A decorative graphic on the left side of the slide consists of a 3x3 grid of colored squares. The colors are: top row (salmon, lime green, orange), middle row (cyan, orange, light green), and bottom row (lime green, light green, salmon).

# Solving Techniques for The Rubik's Cube in Software and Robotics Environments

A small red square icon.

Supervisor


Lect. PhD Mihai-Simion Nechita

Author

Mircea Măierean

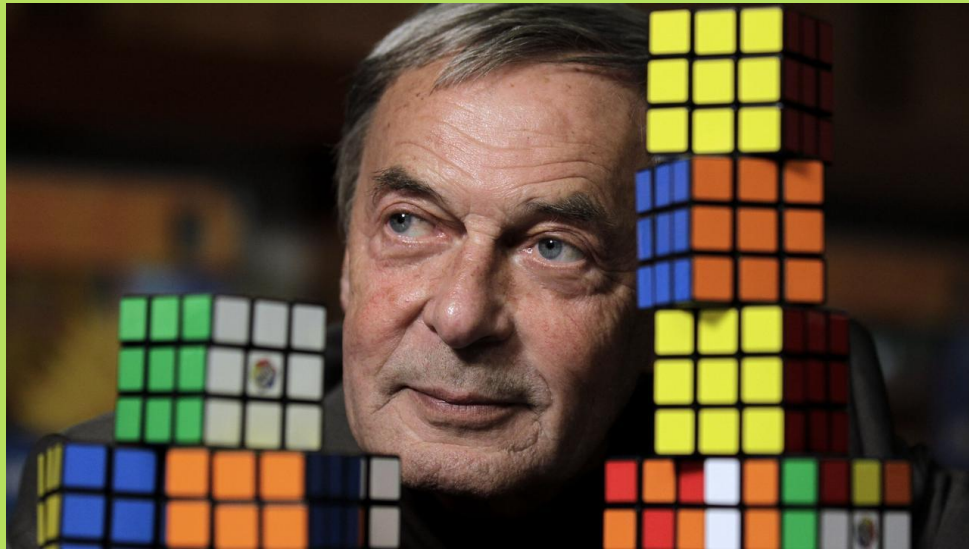


# 1) Introduction



*The Cube is, at the same time, a symbol of  
simplicity and complexity – Erno Rubik*

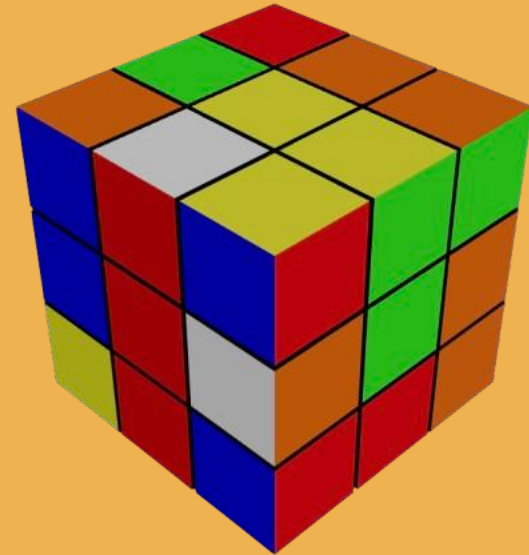
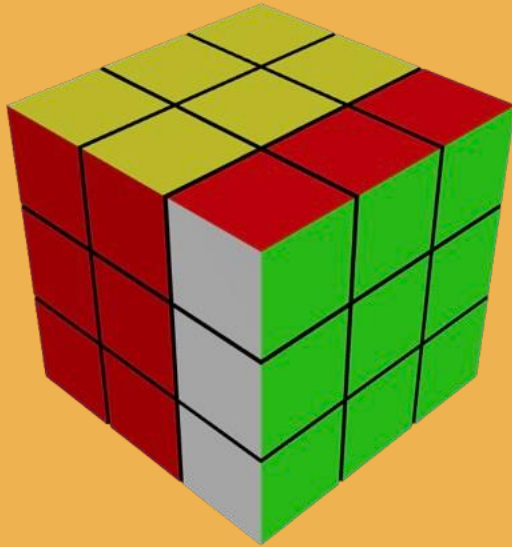
# Context



*Erno Rubik, creator of the puzzle*

- Invented in 1974, the Rubik's Cube is the **World's Best Selling** 3D Puzzle
- In total, there are **43 quintillion** possible permutations
- Despite its popularity, it represents an **impossible** task for many individuals
- **How** can we make the Rubik's Cube more approachable for non-experts?

# Can you solve?



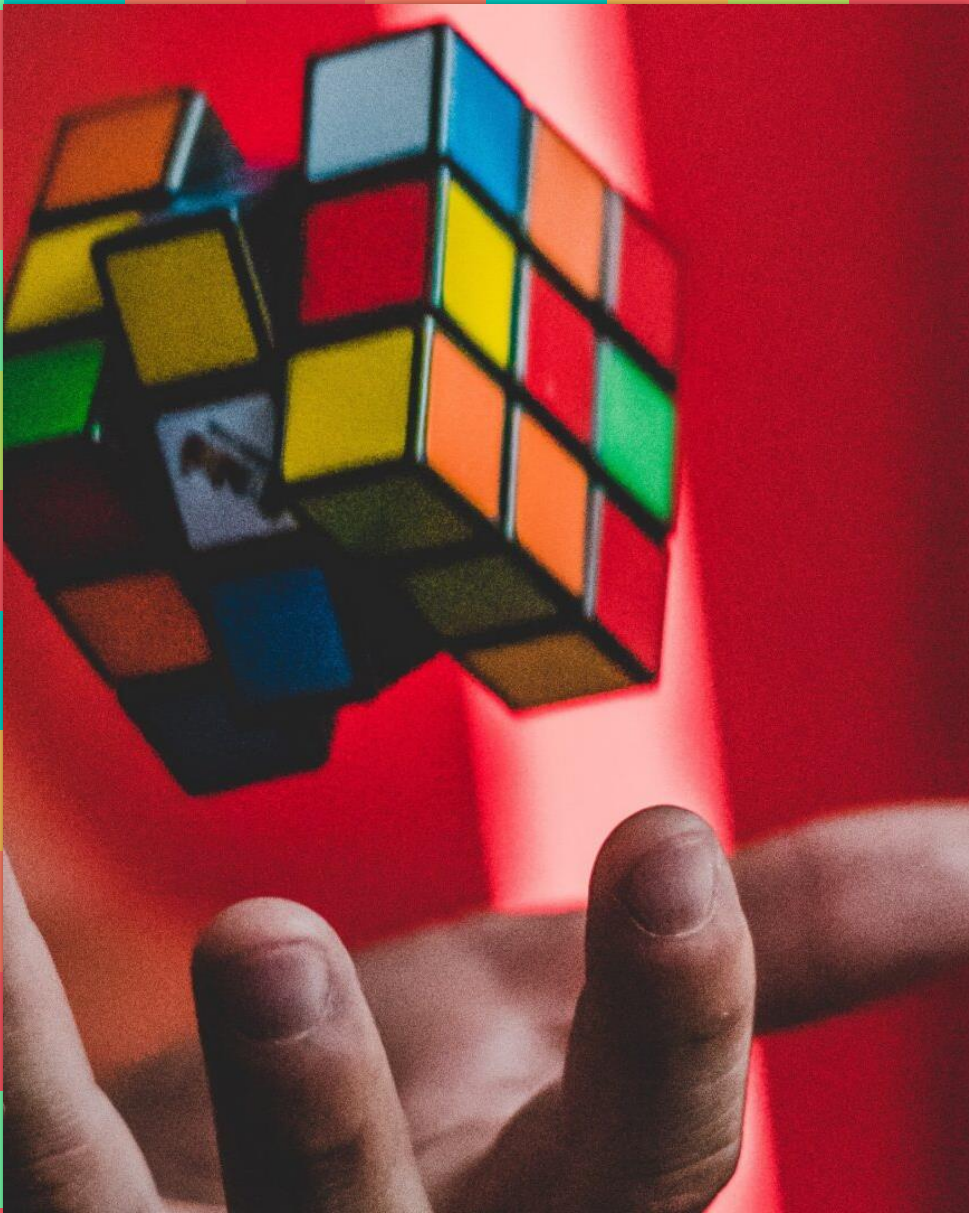


*Me, when I was a kid*

## Motivation



- ◎ Many beginners **struggle** to solve the cubes
- ◎ Despite the existence of tutorials, **nothing** is tailored for specific scrambles
- ◎ **Interdisciplinary areas** are unexplored by young people

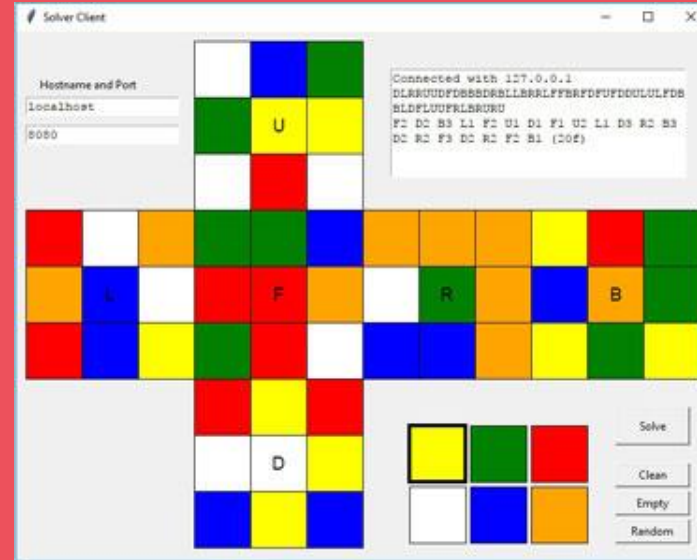




## 2) Objectives

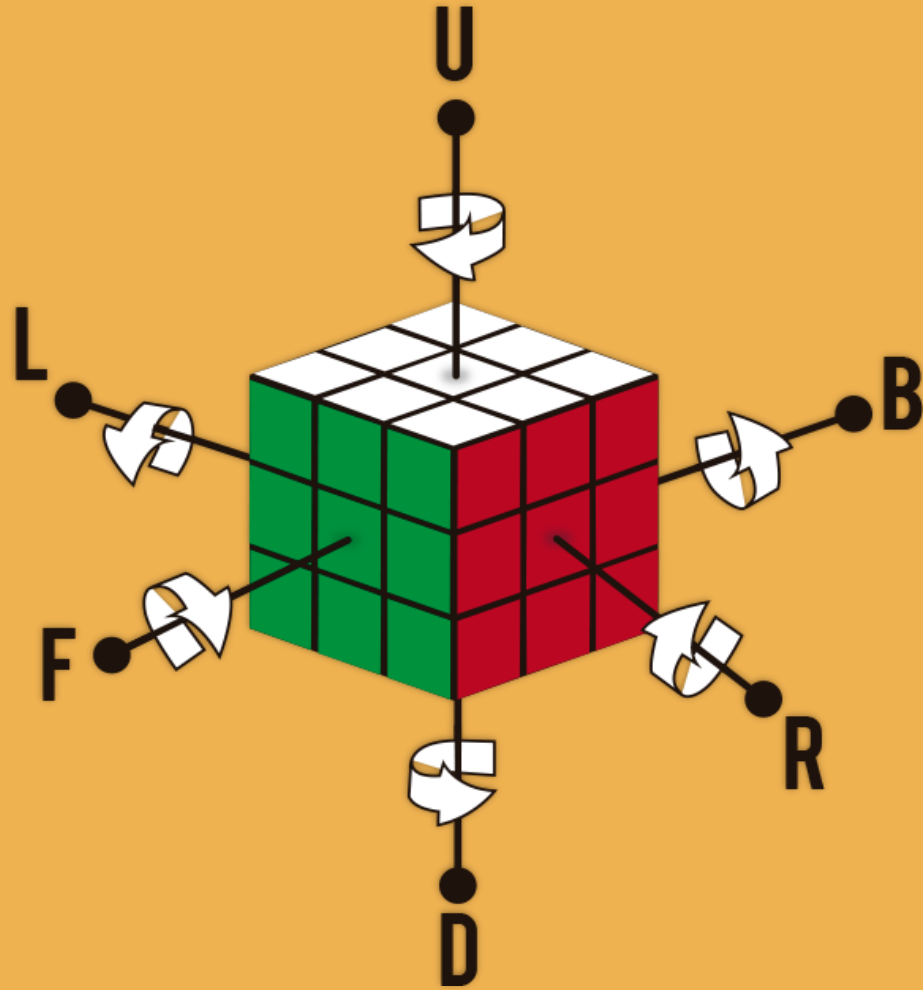
*I'm wondering how people are so creative, and how many things were born out of and inspired by the Cube – Erno Rubik*

- 
- 
- ③ Develop a web-based application that provides step-by-step instructions to solve **any Rubik's Cube configuration**
  - ③ Integrate a **color detection** service to automatically identify and virtualize the cube's current state
  - ③ Offer an **optimized** solving path as an alternative to standard human methods
  - Design and build a functional **robot** capable of solving the Rubik's Cube autonomously



## Related Work





### 3) How to solve a Rubik's Cube?

*If you are curious, you'll find the puzzles around you. If you are determined, you will solve them – Erno Rubik*

- All the methods rely on **moves** on each face of the cube, defined by the David Singmaster
- A sequence of moves is defined as an **algorithm**
- Any existing solving techniques is broken down into steps, each one with **specific** algorithms

# Methods for solving the cube

## Human Solutions

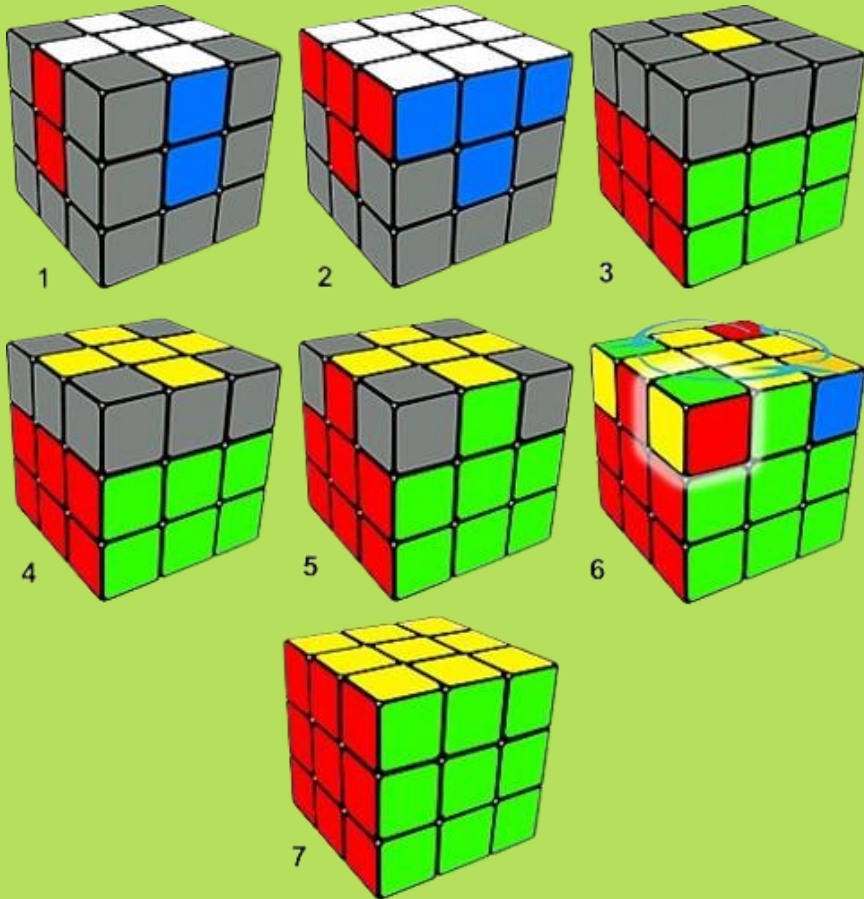
- Layer by layer method – Beginners
- Roux Method – Intermediate
- Petrus Method - Advanced
- CFOP – Advanced
- ZZ Method - Advanced

## Computational Solutions

- Kociemba's Algorithm
- Thistlethwaite's Algorithm

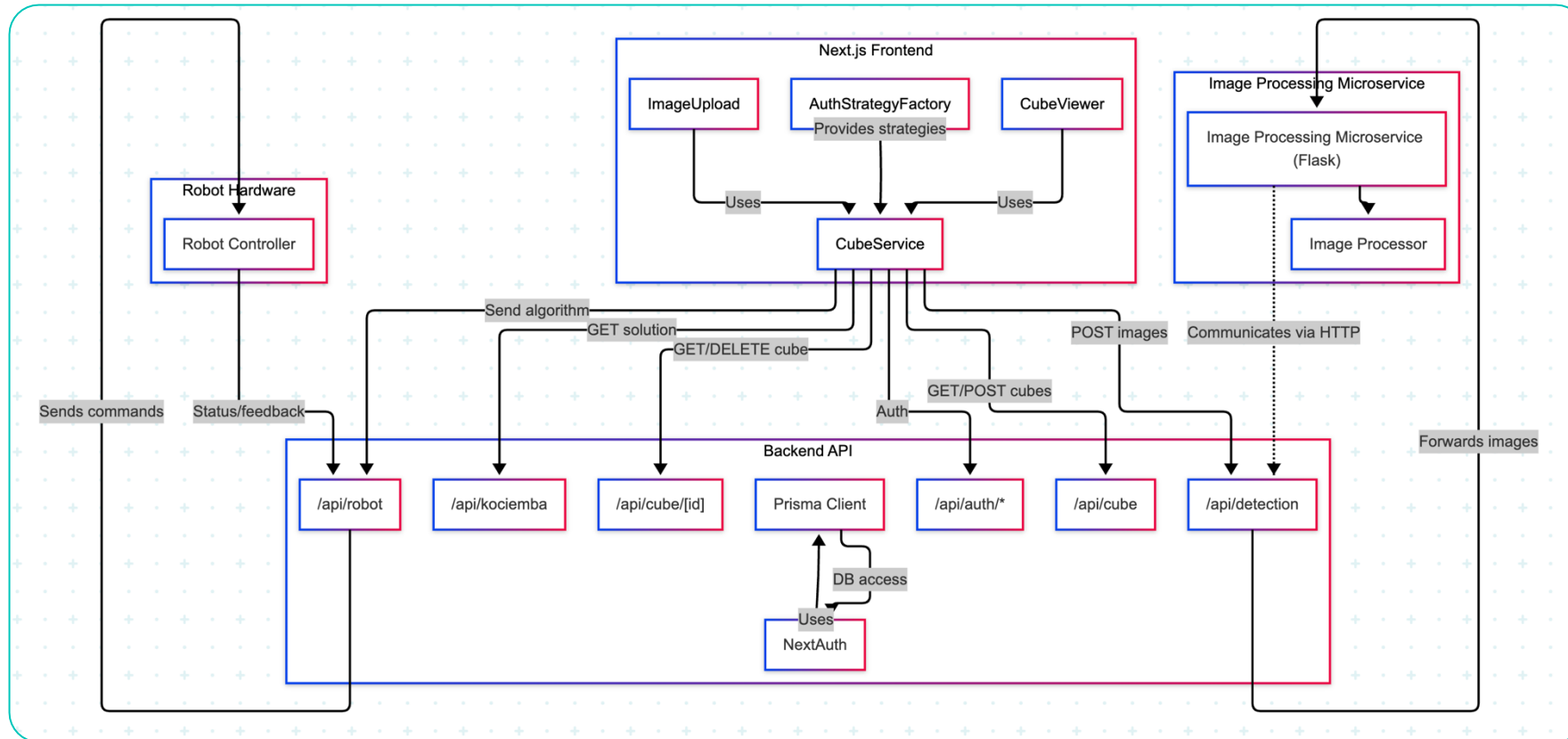
# Layer by layer method

- Most popular existing method, **starting point** for speedcubers
- Consists of 7 steps:
- Step 1: **White** Cross
- Step 2: **White** Corners
- Step 3: **Middle layer**
- Step 4: **Yellow** Cross
- Step 5: **Yellow** Corners
- Step 6: Permute **Yellow** Corners
- Step 7: Permute **Yellow** Edges
- The solutions provided in the app are based on this method



# Kociemba's Algorithm

- Widely-used fast **optimal solver** in cube apps
- Runs in **2 phases ( $\leq 20$  moves total)**
- Let  $G1$  be a subset =  $\langle U, D, R2, L2, F2, B2 \rangle$ .
- In phase 1, the algorithm looks for maneuvers which will transform a scrambled cube to  $G1$  using **iterative deepening A\*** with a lower bound heuristic function  $h1$  (IDA\*).
- The heuristic function  $h1$  is a memory-based lookup table and allows **pruning** up to 12 moves in advance.
- In phase 2 the algorithm **restores** the cube in the subgroup  $G1$ , using only moves of this subgroup.



## 4) Web application

*I'm calling the Cube, it's a piece of art. At the same time, it's an intellectual task as well – Erno Rubik*



# User Interface: Next.js

## Frontend

- Provides a menu for the user where it can upload images with the **scramble** of the cube.
- The cube has an internal state, that can be rendered either as a **2D** or a **3D representation**
- User has the option to select the method for solving the cube (**layer by layer, Kociemba**)
- User has the option to solve a specific step, make a specific movement, and all the applied algorithms are displayed on a history panel

## Backend

- Interacts with the database for saving state of the cube
- Manages user registration and authentication
- Sends photos to the color detection microservice and handles the response
- Acts as a main point of interaction with the robot

# Cube

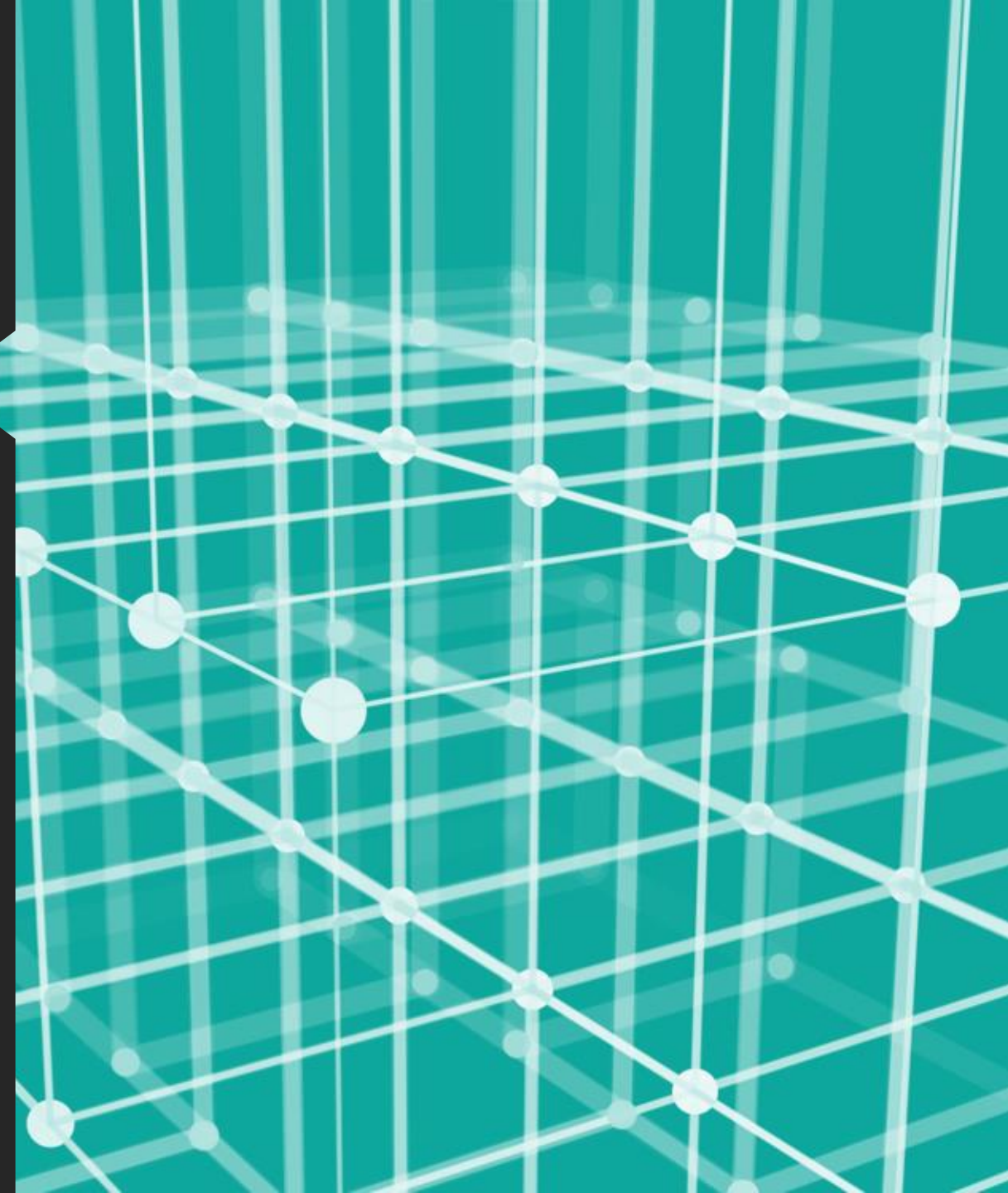
- Cube is represented as a structured data object.
- Each face and sticker is stored programmatically
- Updates state after every move or algorithm
- Centralizes all cube logic and state management
- Lives in the app as a unique object that gets passed around when a move is made

# Moves & Algorithms

- Abstract Move class defines the structure for all cube moves, and has the abstract methods `execute` and `undo`
- Each move (e.g., RMove, BMove, FMove) inherits from Move, and overrides the function `execute` and `undo` to change the cube as intended => Command pattern
- Algorithm class represents a sequence of moves as an algorithm, while still extending Move class => Composite pattern
- Centralizes logic for handling and applying move sequences

# Solver service

- Implements step-by-step functions for solving the Rubik's Cube (e.g., white cross, white corners, middle edges, **yellow** cross, **yellow** corners)
- Each function takes a **RubiksCube** object and analyzes its current state
- Determines the required moves to progress toward the solved state
- Uses helper functions to identify and manipulate specific pieces (edges, corners).



# Cube microservice: Flask

## Color detection endpoint

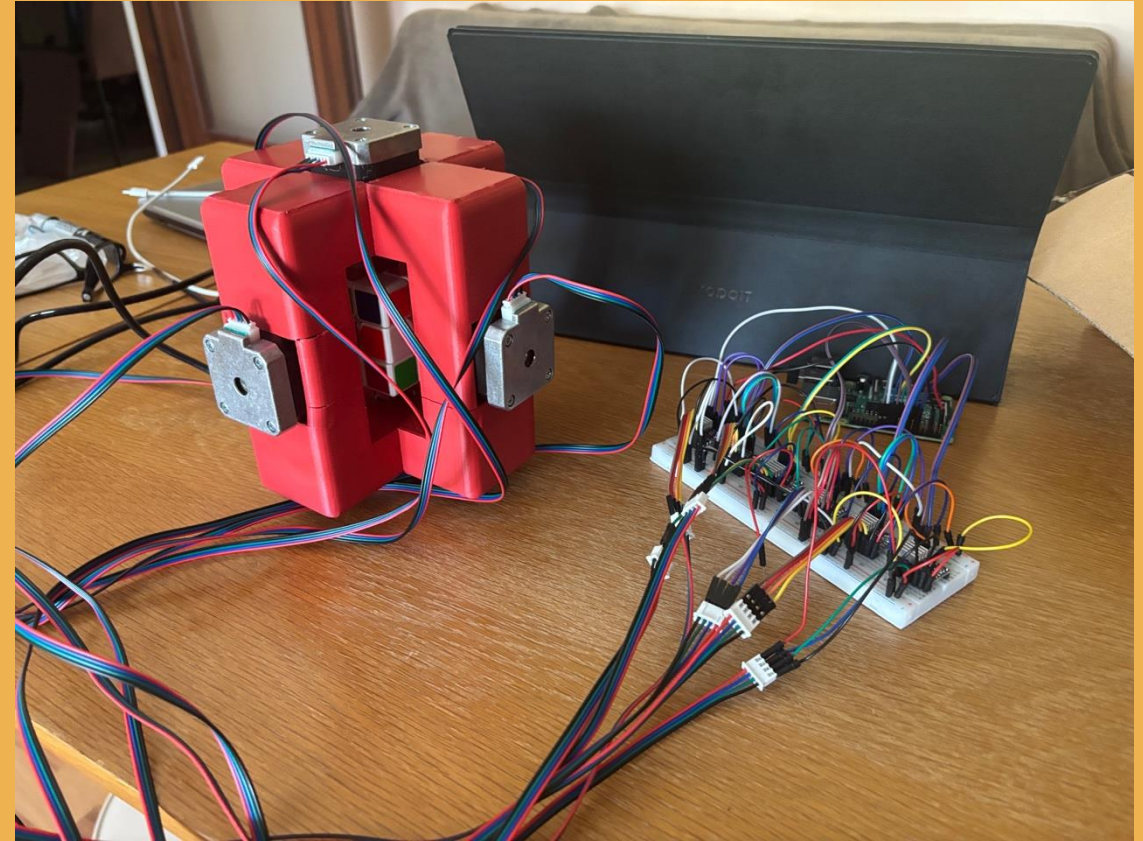
- Images received are analyzed and converted in the HSV space using built-in OpenCV functionalities
- Each image has a pre-defined region of interest from where the dominant values are extracted
- Each value falls in a specific interval that correspond to a single colour

## Fastest solution endpoint

- Computes the fastest solution for a scramble using RubikTwoPhase, that is a library developed by Herbert Kociemba
- Upon its first run, it computes the pruning tables used for the heuristic function
- Returns an algorithm that corresponds to the quickest determinable solution

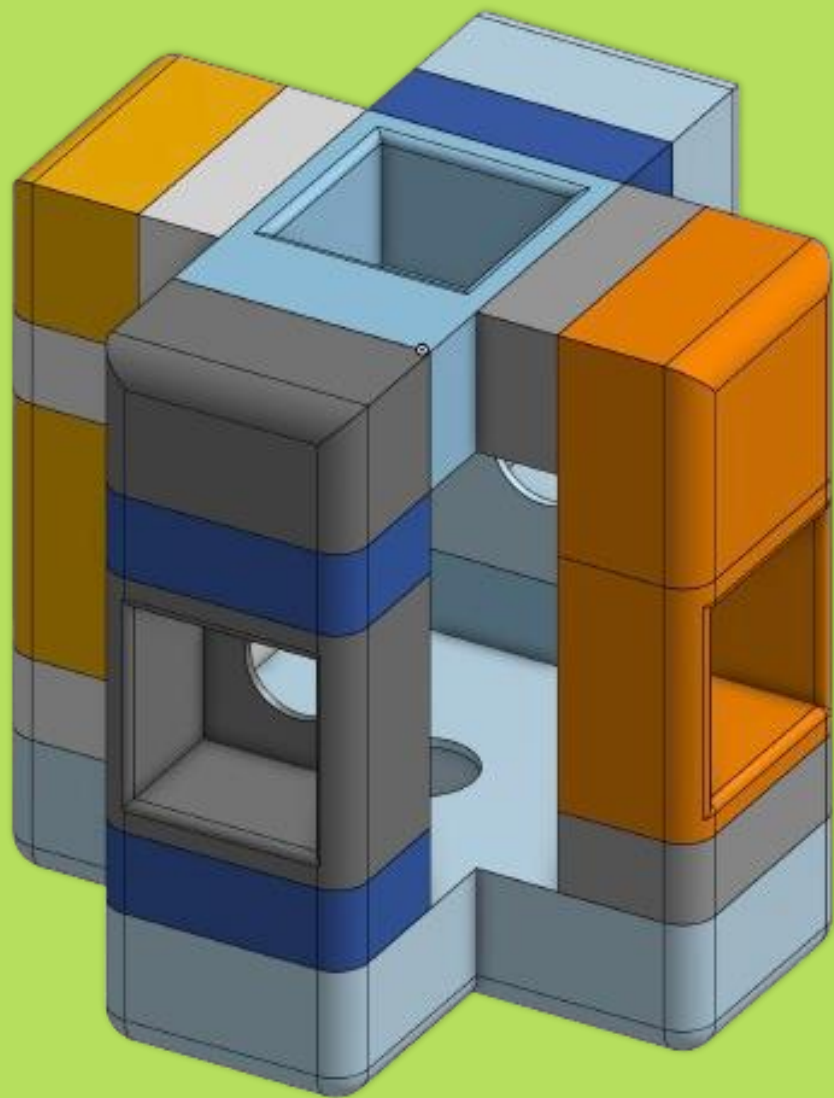


## 5) Hardware experiment: Robot



*The Cube is an imitation of life itself - or even an improvement on life.*  
– Erno Rubik





**Skeleton: Main  
frame of the  
robot was 3D  
modelled and  
printed**



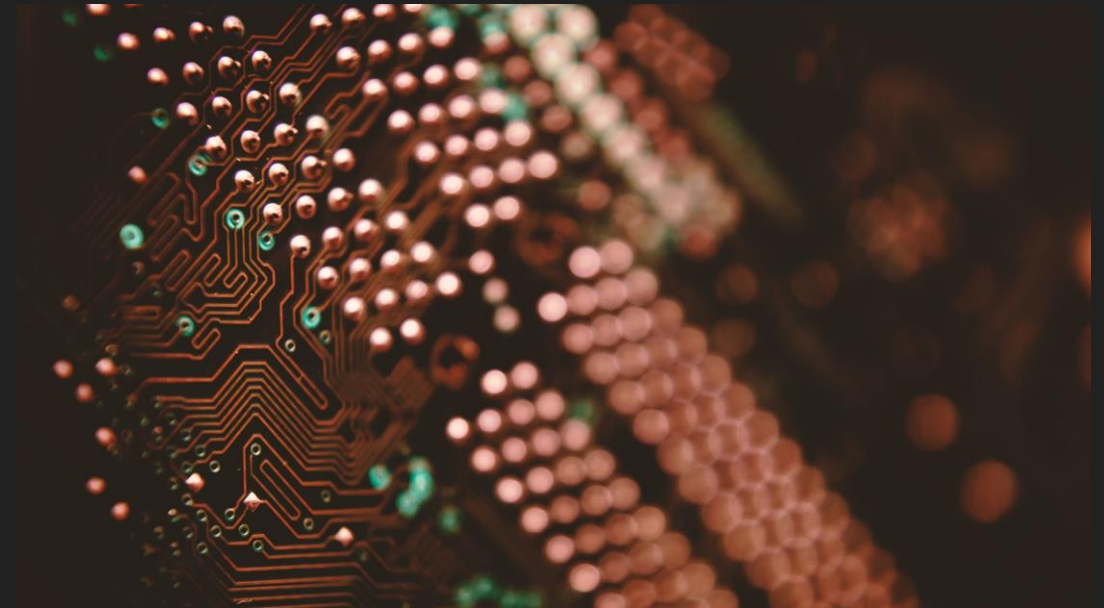
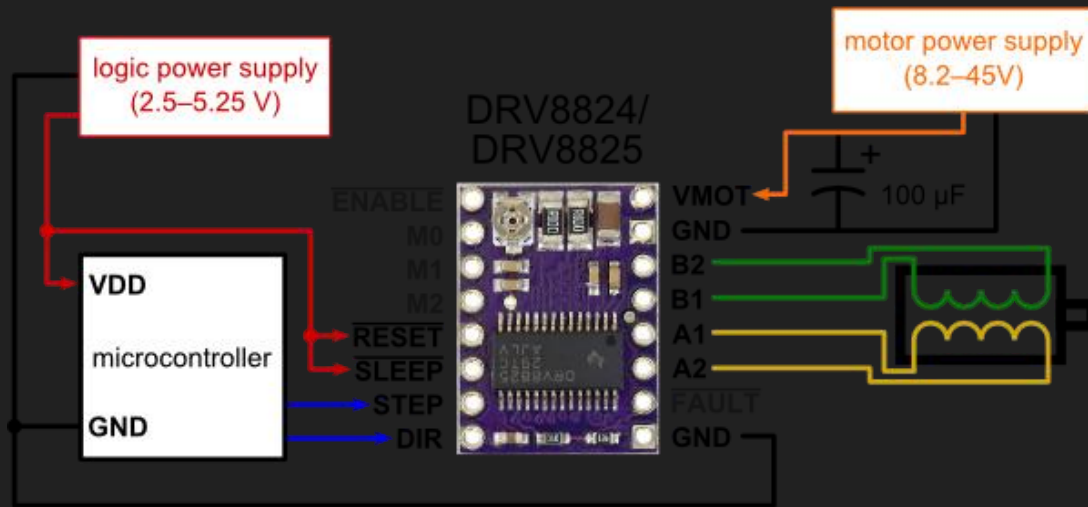
## Arms and legs: Movement is controlled by stepper motors

- Stepper motors are known for their accurate movements
- The robot has 6 motors, each one of them spinning a face
- On top of the motors, there are custom made adaptors to properly fit the center of the face

# Brain of the operation: Raspberry PI

- The main component for software manipulation is a **Raspberry PI**
- The board has specific **ports** that allow connecting **wires** for manipulating hardware components
- On it, a **Flask** server runs, that has an endpoint for getting request with algorithms to perform
- The server is exposed using **Ngrok**, allowing **external** users to connect to the robot





**But the neurons are  
still the conductors:  
DRV8825**

- In order to communicate between the motors and the Raspberry Pi, a special part named **driver** is required for properly handling the signals
- Besides this, the driver has an important part of not overwhelming the motors with more intensity than they support, preventing burning the motors

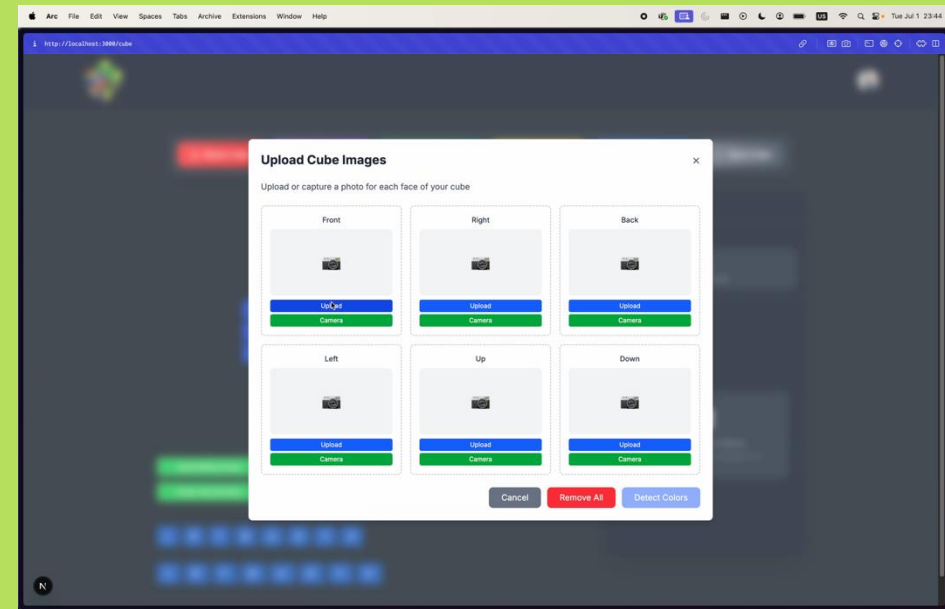
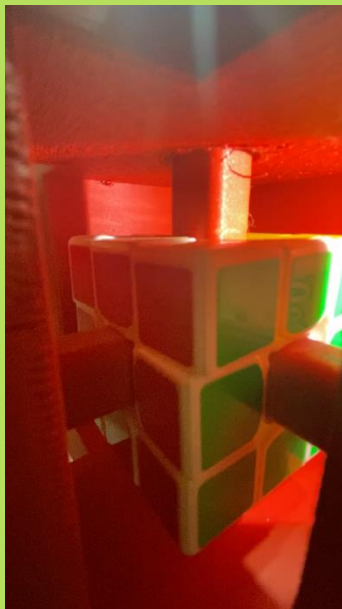


## 6) Demo

*The Cube gives me hope people can solve their problems  
and survive – Erno Rubik*



# Robot Demo – solved a cube in 20 seconds



## 7) Conclusions and future improvements

- All the initially set objectives were **accomplished**
- App **successfully** a solution for any valid scramble of a cube
- The Kociemba algorithm is a **reliable** mechanism for fast solutions
- Color detection software can be extended for adapting to colors specific to cubes (eg. **Pink**)
- **Interdisciplinary** projects and **STEM** education provides better understanding of underlying mechanisms
- Robot should be more portable

*We turn the Cube and it twists us – Erno Rubik*

**Thank you**



# References

1. <https://github.com/hkociemba/RubiksCube-TwophaseSolver>
2. <https://youtube.com/shorts/le6lxzmqUB8?feature=shared>
3. <https://youtube.com/shorts/ue2gZ2vxs48?feature=shared>
4. <https://www.ti.com/lit/ds/symlink/drv8825.pdf>
5. <https://kociemba.org/math/twophase.htm>
6. <https://www.brainyquote.com/authors/erno-rubik-quotes>