Self-Solving Rubik's Cube

DESIGN DOCUMENT

Team 29
Dr. Zambreno
Dr. Zambreno
Taylor Burton
Jacob Campen
Casey Cierzan
Joe Crowley
Annie Lee
Keegan Levings-Curry
Luke Schoeberle
sdmay20-29@iastate.edu
Team Website

Revised: October 6, 2019/Version 1

Executive Summary

Development Standards & Practices Used

- Follow IEEE standards
- Push early and push often
- Document every part of the process

Summary of Requirements

- Self-contained within the cube
- Easily scrambled by hand
- Solved within two minutes
- Lasts for three or more years
- Charges from an external source
- Fits within the typical Senior Design budget

Applicable Courses from the Iowa State University Curriculum

- ComS 309
- CprE 288
- ComS 228
- EE 224
- EE 311
- Phys 222

New Skills/Knowledge acquired that was not taught in courses

- Motors
- Clutches
- Rubik's cube algorithms
- 3D printing
- CAD tools

Table of Contents

List of figures/tables/symbols/definitions (This should be the similar to the project plan)

1 Introduction

1.1 Acknowledgement

We would like to thank Dr. Joseph Zambreno and Lee Harker for their support.

1.2 Problem and Project Statement

Our project is to create a self-solving Rubik's cube by adding components within a cube. After the user scrambles the cube by hand, the cube must solve itself within two minutes.

This cube is designed to inspire prospective students to pursue Electrical and Computer Engineering (ECpE). By witnessing the potential of an ECpE degree, they will become motivated to study ECpE at Iowa State University (ISU).

1.3 Operational Environment

The finished cube must operate indoors on a fixed display table at room temperature.

1.4 Requirements

Here is a list of requirements:

- Can be scrambled by hand without requiring much force from the user
- Contains no external components (e.g. cameras, robot arms) except for charging devices
- Reliably solves the cube within two minutes
- Durable enough to last for at least three years
- The entire budget cannot exceed $750

1.5 Intended Users and Uses

Our cube's primary users are ISU tour guides and prospective families.

The tour guides will maintain the cube, which includes cleaning, charging, and lubing.

The prospective families will scramble the cube, which may include high school students, parents, and small children.

1.6 Assumptions and Limitations

Here is a list of assumptions:

- The cube will always begin in a solved state
- Users will not need to indicate that they have finished scrambling the cube
- Users will only make turns in 90°-increments (e.g. 0°, 90°, 180°)

Limitations:

- The cube must be solved within two minutes
- The cost for the cube cannot exceed $750
- The cube's side length must be in the range [5.7, 18] (in cm)

1.7 Expected End Product and Deliverables

Our end product is a self-contained, self-solving Rubik's cube. The final cube and all prototypes will be delivered to Dr. Joseph Zambreno for future tours. We will also provide a charging cord and an adapter with the cube. Additionally, we will provide our source code to Dr. Zambreno for future enhancements of this project. All of these deliverables will be submitted by Friday, May 8th.

2. Specifications and Analysis

2.1 Proposed Design

Our cube's side length must between 5.7 cm and 18 cm, and the cube will have a standard color scheme and layout. Each face will be independently controlled by electric motors. Since the cube must be easily scrambled by hand, a clutch will control when the motors are engaged, which will ultimately ease scrambling and increase the motors' lifespan. To sense face movement, two Hall Effect sensors under each face will be paired with magnets within each face. When a turn occurs, the sensors will transmit meaningful analog voltages to the microcontroller, which will be interpreted by embedded software on the microcontroller.

The software will adhere to IEEE Standards. The software will control the clutches, ensuring that the motors are disengaged during the scrambling process. It will also track the cube's physical rotations during the scramble, and it will simulate the rotations in software. After a period of inactivity, the software will initiate a solving algorithm based on the simulated cube's state. In doing so, it will send turn instructions to the motors until the cube is solved. By using algorithms that require few moves, the software will be able to solve the cube in less than two minutes.

At this point, the sensors have been tested and the software is in the early development stages.

2.2 Design Analysis

As of 10/1, we have researched previous solutions, discussed multiple designs, and started to create our first prototype. Initially, we are using a large Rubik's cube for prototyping, and we have tested a few small magnets and Hall Effect sensors. We noticed that we may need different magnets since it is difficult to use the small magnets. Ultimately, we hope that we can use four magnets per face to capture the cube's rotation during the scrambling process. If we can do so, we will be able to keep every component within the Rubik's cube, as desired by our client.

2.3 Development Process

For our project, we will be using a modified Agile workflow. We begin with the typical waterfall steps of high-level requirements specifications and design. However, we then follow an iterative Agile process of development, which will incrementally push us towards our final solution. By frequently displaying our progress to our client, we will have a better understanding of our client's needs and desires, so we will ultimately create a more suitable final product.

For more details, see Figure 1 below:

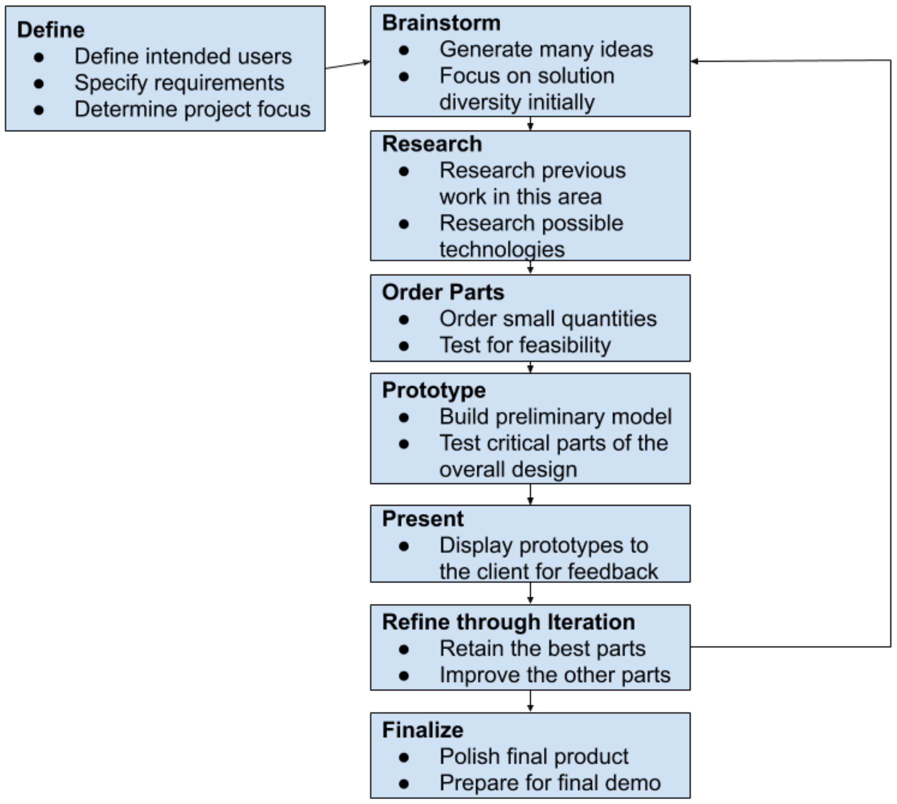

Figure 1: Development Process

2.4 Design Plan

We will initially design a large Rubik's cube that meets the basic requirements. Although a large cube may satisfy the requirements, it will not be ideal because larger cubes are more difficult to scramble. Once we have a functional larger cube, we will try to make a smaller cube by reducing the size of our internal components as much as possible.

3. Statement of Work

3.1 Previous Work And Literature

Similar existing products provided by client:

1) Computer vision based:

Advantage: Fixed position, easier to identify colors and status of cube. External force to spin cube.

Shortcoming: Not very interactive.

2) Self-rotating cubes:

Advantage: More engagement with audience.

Shortcoming: Difficult to identify cube orientation.

Solving algorithms:

1) Petrus: Best algorithm, fast but complicated.

2) Thistlethwaite: Performance and time are both average.

3) Old time Pochmann: Easiest to implement but time consuming.

Other resources:

1) Design of self-rotating cube: Started from implementation of a large cube and sized-down to regular sized cube. Uses a specialized cube that has a larger inner space.

2) Open Source Java Applet: Multiple algorithm implementation and visualization. Open sourced, so this can be used in our project implementation. May need to rewrite code from Java to C.

3.2 Technology Considerations

Highlight the strengths, weakness, and trade-offs made in technology available.

Discuss possible solutions and design alternatives

3.3 Task Decomposition

Design

This step is the preliminary design on paper. All hardware and much of the software design comes from this component.

Component tests

Once we have the required sensors and motors we will test them to ensure that they work as expected and that they are sized appropriately, have the required durability, etc. This will lead directly to the hardware prototype.

Hardware prototype

Once we have components that meet our requirements, we can begin putting them together into our first prototype. The initial prototype need not work in conjunction with the software. This leads directly to the combined prototype.

Software

This section is the code that receives the turn data and analyzes it to determine the next step to solving the cube based upon the chosen algorithm. This ties directly into the combined prototype.

Combined Prototype

This section ties together the software and the hardware prototype to create a full-fledged prototype. The goal of the prototype is the test all components in conjunction with each other and flush out any issues before miniaturizing and finalizing with the final cube.

Final Cube

The final cube is the end product that will be submitted at the end of the spring semester. If it is not up to scratch, once miniaturized, we can revert back to the combined prototype stage but in the smaller model.

3.4 Possible Risks And Risk Management

Because we are limited by size and useable space, the majority of our risks are in the realm of sizing of components and their accuracy. Additionally, as time goes on and if sizing becomes and issue, cost management for the materials could begin to creep up on us. To mitigate the risk of costs, we will take meticulous records of purchases and remaining funds so as to not exceed our budget.

3.5 Project Proposed Milestones and Evaluation Criteria

To keep our project on pace, we have set a few milestones, divided between the different sections. Namely:

Algorithm implementation

When the algorithm is coded, we will be able to test it with our virtual cube without needing to connect it to the physical cube.

Cube hardware

When the motors and sensors are set up properly, we will be able to set up the cube such that we can test the motors by having them rotate a face the desired number of turns. We will also have it turn many times in the same directions to test the wires and ensure that they do not tangle.

3.6 Project Tracking Procedures

In order to track and gauge our progress on the project over the course of the next two semesters, we will keep up with our scheduled milestones and reevaluate our progress if we fall short.

3.7 Expected Results and Validation

Our desired outcome is a self-contained, self-solving Rubik's cube. We will be able to know that our solution works if a user is able to pick up the cube, scramble it, set it down on an arbitrary flat surface, and the cube solves itself within 2 minutes.

Our stretch goal is to fit our components in an actual Rubik's cube, and to have the solve be a minimal number of turns. Once we are able to reach the above goal with a larger cube, we will be doing the same with a smaller cube, with potentially new internals.

4. Project Timeline, Estimated Resources, and Challenges

4.1 Project Timeline

The Gantt chart below is a tentative schedule of project. With week 1 as the first week of Fall 2019 semester, excluding thanksgiving, winter, and spring break. The timeline is being proposed based on the tasks to be done and order of completion .

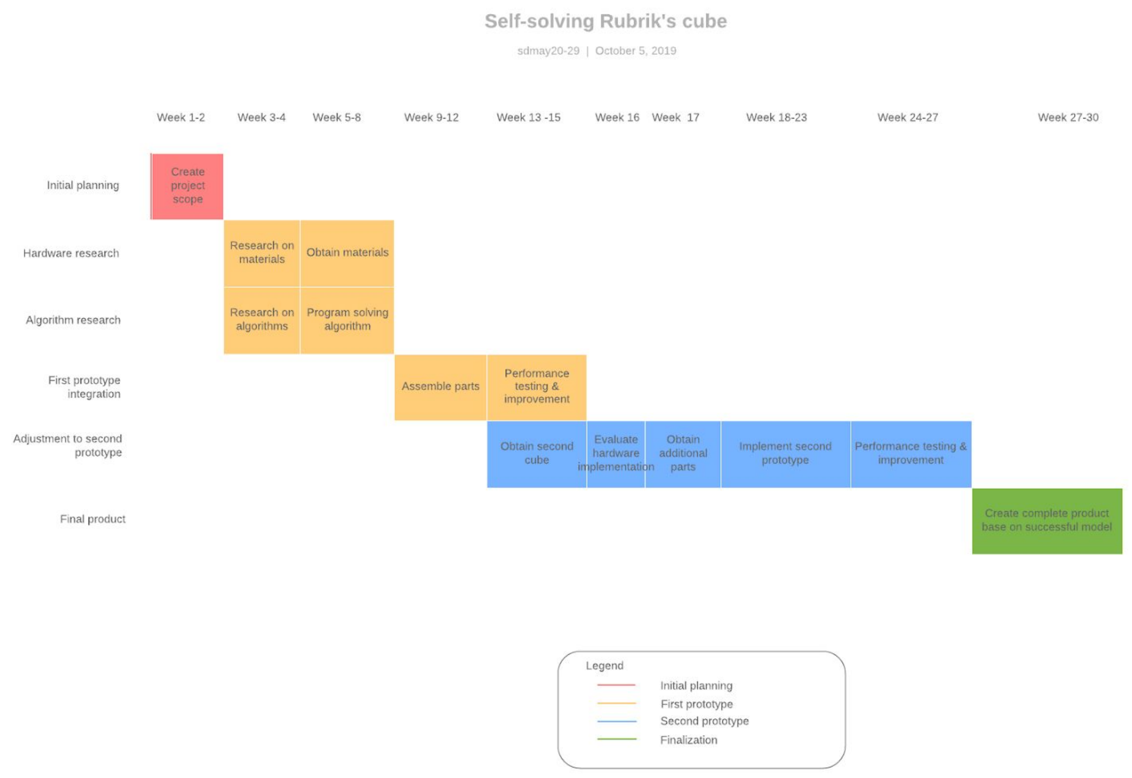

Figure 2: Project Timeline

4.2 Feasibility Assessment

Our project will be creating a self-solving Rubik's cube from scratch. It will either be a regular sized Rubik's cube or bigger, depending on space requirements for our internal components. The largest challenge will be our team's lack of knowledge on motor systems, and what we will need to use to have our sides be able to be freely turned, and then turned by some motor. We plan to use a clutch system, but again, we need to learn more before we are able to finalize our design.

4.3 Personnel Effort Requirements

Include a detailed estimate in the form of a table accompanied by a textual reference and explanation. This estimate shall be done on a task-by-task basis and should be based on the projected effort required to perform the task correctly and not just “X” hours per week for the number of weeks that the task is active

4.4 Other Resource Requirements

(Parts will be added from our materials list once we know which ones will be used for our final product)

4.5 Financial Requirements

(Budget total will be listed here once project is completed)

5. Testing and Implementation

Testing is an **extremely** important component of most projects, whether it involves a circuit, a process, or a software library

Although the tooling is usually significantly different, the testing process is typically quite similar regardless of CprE, EE, or SE themed project:

1. Define the needed types of tests (unit testing for modules, integrity testing for interfaces, user-study for functional and non-functional requirements)
2. Define the individual items to be tested
3. Define, design, and develop the actual test cases
4. Determine the anticipated test results for each test case
5. Perform the actual tests
6. Evaluate the actual test results
7. Make the necessary changes to the product being tested
8. Perform any necessary retesting
9. Document the entire testing process and its results

Include Functional and Non-Functional Testing, Modeling and Simulations, challenges you’ve determined.

5.1 Interface Specifications

– Hall Effect Sensors

The Hall Effect sensors will transmit an analog signal to the microcontroller. This signal carries the position of the cube, which will have to be tracked in software. Total of 12.

– Clutches

The motors will be connected via electrically actuated clutch, of which the static state is engaged. This means the software will have to detect movement and disengage the clutch. 6 total.

5.2 Hardware and software

Hardware

- Full range of motion
- Repeated scramble and unscramble possible
- Time test, withstand repeated use

Software

- Unit tests for cases of unsolved cubes
- Be able to record a scramble of upwards of 100 moves
- Be able to translate scramble rotations into an unsolved cube in the program's memory

5.3 Functional Testing

Hardware

- Rotations must be 90 degrees
- Clutch must disengage during a scramble
- Clutch must engage during a solve
- Power supply must last through a solve
- All wired connections must stay connected during rotations

Software

- Given an unsolved cube, must be able to produce a list of rotations required to lead to a solved cube
- Must be able to send turn signals in an appropriate manner

Non-Functional Testing

Performance: Our cube must be durable, and last for a couple years. During a solve, it must not drain the battery fully and stop midsolve. Our cube must solve itself in 2 minutes.

Security: Our cube must not have sharp edges and must not shock a user if they touch the charging port.

Usability: Our cube must be able to be held effortlessly so that our end user can scramble. For the scramble, our sides need to turn easily without grinding the gears that will turn the sides during a solve.

Compatibility: We will need a power supply cable that can be plugged into a standard outlet.

5.4 Process

Our testing process will follow the same scheme as our design process. We start by defining what needs to be tested, and what our expected outcome should be able to do. We then brainstorm how to test for this quality, and then we jump to prototyping our testing environment. We will present the findings of our tests to each other first, in order to ensure that a good testing process was followed. Finally, we will refine our product through the results of our testing.

5.5 Results

- We had success in testing our Hall Effect sensors with our ordered magnets, and we are able to visualize how the control unit will fit in the middle of the cube.
- Our first setback is figuring out the motors and components needed to actually rotate our sides, and our lack of motor knowledge will be a hurdle to overcome.

-**Modeling and Simulation**: This could be logic analyzation, waveform outputs, block testing. 3D model renders, modeling graphs.

- To model our end product, we may use CAD software to see how it all fits together before buying all our components.
- To test our solving algorithms, we will be writing unit tests with different scrambles of a Rubik's Cube.

6. Closing Material

6.1 Conclusion

In conclusion, we have completed the initial design steps, and we have started to create prototypes. Currently, we plan on using magnets, Hall effect sensors, and stepper motors in our prototype. We plan to present this prototype to our client by the end of the semester.

6.2 References

Flatland, Jay. "World's Fastest Rubik's Cube Solving Robot - Now Official Record Is 0.900 Seconds." YouTube, YouTube, 11 Jan. 2016, https://www.youtube.com/watch?v=ixTddQQ2Hs4.

Controller, Human. "Self Solving Rubik's Cube." YouTube, YouTube, 17 Sept. 2018, https://www.youtube.com/watch?v=xCoH2AORcEQ.

"Petrus Method." Petrus Method - Speedsolving.com Wiki, https://www.speedsolving.com/wiki/index.php/Petrus_Method.

"Thistlethwaite's Algorithm." Thistlethwaite's Algorithm - Speedsolving.com Wiki, https://www.speedsolving.com/wiki/index.php/Thistlethwaite's_algorithm.

"全自動ルービックキューブ　Self Solving Rubik's Cube by Human Controller." DMM.make, https://media.dmm-make.com/item/4462/.

"Blind Solving Algorithms." SpeedCubeReview.com, https://www.speedcubereview.com/blind-solving-algorithms.html.

"Rubik's Cube Java Applet Program." Rubik's Cube Java Applet Program, https://ruwix.com/the-rubiks-cube/rubiks-cube-java-applet-software-program-josef-jelinek-animating-rubix/.

6.3 Appendices

We do not currently have any appendices.