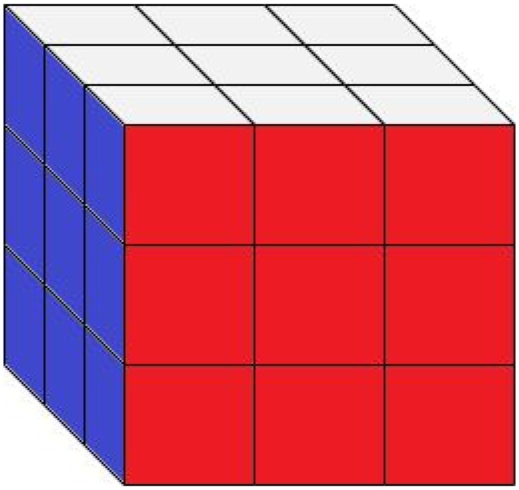
Solving the Rubik’s Cube



David Nelson
CIS667
Semester Project
10/28/2020 (Updated 11/14/2020)

Introduction

This document begins by describing the game, including the state of the game and how it is represented, the goal state, and how to perform actions to transition from one state to the next.

Following the description of the game, this document will go over the structure of the rubik's cube solver program. This will begin with installation instructions and prerequisite software and will then dive into usage of the program for the purpose of simulating a cube that the player can rotate via commands.

The document will then go into describing the rubik's cube solver program in detail. (TODO: Next release)

Terms

Cubie	A single sub-cube of the full rubik's cube. For example, a 3x3x3 rubik's cube contains (3x3x3) = 27 cubies.

The Rubik’s Cube

The Rubik’s cube puzzle is an N-by-N-by-N (N >= 2) cube composed of smaller cubes known as cubies. The player makes an action by rotating a row or column of cubies. The goal of the game is to solve the puzzle, which is to ensure that each face of the Rubik’s cube contains only a single color.

Here is a simple example using a nearly-solved 3x3x3 cube:

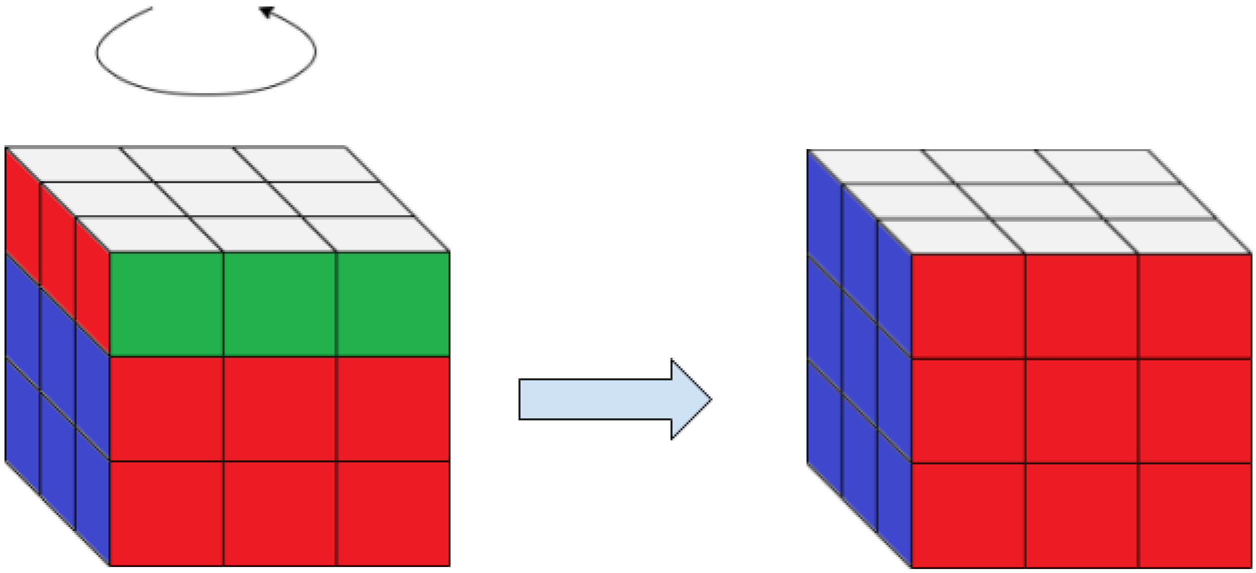


Figure 1: Performing an action and solving the Rubik’s cube.
In this example, the player makes the action of rotating the top row of cubies to the right 90°. This transitions the cube from the state on the left, to the state on the right (the goal state).

The player is allowed to rotate any of the rows or columns of cubies by 90°, 180°, or 270° during the course of the game.

Installation Instructions

Prerequisite Software:
Python interpreter - https://www.python.org/
NumPy - Python vector computing library - https://numpy.org/

If these are already installed then you are ready to run the Rubik's Cube Solver program. If not, you can follow the instructions below to get started using Anaconda which installs Python automatically and offers a simple NumPy installation procedure.

Installing Anaconda

1. Go to https://www.anaconda.com/products/individual
2. Depending on your OS and hardware, choose the appropriate installer and follow the instructions in the installer.

Installing NumPy

1. Start the Anaconda command prompt. (On Windows 10, this can be opened by going to Start -> Anaconda3 -> Anaconda Prompt.)
2. Execute the command "conda install numpy".
    a. If prompted to Proceed with installation, enter "y".

You are now ready to execute the Rubik's Cube program.

Obtaining the Code

Github repo: https://github.com/drnelsoniv/CIS667_rubiks.git

Running the Program

1. Start the Anaconda command prompt. (On Windows 10, this can be opened by going to Start -> Anaconda3 -> Anaconda Prompt.)
2. Execute the program: “py [your_path_here]/rubiks.py”
   Replace [your_path_here] with the path to the Rubik’s Solver script.

Game Setup

rubiks.py accepts interactive input from the user to set up the cube and gameplay.

1. Enter the size of the cube when prompted. This value must be within the range 3-7 inclusive. If no input is given, this value will default to **3**.
2. Enter the game mode when prompted. This value can either be
   a. **0 - Manual gameplay -** This mode allows the user to interact with the cube manually.
   b. **1 - Tree-based AI -** This mode builds a search tree generated by performing valid actions and testing the state of the cube against a game over state (where for each face of the cube, all cubies on that face of the cube are the same color.)

Playing the Game

A player performs an action by rotating one of the rows/columns of the presented cube. This is achieved by entering a valid action.

For a 3x3x3 cube, valid actions include:

Command	Description
0X90	Rotates index 0 (left-most) column of front face from top-to-bottom.
1X90	Rotates index 1 (center) column of front face from top-to-bottom.
2X90	Rotates index 2 (right-most) column of front face from top-to-bottom.
0Y90	Rotates index 0 (bottom) row of front face from left-to-right.
1Y90	Rotates index 1 (center) row of front face from left-to-right.

2Y90	Rotates index 2 (top) row of front face from left-to-right.
0Z90	Rotates index 0 (closest) column of left face from bottom-to-top.
1Z90	Rotates index 1 (center) column of left face from bottom-to-top.
2Z90	Rotates index 2 (furthest) column of left face from bottom-to-top.

(Note: For larger cubes, the valid actions include commands supporting the larger indices to refer to the additional rows/columns. Additionally, 90 represents a 90 degree rotation. Additional rotation amounts are supported, including 180, 270, -90, -180, and -270. If the rotation is negative, this symbolizes rotating the row/column in the opposite direction as described above.)

Notes to the Professor

These are included to assist with grading:

- Pieces of the cost functionality that you requested have been implemented, but no accounting is being done at the present time, so nothing is printed to you. This is the next item on my to-do list.
- The tree-based search is doing a breadth-first search over all possible states, and the search space is huge. As a result, I've intentionally dumbed down the script to only generate simple-to-solve cubes. I'm going to give some thought to this to try to make the search space much more manageable.

Bibliography

Anaconda, Inc. (2020). Anaconda3 (Version 2020.07) [Computer software]. https://www.anaconda.com/

Various. (2020). NumPy (Version 1.18.5) [Computer software]. https://numpy.org/

Zelle, John. (08/26/2016). graphics.py (Version 5) [Computer software] https://mcsp.wartburg.edu/zelle/python/graphics.py