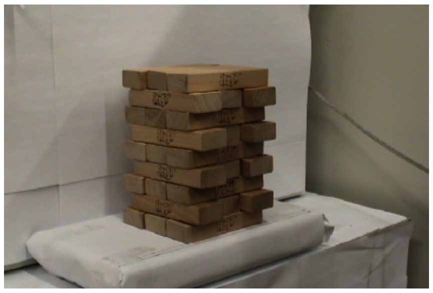
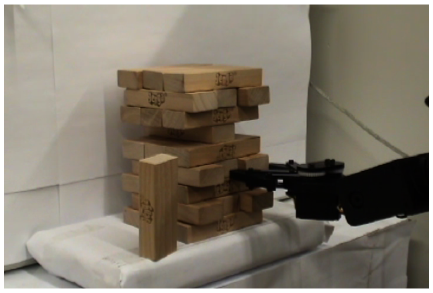
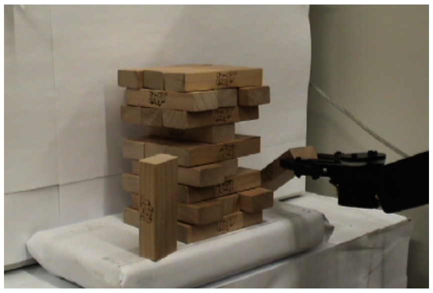
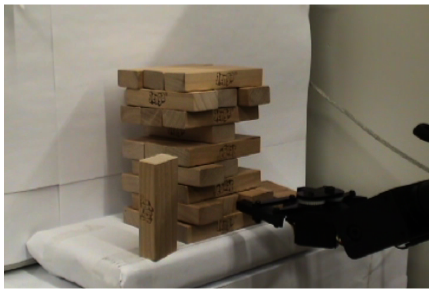


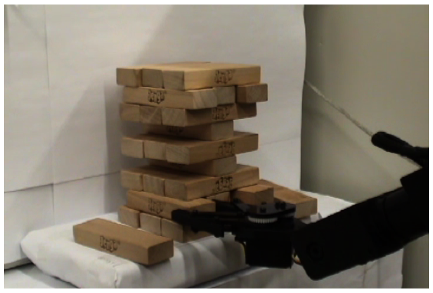
Fig. 6. Average number of successful extractions per trial out of 20 trials.


(a) Full tower setup with eight side blocks pulled 0.5cm forward.


(b) A new block is selected by the Block Removal Planner.


(c) The block is successfully extracted from the tower.


(d) A new block is selected by the BRP.


(e) The previous extraction attempt exceeded the movement threshold. A new block is selected.

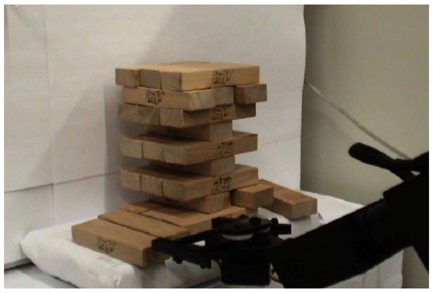

(f) The block is successfully extracted from the tower.

Fig. 7. Example block extraction run showing the decision making process and error detection.

V. Conclusions

In this paper, we have presented the design and evaluation of an autonomous Jenga robot, using low-cost, off-the-shelf components. The main contributions of this work are the strategies in a Jenga game, which consist of block removal, tracking extraction history, and the physics behind tower stability. We have greatly reduced the number of sensors required for block extraction compared to previous implementations, and given the limited resources available, the results are highly encouraging.

Some immediate steps towards a better Jenga system can be taken in the vision and physics domain, as was discussed in the previous section. A more advanced manipulator could be used to remove many of the limitations and assumptions. More sophisticated pose estimation and movement detection could greatly reduce the failure rate for the manipulation process. Finally, the integration of force feedback may further reduce failures due to tower movements and hence enhance playing capabilities.

Acknowledgments

We are grateful to Dr. Ayanna Howard for providing the equipment used and Dr. Jim Rehg for his advice in the vision components of this project. We also thank Peter Barnum, Chung Hyuk Park, Hae Won Park, Sung Hyun Park, Sekou Remy, and Stephen Williams for their valuable comments and suggestions. This work was done as part of the fall 2008 course *Robot Intelligence: Planning in Action* at Georgia Tech.

References

[1] (2008, December). [Online]. Available: http://www.hasbro.com/games/family-games/jenga/

[2] M. South, "A Real-Time Physics Simulator for Jenga^{TM}," Master's thesis, University of Sheffield, South Yorkshire, England, May 2003. [Online]. Available: http://www.dcs.sheffield.ac.uk/intranet/teaching/projects/archive/ug2003/pdf/u0ms4.pdf

[3] J. Ziglar, "Analysis of Mechanics in Jenga," April 2006. [Online]. Available: http://www.cs.cmu.edu/afs/cs/academic/class/16741-s06/www/projects06/ziglar_jenga.pdf

[4] P. Barnum, "Algorithmic Analysis of the Stability of Stacked Objects," 2006. [Online]. Available: http://www.cs.cmu.edu/afs/cs/academic/class/16741-s06/www/projects06/BarnumAlgorithmic2006.pdf

[5] T. Kroger, B. Finkemeyer, S. Winkelbach, L.-O. Eble, S. Molkenstruck, and F. M. Wahl, "A Manipulator Plays Jenga," *Robotics & Automation Magazine, IEEE*, vol. 15, no. 3, pp. 79–84, September 2008.

[6] A. Boeing and T. Bräunl, "Evaluation of real-time physics simulation systems," in *Proceedings of the 5th International Conference on Computer Graphics and Interactive Techniques in Australia and Southeast Asia*. New York, NY, USA: ACM, 2007, pp. 281–288.

[7] J. Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 679–698, 1986.

[8] S. Suzuki and K. Abe, "Topological structural analysis of digitized binary images by border following," *Computer Vision, Graphics, and Image Processing*, vol. 30, no. 1, pp. 32–46, 1985.

[9] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active contour models," *International Journal of Computer Vision*, vol. 1, no. 4, pp. 321–331, 1988.

[10] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge university press, 2003.

[11] C. Harris and M. Stephens, "A combined corner and edge detector," in *Alvey Vision Conference*, vol. 15, 1988, p. 50.

[12] J. Shi and C. Tomasi, "Good features to track," in *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR'94., 1994 IEEE Computer Society Conference on*, 1994, pp. 593–600.

[13] S. Baker and I. Matthews, "Lucas-Kanade 20 Years On: A Unifying Framework," *International Journal of Computer Vision*, vol. 56, no. 3, pp. 221–255, 2004.

[14] B. Gerkey, R. Vaughan, and A. Howard, "The player/stage project: Tools for multi-robot and distributed sensor systems," in *Proceedings of the 11th International Conference on Advanced Robotics*, 2003, pp. 317–323.