Dropping Disks on Pegs: a Robotic Learning Approach

Adam Campbell

Cpr E 585X Final Project Report

Dr. Alexander Stoytchev

21 April 2011

Table of Contents:

Introduction..3

Related Work..4

Experimental Setup..5

Method...6 - 8

Results...9

Future Work..10

Bibliography...11

Introduction:

This project aims to create a developmental approach to the problem of dropping disks on pegs. Although the concept of dropping disks on pegs does not have any overtly obvious applications, the development process and learning methods behind the project could be applied to other projects in the future.

Many children, in their developmental stages early in life, interact with different objects to learn the properties of those objects. There are many toys which challenge children to think, and several of these toys involve putting disks on pegs. It's not that the act of placing a disk on a peg is terribly useful in the real world, but rather the knowledge that certain objects can fit through other objects, and other such realizations help the children to proceed in development.

An application that could prove useful for the ability to drop disks on pegs would be to solve the Towers of Hanoi. Invented in 1883 by Edouard Lucas, the game's objective is to move the stack of disks from one peg to another [2]. Only one disk may be moved at a time, and a disk can only be placed on top of a disk that is smaller than it.

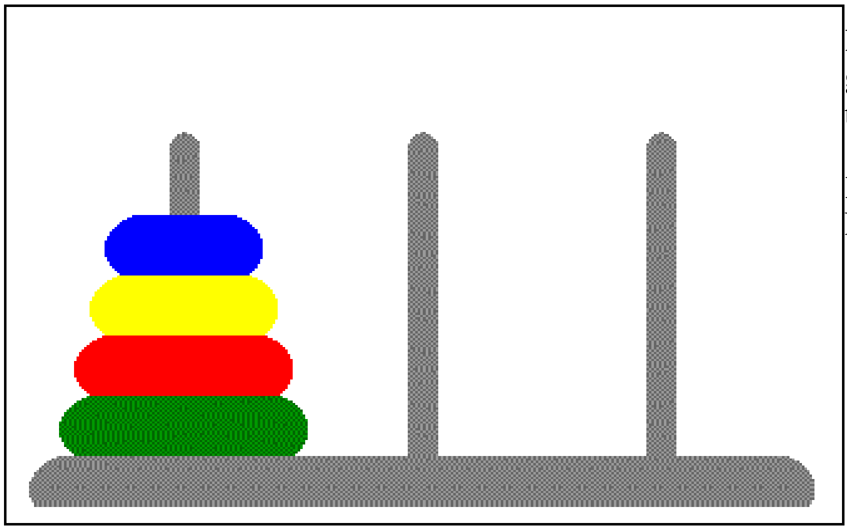

Figure 1. The Towers of Hanoi game is an application of the ability to drop disks on pegs.

Image source: http://www.numerit.com

Related Work:

Nate Koenig of the Interaction Lab at the University of Southern California carried out a study that had people act as teachers to help a robot learn how to solve the Towers of Hanoi [3]. Koenig believes that people will need to help robots learn in the future because it will be unfeasible for them to come preprogrammed with everything. Similarly, in 2007, Chang *et al.* at Carnegie Mellon University made a robot that could solve the Towers of Hanoi problem [4].

While these examples showcase the solving aspect of the Towers of Hanoi problem, this project focuses on the process of getting the disk on the peg. This project is not concerned with solving the Towers of Hanoi, but in order to do so, a robot must be able to place disks on pegs.

Experimental Setup:

The experimental setup (pictured in Figure 2) consisted of the following:

- The robot, which is an upper-torso humanoid robot. It has two 7-degree-of-freedom whole-arm-manipulators and two BarretHands by Barret Technology.

- The peg, which is part of the Rock-A-Stack toy from Fisher-Price, Inc.

- The disk, which is made of hard styrofoam and has been spray-painted green to make color-tracking easier.

- A Linux computer issuing commands to the robot at 500 Hz, and at the same time storing the data collected by the robot.

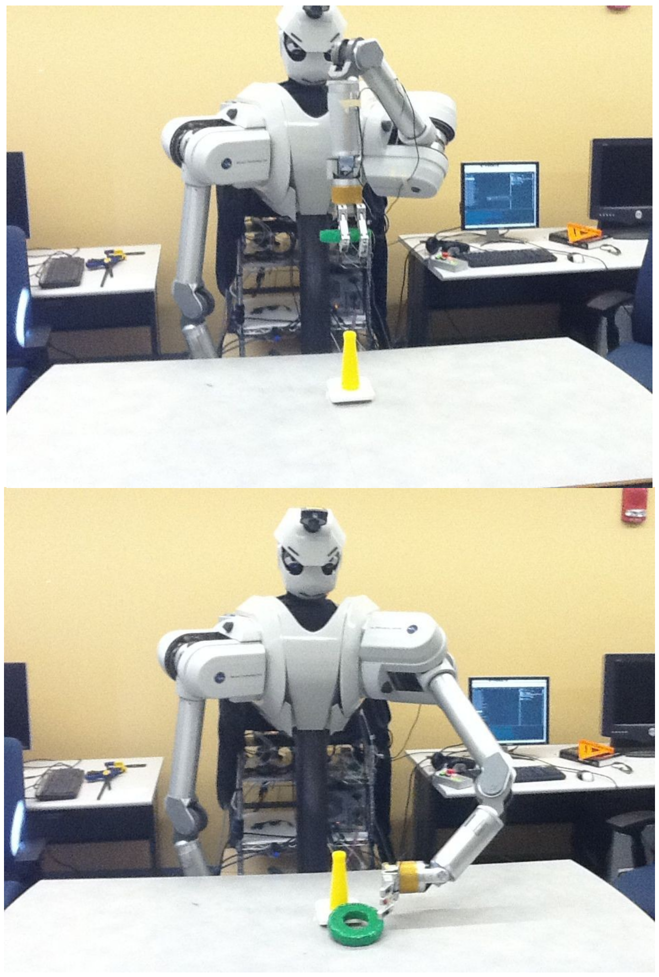
Figure 2. The setup used in the experiment. A ZCam by 3DV Systems mounted on the robot's head allows it to collect audio and visual data, and the joint positions are constantly being monitored. The robot would first drop the disk on the peg (top) and then push the peg (bottom).

Method:

The following pseudocode outlines the procedure used to experiment and collect data.

```
Repeat:
    Position peg & platform on center of table
    Move arm to default position above peg
    Put disk in robot’s hand
    Close robot's hand
    Move arm to random_position()
    Drop disk , collect audio data during drop
    Take picture after drop
    Push peg & platform
    Take picture after push
End repeat
```

The following pseudocode outlines the approach used to determine if the drop was successful or not:

```
for each TRIAL in TRIALS:
    before_push_image = TRIAL->get_before_push()
    after_push_image = TRIAL->get_after_push()
    peg_image_before = threshold_image_for_peg(before_push_image)
    disk_image_before = threshold_image_for_disk(before_push_image)
    peg_image_after = threshold_image_for_peg(after_push_image)
    disk_image_before = threshold_image_for_disk(after_push_image)
    peg_centroids = get_peg_centroids(before_and_after_images)
    disk_centroids = get_peg_centroids(before_and_after_images)
    centroid_distance = distance_between_peg_and_disk_centroids
    if (change_in_centroid_distance < threshold)
        add_to_successful(joint_data_at_drop_position)
    else
        add_to_unsuccessful(joint_data_at_drop_position)
    end if
end for
```

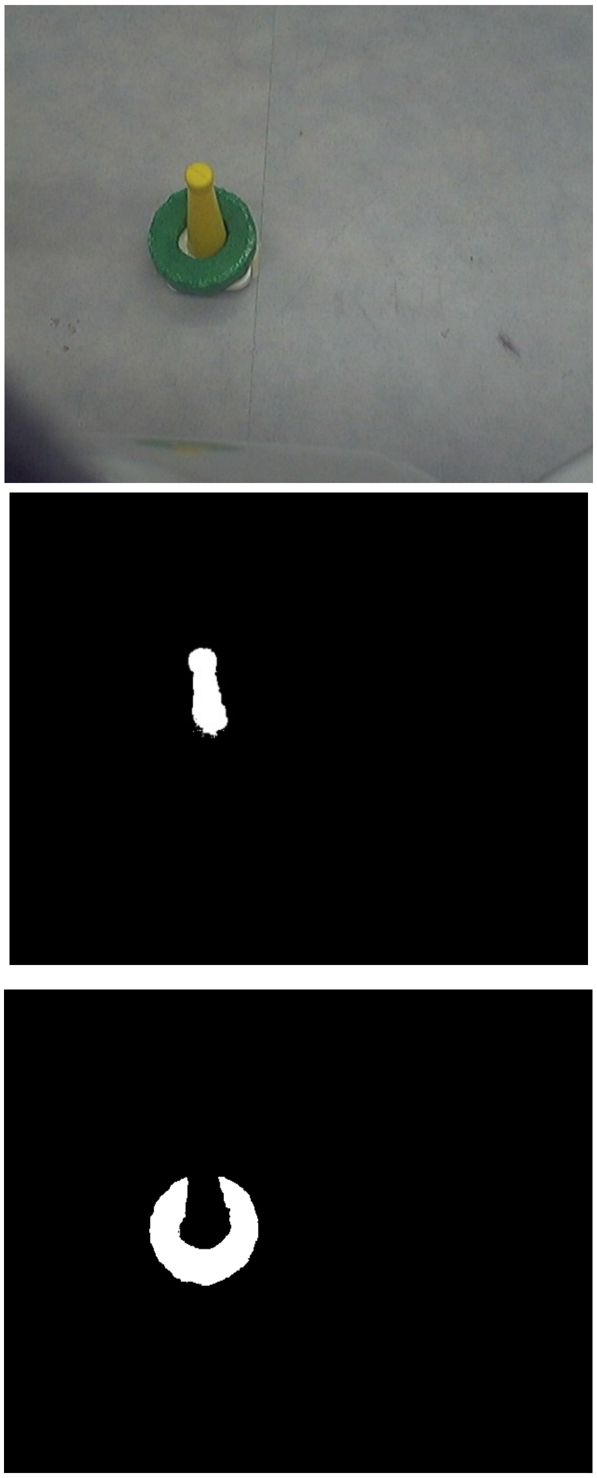
Figure 3. The visual data collected after a successful drop. The image was thresholded to retrieve the peg and disk blobs, which could be used to determine the distance between the disk and the peg. An unsuccessful trial had similar images, except that the disk blob would be farther away from the peg blob, and it would be a complete circle instead of the c-shape shown here.

After the drop, the robot collected video data of the scene. The robot was continuously recording video throughout the entire experiment, so it was possible to take a picture by simply doing nothing for a brief period of time.

Visual analysis was done using the OpenCV computer vision library [5]. To determine whether or not a drop was successful, the images were first thresholded to produce blobs for the peg and the disk (see Figure 3). The thresholding was done manually, as specific colors for the disk and the peg were given as parameters. The constraints on the thresholding were such that it was general enough to fit every shade of that colors imaginable, but separate colors would not be confused with each other.

After the thresholding, the centroids of the blobs were found using the moments of the image. This made it possible to find the distance between the peg and the disk. If the distance was greater than a certain limit, then the disk could not possibly be on the peg, and the drop was declared a failure.

This process was applied to the images both before and after the robot pushed the peg, so the total distance traveled by each component could be found. If the peg and the disk did not move approximately the same distance, then no comovement was detected and the drop was declared a failure (see Figure 4).

Since it was also possible that the disk would fall off the table completely, and therefore not be visible in the images, the drop was declared a failure if no disk was detected after thresholding (see Figure 5).

Any trial that did not fail the above tests was put in the 'successful' category.

To keep track of results, two log files were created, representing successful and unsuccessful trials. After a drop was deemed either successful or unsuccessful, the appropriate log file was updated with the Cartesian position of the robot's arm during the drop.

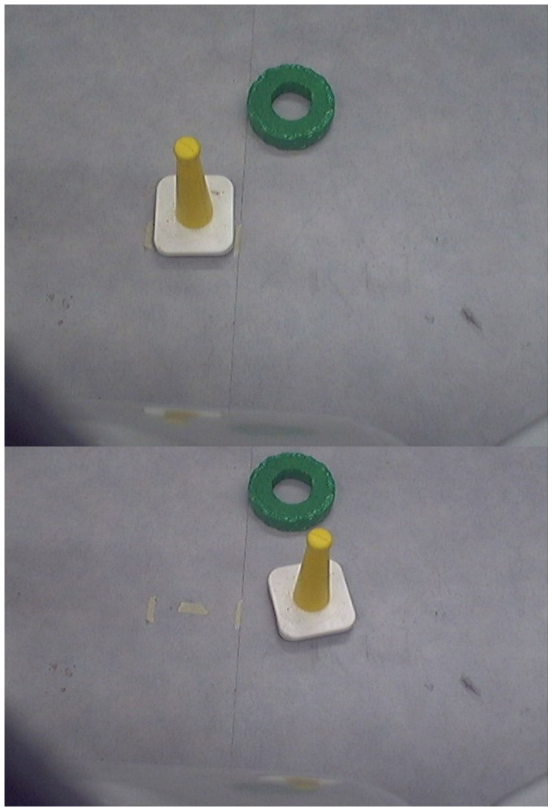
Figure 4. An unsuccessful drop before the push (top) and after the push (bottom). In most of the cases, the disk was far enough away from the peg that the centroids of the blobs in the thresholded images were noticeably separate. In these cases, the distance moved by the peg and by the disk will be different. These drops were labeled as failures.

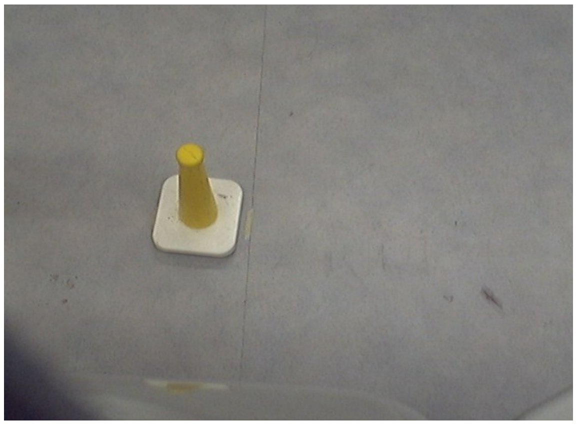
Figure 5. Several of the drops resulted in the disk landing in a way that allowed it to roll off the table completely. In such cases, the program was unable to find a disk in the image, and thus declared the drop to be a failure.

Results:

After applying the analysis method to the data from all the trials, there were only five successful drops out of the fifty-six total trials. As you can see in Figure 6, the successful trials (green markers) are located near the center of the area, and the unsuccessful trials (red markers) are mostly located farther from the center.

There are several red markers located inside the green marker region, which can be attributed to the way the robot released the peg. Because the robot's fingers are made of metal, and the disk had been spray-painted green, the robot's fingers would occasionally stick to the disk briefly when the hand was opening. This caused the disk to fall at an angle on the peg, and was therefore a failure even though the position would have resulted in success.

The distribution of positions forms an interesting shape, as it appears the majority of drop attempts were along a diagonal line. This was most likely a

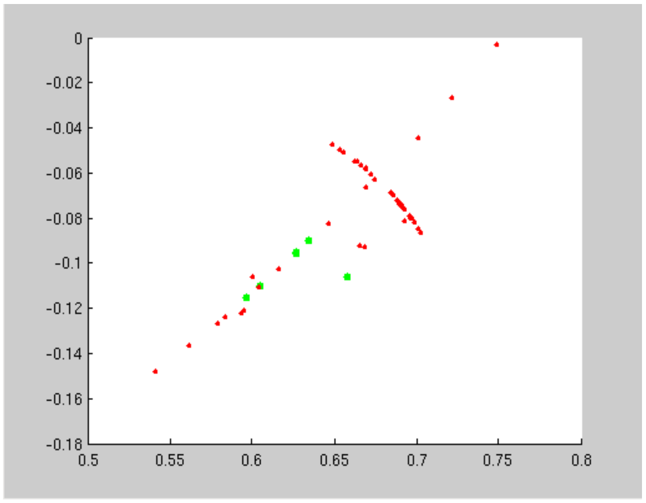

Figure 6. The plot of joint positions during each drop. The green markers indicate a successful drop, and the red circles indicate failure. The axes represent the Cartesian space in which the robot's arm was able to move. Since the robot was not allowed to move it's arm up and down during the random movement, the z-axis is not present.

subtle error in the function used to generate random positions, but the data collected was still valid.

The main reason behind collected the joint positions for each drop is so that, if you were to ask the robot to drop the disk on the peg, it would be able to get the joint position with the highest probability of success by picking a position inside the green marker zone. By updating the map of positions after each additional trial, accuracy can further be improved.

Future Work:

In the future, this experiment could be expanded by instructing the robot to pick a position with the highest probability of success, and drop the peg there. The robot would be able to determine if the drop was successful, and it would update the map of positions to include the new information. Another idea would be to have the robot explore the boundaries in which it it the most uncertain. That is, the regions on the map around the edge of the green marker zone. As Sukhoy *et al.* have shown [1], the uncertain-driven method of exploration is more effective than a random method or the most-certain method.

To improve the methods in the experiment, a more learned approach to analysis could be implemented. Due to time constraints on the project, I hard-coded color threshold values in for the thresholding method to get the information about the disk and the peg. Predictably, this method won't work if you change the color of the disk or peg.

A way to solve this could be to use background subtraction. The robot could take a picture of the empty table, then a picture with the empty peg, and then a picture after the drop. Each time, the background could be subtracted to obtain the 'new' object in the image, and this method would be much more reliable and generalized than hard-coded color thresholding.

Another method to try in the future might be using auditory data to cluster the drops based on the noise the disk makes when it falls. The robot is constantly listening, and the sound of a disk falling on a peg is distinct enough from the sound of a disk dropping on the table or falling on the floor. The robot could perform the trials, then cluster the sounds by success.

The robot could be allowed to listen to someone drop the disk on the peg, and it would be able to determine if the drop was successful or not, depending on how closely the sound fit into one of the clusters.

Bibliography:

1. Sukhoy, V., Sinapov, J., Wu, L., and Stoytchev, A., "*Learning to Press Doorbell Buttons,*" In Proceedings of the 9th IEEE International Conference on Development and Learning (ICDL), Ann Arbor, Michigan, August 18-21, pp. 132-139, 2010.

2. "LHS: Tower of Hanoi Facts." *Lawrence Hall of Science*. Web. 21 Apr. 2011. <http://lawrencehallofscience.org/java/tower/towerhistory.html>.

3. "Robots as Students: Towers of Hanoi." Web. 21 Apr. 2011. <http://www.ros.org/news/2009/10/robots-as-students-towers-of-hanoi.html>.

4. Chang, J., Rubi, N., and Hassavayukul, P. "Towers of Hanoi Final Report". 2007. <http://www.ece.cmu.edu/~ece578/teams/H/>.

5. *OpenCV Wiki*. Web. 21 Apr. 2011. <http://opencv.willowgarage.com/wiki/>.