./

Report – Project Name

Course Code: <CODE>



Version Number:

Team Members :

Team No:

Module: Model Based System Engineering

**Table of Contents**

1. Introduction
2. Block diagram with working explanation
3. Psuedo code
4. Future enhancement
5. References

# Abstract

In this project we are trying to generate all the steps required to solve a Rubix Cube. Here we are using beginner’s approach for solving the cube.

**INTRODUCTION**

Rubix cube is a 3-D combination puzzle invented in 1974 by Hungarian sculptor and professor of architecture Erno Rubik. So here we are trying to solve this rubix cube using c++. The main objective of this project is to display series of steps needed to solve the rubix cube and displaying state of the rubix cube in each step. And also to provide a good user experience.

There are many algorithms used to solve the cube. We are using beginner’s algorithm to solve the rubix cube. This might take extra steps it getting the solution. But our intention is to make it user friendly for the beginner’s who want to learn rubix cube.

# BLOCK DIAGRAM & EXPLANATION

1. Representing color
2. Corner piece

|  |  |  |
| --- | --- | --- |
| Direction 1 | Direction 2 | Direction 3 |
| Color 1 | Color 2 | Color 3 |

1. Middle piece

|  |  |
| --- | --- |
| Direction 1 | Direction 2 |
| Color 1 | Color 2 |

1. Center piece

|  |
| --- |
| Direction 1 |
| Color 1 |

1. Representing the entire 3-D cube
2. Front layer

|  |  |  |
| --- | --- | --- |
| Corner piece | Middle piece | Corner piece |
| Middle piece | Center piece | Middle piece |
| Corner piece | Middle piece | Corner piece |

1. Middle layer

|  |  |  |
| --- | --- | --- |
| Middle piece | Center piece | Middle piece |
| Center piece | None | Center piece |
| Middle piece | Center piece | Middle piece |

## 

1. Back layer

|  |  |  |
| --- | --- | --- |
| Corner piece | Middle piece | Corner piece |
| Middle piece | Center piece | Middle piece |
| Corner piece | Middle piece | Corner piece |

## Explanation :

## In the rubix cube there are 26 pieces which can be divided as corner pieces, center pieces &

## middle pieces. After shuffling the rubix cube, these pieces get scattered all over the place, so it is

## necessary to store both the direction as well as the color associated with each piece. Above tabular

## columns are used to visualize the entire rubix cube in the memory.

## Three dimensional matrix is used to represent each node.

## Essential functions present in class SolveCube are as follows :

# void PlusBottom();//solves the plus pattern in the top layer

# void LayerBase();//solves the first layer

# void LayerMiddle();//to solve bottom 2 layers

# void PlusTop();//to solve the top plus

# void AlignCenter();//to allign all the top center pieces

# void TopCorner();//to solve the top corners

# void FinalStep();//to get the final output

## When Object of this class is created constructor will be executed. This sets up the Rubix cube by

## asking the user to enter all the colors. So constructor calls Algorithm class constructor which calls

## Rubix constructor which calls setRubix function to insert colors into each nodes.

## 

# PSUEDO CODE

# 

# Plus Bottom

# Loop until bottom plus is formed

# If ( middle piece of top color is found)

# Use functions of class Algorithm or Rubix to put the middle piece into position

# Else

# Continue

# End loop

# Layer Base

# Loop until base layer is solved

# If( corner piece of top color is found)

# Use functions of class Algorithm or Rubix to put the corner pieces into position

# Else

# Continue

# End loop

# Layer Middle

# Loop until middle layer is solved

# If( middle piece is found at the top middle portion)

# Use functions of class Algorithm or Rubix to put the middle pieces into position

# Else

# Continue

# End loop

# Plus Top

# Loop until top layer plus is solved

# Use functions of class Algorithm or Rubix to put the corner pieces into position

# End loop

# Align Center

# Loop until the center is aligned

# If(center is not aligned properly)

# Use functions of Rubix or Algorithm class

# Else

# Continue

# End loop

# Top Corner

# Loop until the top corner is placed the right position

# If(corner is not aligned properly)

# Use Functions of Rubix or Algorithm class to align the corner

# Else

# Continue

# End loop

# Final Step

# Loop for 4 iterations

# If(corner is not solved)

# Use functions of Rubix or Algorithm class to solve the corner

# Else

# Perform upHorizontal

# End loop

# FUTURE ENHANCEMENTS

* Using graphics designing to help beginners learn rubix cube.
* Optimizing the numbers steps in solving rubix cube using latest algorithms.

# REFERENCES

# <https://www.stackoverflow.com>

# https://www.geeksforgeeks.org

# 