A Level Programming Project:

(3x3) Rubik’s Cube Solver

By Hamza Salman

Problem Identification:

Though the 3x3 Rubik’s Cube is one of the most well-known toys and puzzles, with 350 million of these cube’s being sold, only around 20 million people are able to solve the cube by closely following a published method, with only 1 million people having actually memorized a published method to solving the Rubik’s Cube.

Therefore, this program is a Rubik’s Cube Solver, with the purpose of providing people who do not know how to solve a 3x3 Rubik’s Cube, easy to follow steps, to allow them to solve their (shuffled) Rubik’s Cube. The method of solving the cube is based on an easy method for solving a cube: The Layer by Layer Method.

My program will implement two different modes in solving the cube:
One with more optimized steps, to allow people to solve Rubik’s Cubes quickly. This will be for people who do not necessarily wish to learn the algorithms behind solving the cube, but rather wish to hold a completed cube only.
Another mode with little to no optimized steps to help teach beginners how to solve a Rubik’s Cube. This mode will clearly show the logic in solving the cube using this method, and thus, the algorithms being used will be displayed to the user, so the user can learn the basics to reading Rubik’s Cube Algorithms, and more importantly, which algorithm to use when.

I will also implement a way to save the cube’s state (in case somebody wishes to come back to their cube later): loading a saved cube will be much quicker than setting a new cube to solve. This would most likely be in unencrypted xml format, since there is no sensitive data involved and this format will also increase the user ability, for those who wish to directly edit the cube state from the xml file, making the program more flexible to the end users.

Computationally Solvable and why this program is a good solution:

The best current method to learn how solve a Rubik’s Cube, is to go onto YouTube and to learn how to solve the Rubik’s Cube from observing what the people do to solve the Rubik’s Cube, or actually having an individual, who knows how to solve the cube, teach them.
However, most people do not have access to anyone, who is willing to teach them how to solve the cube. And as for YouTube videos, even though this method of learning is largely effective, it is still extremely difficult to account for all issues and situations that beginners may run into, in just a video tutorial.
So, my program will expand on what the YouTube videos cover, by showing graphically, interactively and step by step, how to solve the Rubik’s Cube. My program may also allow learners to better understand how to use this algorithm to solve the cube, giving them the option to memorize the algorithms to solve the cube via the layer by layer method.

Stakeholders:

The main stakeholders to this program will include the people who have a Rubik’s Cube, and wish to solve their cubes. These people would use the program to solve the Rubik’s Cube both easily, and reasonably quickly (below 5 minutes).
However, another slightly different stakeholder would be beginners who want to learn how to solve the Rubik’s Cube, and memorize the method too. These people would use the program to learn the algorithms and logic being used, so they can eventually solve their cubes without use of the program.

Features:

There will be a 3D graphical cube, which will allow users to view the virtual on-screen cube from any side of the cube they wish, improving the YouTube video tutorials' problems of not being able to freely see all sides of specific Rubik's Cubes.

Completing set algorithms are a big strength of computers, and algorithms are an integral part of solving Rubik's Cubes. Also, computers will be able to spot optimizations in the steps required to solve the Rubik's Cube, and be able to shorten the required steps to solve the cube (, for people who wish to solve the cube without learning any algorithms).

There must also be a way of inputting the Rubik's Cube tile information, which can be done by displaying a net of the cube, and then allowing users to switch the colours of each tile, by repeatedly clicking on a tile, until the colour of the tile is correct. Although this involves a lot of clicking, it does achieve the goal, without use of a menu. Using a menu, requires moving the mouse to set the colour for each tile, which is more difficult and will take longer than simply clicking until the correct colour is set.

Limitations:

This program will only be used on desktop only. Coding a mobile interface may be difficult, considering that it will not be as easy to input data from phone, as it would be from a desktop. Also, viewing the cube on phone would require using a hand to move the cube on-screen, which would mean not having two hands on the cube (which may make it difficult to turn slices on the cube), so this functionality would probably not be implemented.
The program must be optimized, so that the program can run as smoothly as possible, so users of the program do not spend too long solving their cubes (as this ruin user satisfactions). Therefore, the number of turns available should be executed within 5 minutes, by the user with reasonable speed and familiarity with the program.
The data saved must also be optimized, if there is any, as this program will contain a lot of data, all of which does not have to be stored. Using the methodical nature of the program, some data can be omitted, like state of the cube at each turn. Instead, only the starting state, and the turns required to solve the cube should be saved.

Requirements:

The desktop (or laptop) being used, must have a functioning graphics card, as the program will utilize the OpenTK module, which makes use of graphics cards.
The desktop (or laptop) must have at least 100MB of RAM available for the program, for proper execution.
The desktop (or laptop) must also have a functioning mousepad, or mouse, as the main way of viewing the cube will involve using the mouse to scroll around the cube.