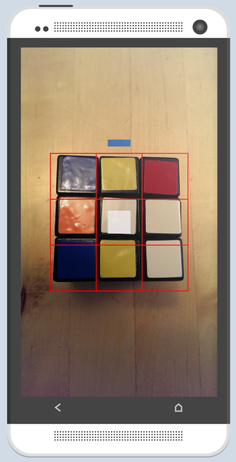
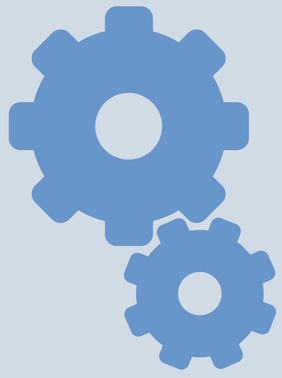
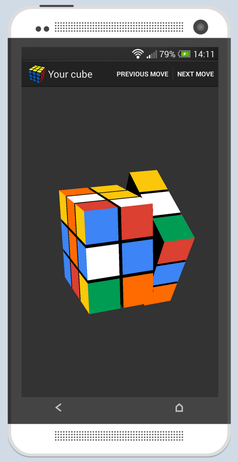
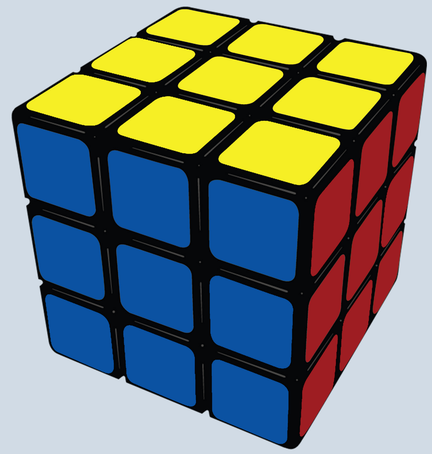
RUBIKS CUBE SOLVER

Lucas Slot Maico Timmerman Govert Verkes Tycho van der Ouderaa Robin Klusman Boudewijn Braams









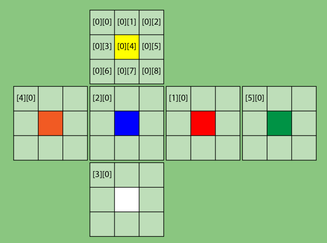
ANALYZE 1

The camera is used to capture every side of the cube, ***OpenCV*** converts these frames to matrices. Using these matrices (which represent the frame in RGB) the mean values for red, green and blue are calculated for every facelet on the cube. The matrices are then converted to the color representation HSV, the mean value for Hue will be calculated for every facelet on the cube. HSV color representation is used to decrease the influence of light (shadow) on the acquired frames. Using a combination of Hue mean values and the RGB mean values, the best fit for each facelet is calculated. Although using Hue mean values is better for cancelling out light effects, the RGB mean values make it easier to distinguish facelets with colors that have very similar Hue values (e.g. red and orange). When a best match is found for every facelet, each corner is checked to see if its a viable corner. If not, it is replaced by a corner that is viable but still holds most found colors.

SOLVE 2

To solve the cube the two phase ***Kociemba*** algorithm is implemented. Two phase meaning that in order to solve the cube it first needs to bring the cube to a certain configuration, for which the set of subqequent solving moves is known. Using this method, it is guaranteed that no more than 25 moves will be required to solve the cube.

Internal cube representation (2D array):



DISPLAY 3

In order to visualize the required moves to solve a cube, a 3D simulation is provided. ***OpenGL*** is used to render a Rubik's cube model and animate the rotations. The model is constructed by translating 26 individual cubes (the middle one being left out) to their appropriate locations (providing some spacing in between the cubes). In solving a Rubik's cube one deals with faces, not individual cubes. In order to facilitate face rotation, each face is tied to certain set of cubes (the edge cubes even belonging to multiple faces). Relative positions of cubes to their face's middle block are pre-defined in order to calculate the desired transformation matrix for the rotation. After completion of each move, the cube is re-initialized with the resulting color configuration; this in order to preserve the aforementioned relative positions. The overall view and perspective on the cube as a whole is driven by touchscreen input, this allows the user a custom view of the model.

APPLY 4

The user can step through all rotations individually, and thus apply them to the real life cube object to solve it.