Presentation on-“Rubik’s Cube Solver Machine with a new technique”

Presented by-

1. Shakur Mahmood

ID: 191008042

2. Md Nasir Uddin

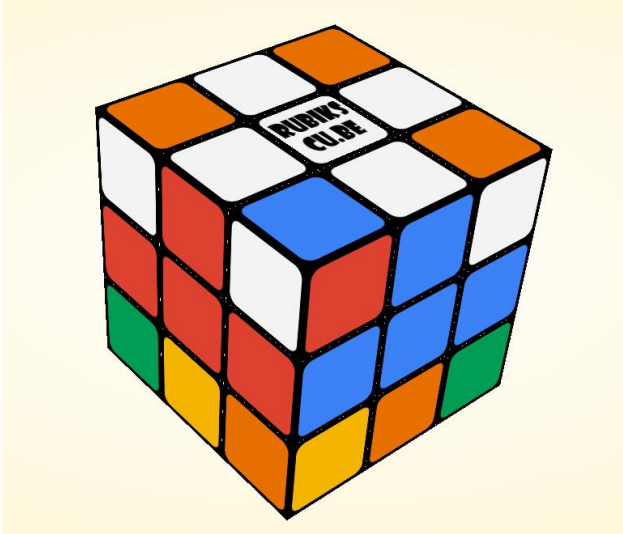
ID: 171005042

3. Shahadat Khan Sakil

ID: 171025042

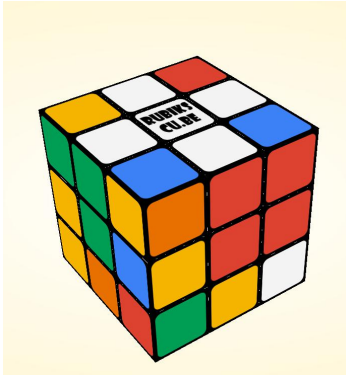
New technique

- If we solve the second layer's 4 edges after solving the white cross, it becomes easier to solve the whole cube.

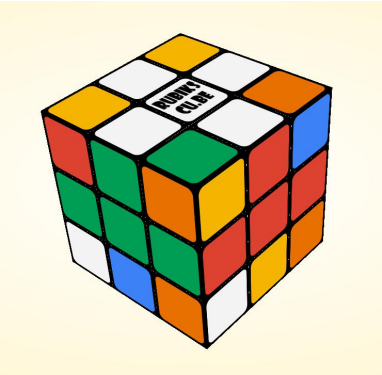


- Infact using only one algorithm we can solve the first two layers.

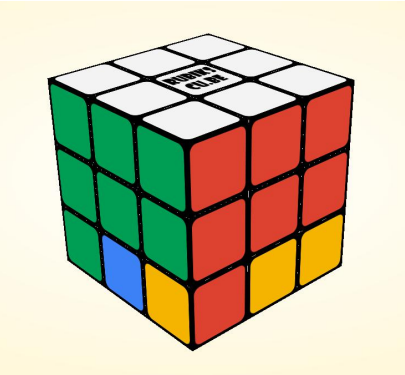
Steps to solve the total Rubik's Cube in this way



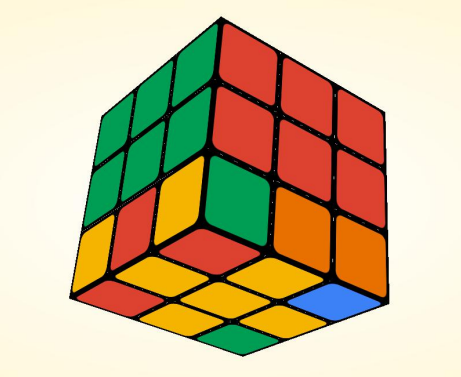
1. Solving White Cross



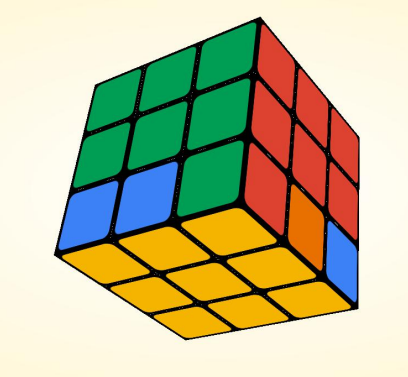
2. Solving 2nd layer's edge pieces



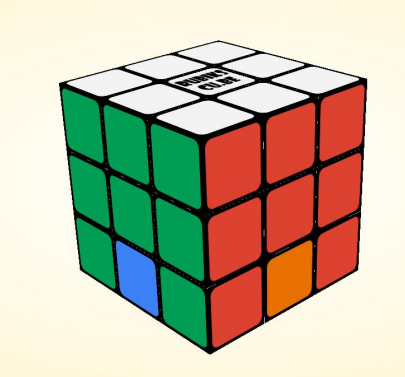
3. Solving the top corners



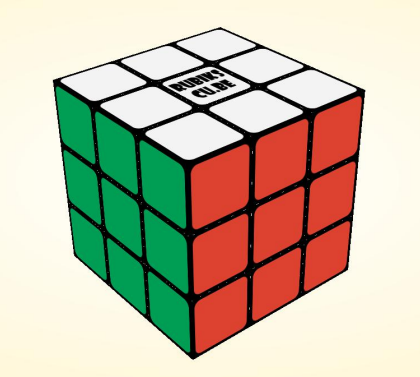
4. Making Bottom Cross



5. Solving the bottom side



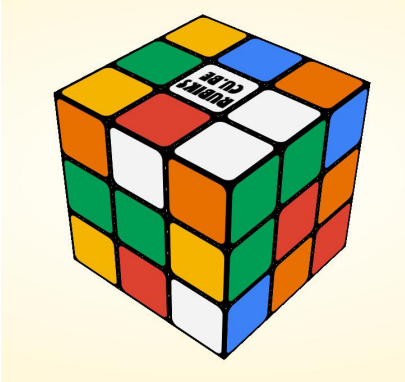
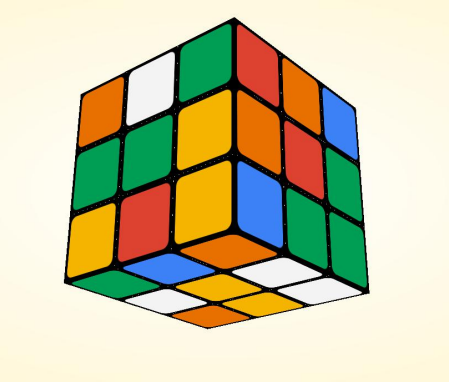
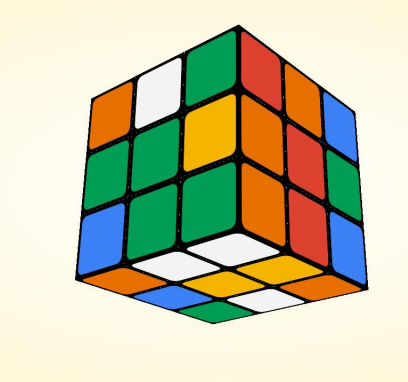
6. Making headlights in all the sides



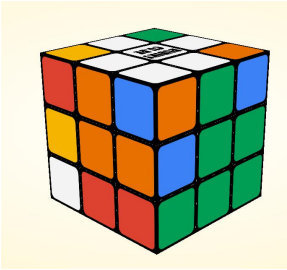
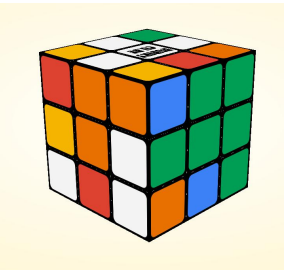
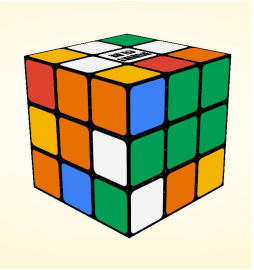
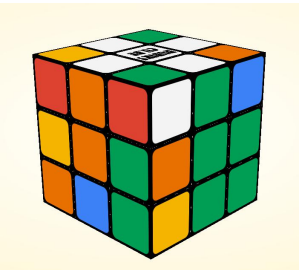
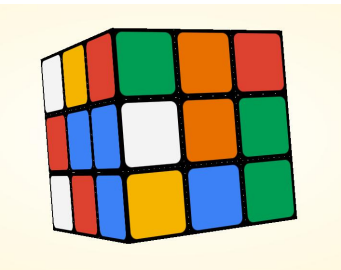
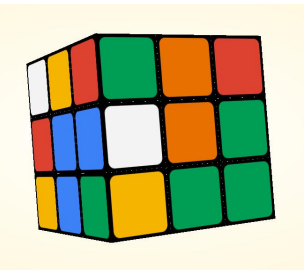
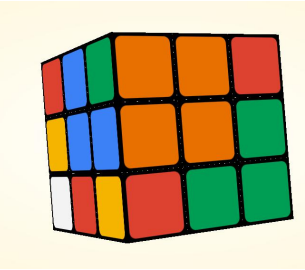
7. Solving the headlights

1. One of the significance of this technique is white color will face always on top and
2. We need not to rotate the top layer at all. So, we can make our solver machine using 5 stepper motors instead of 6.

- To solve the first two layers edge pieces, if we take the correct piece to the bottom layer it would be much easier to solve that edge position like this-


- For solving middle layers' all edges we can also take the correct piece to the bottom layer. Then we need to match the opposite color with the middle layer's center piece. Atlast we need to do "r D R" or, "L d l" depending on the situation. Here, capital latter represents clockwise rotation and small letter represents anticlockwise rotation.

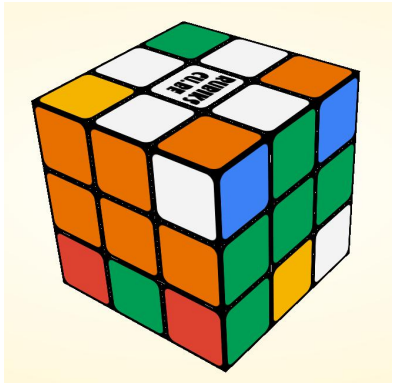








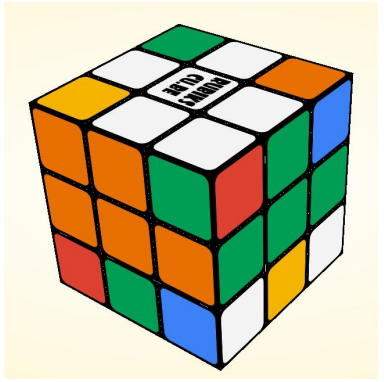
First Algorithm

Now we need to solve our top 4 corners. To do this we need to use our first algorithm. it is-

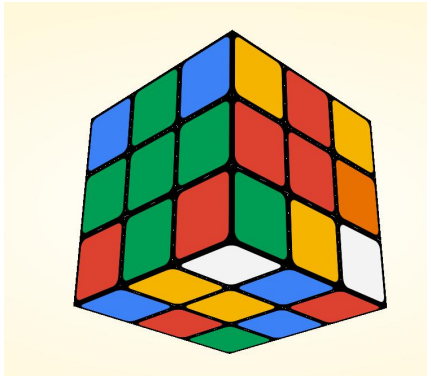
r d R D (right hand algorithm)
or,
L D l d (left hand algorithm)

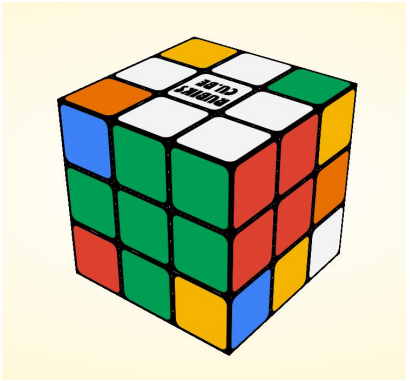
If we find our correct corner piece in the top layer we need to take that piece to the bottom layer by applying this algorithm thrice. Which will not ruin the solved second layer.





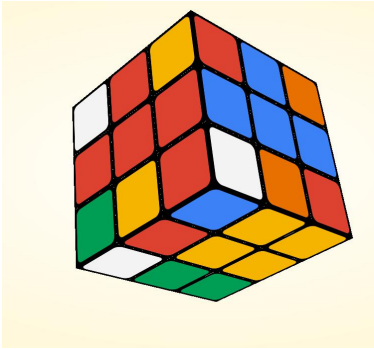
- After getting the correct corner piece in the last layer, we need to put that alligning with it's correect position.
- If white color is facing downward then we need to apply the algorithm thrice just like we have used it to take down a corner piece from top layer.

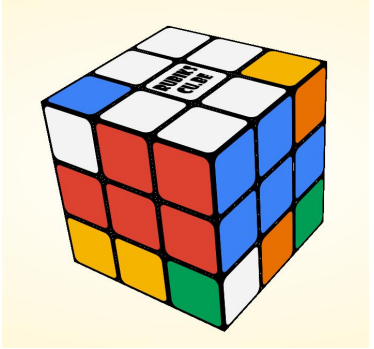




- If white color is facing the right side, we need to use the following algorithm-

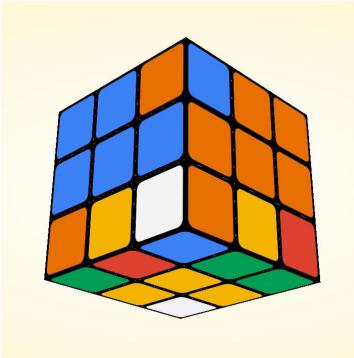
d (r d R D) D (r d R D)

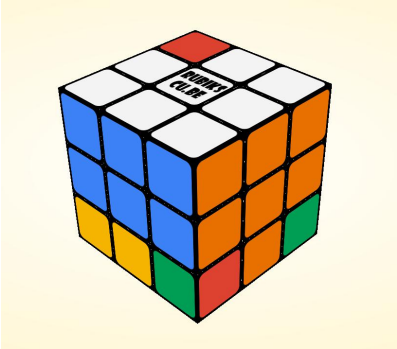




- If white color is facing the right side, we need to use the following algorithm-

D (L D l d) d (L D l d)





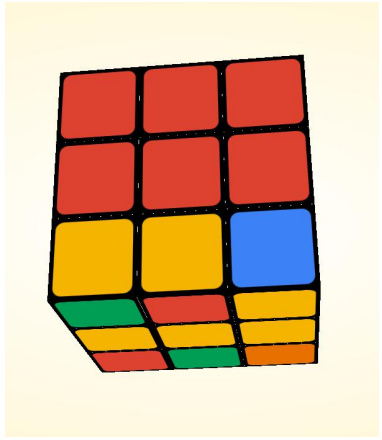
- So, actually using only one algorithm we can solve the first 2 layers.

Second algorithm (making bottom cross)

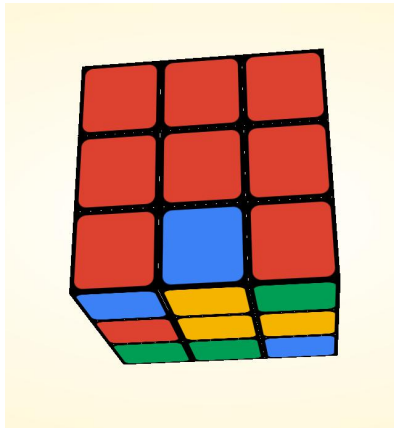
This algorithm is-

f (r d R D) F (right hand algorithm)
or,
F (L D l d) f (left hand algorithm)

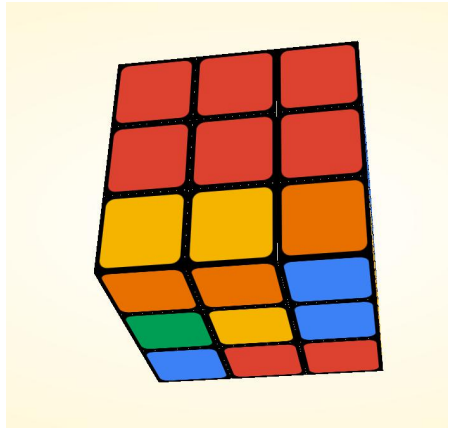
- If bottom cross is not completed automatically, 3 situations can be possible with the center yellow piece-



1. ‘The line’ situation



2. Capital ‘L’ situation



3. ‘Dot’ situation

- When we will get ‘The line’ situation, we need to take the rubik’s cube in such a way so that ‘The line’ remains horizontally. Then we need to apply this algorithm once which will create our ‘bottom cross’.

- In the case of capital ‘L’ situation, we need to take the rubik’s cube in such a way so that ‘L’ remains infront of us. then applying this algorithm will create ‘The line’ situation and we know what to do in ‘The line’ situation.

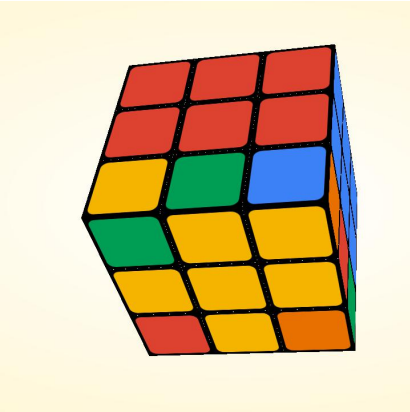
- In the case of ‘Dot’ situation there will be only one yellow dot in the center or, at most there will be another yellow colored piece on the bottom cross area. Then if we apply this algorithm that will create capital ‘L’ situation. And we know what to do in the capital ‘L’ situation.

Third Algorithm

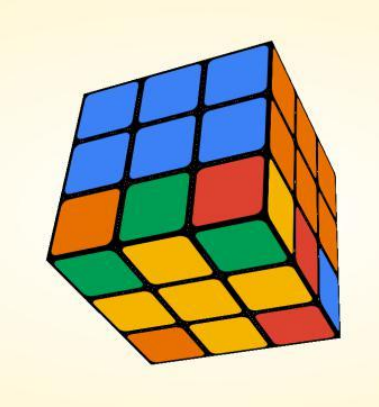
- Now we need to complete our full bottom side. Here we need to use our third algorithm.
- Our second algorithm and third algorithm are part of OLL algorithms.
- Our third algorithm is-

r d R d r d d R (right hand algorithm)
or,
L D l D L D D l (left hand algorithm)

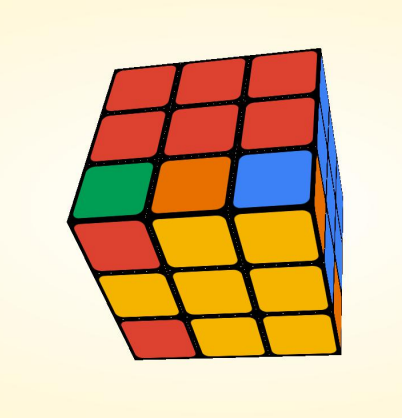
Here also 3 situation can be possible-



1. One solved corner



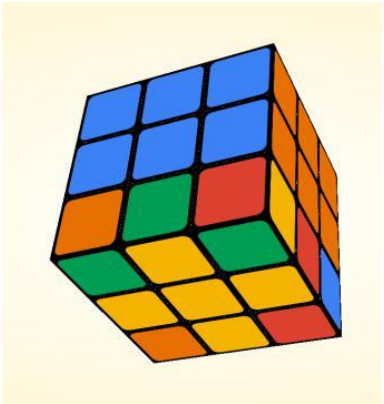
2. No solved corner



3. Two solved corner

- In the case of 1 solved corner we need to put the solved part in the left side in front of us and apply the right hand algorithm or, putting the solved part in the right side in front of us we need to apply the left hand algorithm. Both the cases will take us to either totally solved side or, one solved corner situated in another position. If we apply the 3rd algorithm in same way on that one solved corner will take us to totally solved side.

- In the case where there is no solved corner there must be a side where the is no yellow pieces near to second layer like the picture below-



If we apply 3rd algorithm there that will bring us a one solved corner and for a one solved corner what we are needed to do we have just discussed.

- In the case where there are 2 solved corners if solved parts are situated next to each other we need to put those solved corners in our rignt side and apply the right hand algorithm or, we need to put those solved part in the left side and apply the left hand algorithm.

- For diagonally positioned solved parts we can put one solved corner to our right side and can apply the right hand algorithm or, putting 1 solved corner to the left side we can use left hand algorithm.

- Both the works will take us to either 1 solved corner or, no solved corner. And now, we know what to do in those situations.

Fourth Algorithm

- After solving bottom side's all the yellow colors we have only remaining the bottom sides near to second layer. Now we are needed to use our 4th algorithm.

- 4th and 5th algorithms are part of the 'PLL algorithm'. This algorithm is-

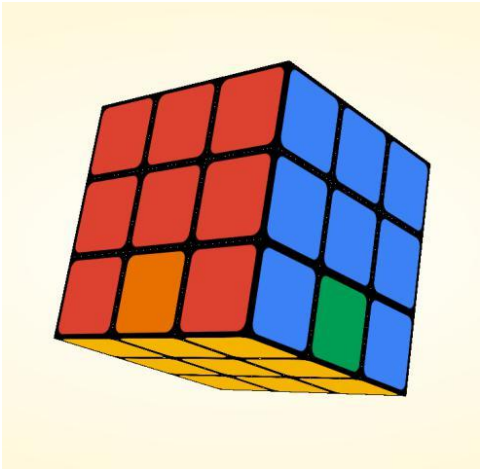
r B r F F R b r F F R R (right hand algorithm)
or,
L b L F F l B L F F L L (left hand algorithm)

- We will use this algorithm to create headlights in all the sides. So, if there is at least 1 side that does not contain headlight we will match it with the second layer's center piece and will use this algorithm.

- If we have not found any headlight containing side we can apply the algorithm to any side which will create at least 1 side containing headlight.

- But if we have found that there is headlights and not all the sides contain headlights we are needed to put one headlight containing side in front of us and needed to apply either of the 2 algorithms above which will create headlights in all the sides.

- After getting that we are needed to match the headlights with the second layer's center pieces. Picture of a headlight is like this-



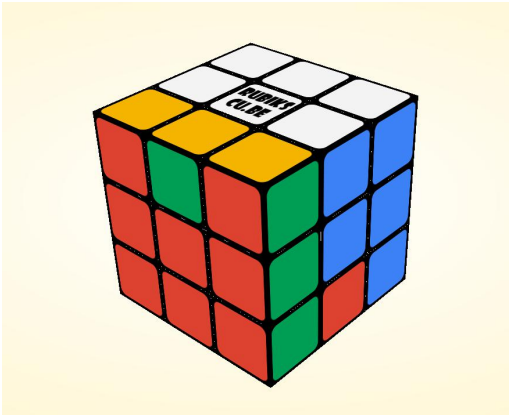
- If we get headlights in all the sides automatically, we need not to use this algorithm.

Fifth Algorithm

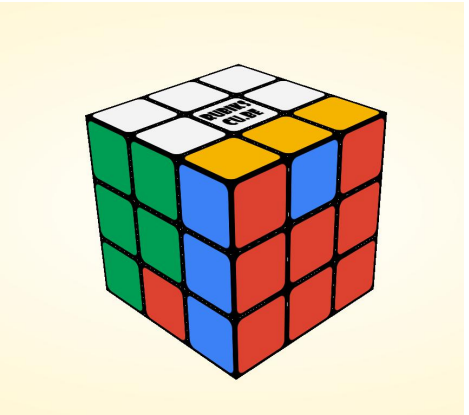
- Now we are needed to use our 5th algorithm. This algorithm is-

F F (d/D) R L F F r l (d/D) F F

- Here, **d/D** means depending on the center pieces it can be either **d** or, **D**. After doing **d** if the edge piece color matches with the center piece color then we need to use **d**. If **D** can do this then we are needed to use **D**. If neither **d** nor **D** can do this then we can chose anyone of the two. But in all the cases we need to remember the term and need to use it in the second **d/D** term.



Here, we need to use **'d'**.

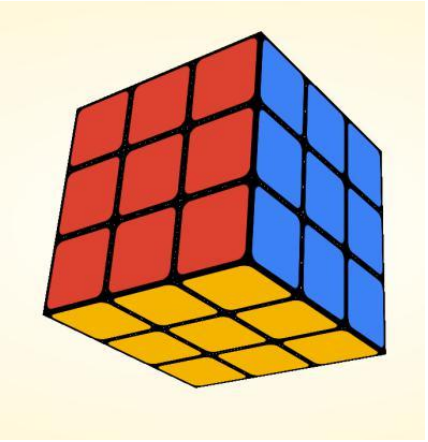


Here, we need to use **'D'**.

- To use this algorithm we are needed to see 3 situations. First of all, there can be a solved side which is called ‘a bar’ containing side or, the headlights are created in such a way where edge colors are matched with the opposite sides center pieces or, edge colors are matched with the adjacent sides center pieces like a ‘lightening bolt’ sign.

- If we find a bar we need to hold the rubik’s cube in such a way so that the bar containing side faces backward. In the case where 3rd layer’s edge pieces are matched with the adjacent sides center pieces, we need to hold the rubik’s cube in such a way so that one solved part faces backward and another solved part faces our left side.

- If we apply this algorithm now it will either solve the rubik’s cube or, will create these types of situation again. If we use this algorithm again for that situation will solve our total cube.



The code

- We have developed our code using python. Python is a well structured language so, though our code is long (near 7500 lines) but we did not get stretch.

- Our main technique of running the stepper motor is we have saved our desired outputs in an array and then creating a for loop we have run our stepper motor running part.

- We have tested our code again and again for a lot of time and the code shows 100% correct output.

- To write the code we have followed our newly invented technique which is much easier to understand and implementing in the code. Here, we were needed to consider all the possible cases. But in worst case, out of 64 possible cases only 4 cases will run.

- Arm based computer like raspberry pi contains input output pins through which our stepper motor drivers(drv8825) are connected. With the drivers our stepper motors are connected. To use those pins the operating system has preinstalled libraries. Only we need to call the necessary libraries from our code like-

```
import RPi.GPIO as GPIO
from time import sleep
```

Sample screenshot of our code for running raspberry pi is as below:

```
# for Raspberry pi

CW = 0     # Clockwise Rotation
CCW = 1    # Counterclockwise Rotation
SPR = 50   # Steps per Revolution (360 / 1.8)

# Defining pin numbers

STEP_L = 23
DIR_L = 24

STEP_R = 12
DIR_R = 16

STEP_B = 20
DIR_B = 21

STEP_F = 10
DIR_F = 9

STEP_D = 19
DIR_D = 26

# Defining MODE (Microstep Resolution GPIO pins)

MODE_L_MOTOR = (14, 15, 18)
MODE_R_MOTOR = (25, 8, 7)
MODE_B_MOTOR = (2, 3, 4)
MODE_F_MOTOR = (17, 27, 22)
MODE_D_MOTOR = (5, 6, 13)

sleep(5)

i=0
for i in range (len(result)) :
    if result[i]=='L' :

        GPIO.setmode(GPIO.BCM)
        GPIO.setup(DIR_L, GPIO.OUT)
        GPIO.setup(STEP_L, GPIO.OUT)
        GPIO.output(DIR_L, CW)
        GPIO.setup(MODE_L_MOTOR, GPIO.OUT)
        GPIO.output(MODE_L_MOTOR, (0, 0, 0))

        step_count = SPR
        delay = 0.001

        x=0
        for x in range (step_count) :
            GPIO.output(STEP_L, GPIO.HIGH)
            sleep(delay)
            GPIO.output(STEP_L, GPIO.LOW)
            sleep(delay)

        sleep(5)

    elif result[i]=='l' :
```

Machine part

Our project's design is simple like the diagram below:

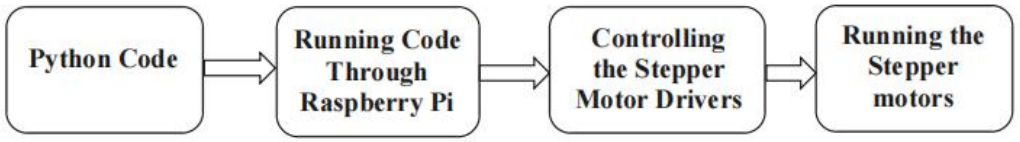


Diagram : Project design

One of the main ingredients of our project is our stepper motor driver DRV8825. Which is designed and developed by 'Pololu Robotics'. It has a small IC named DRV8825.

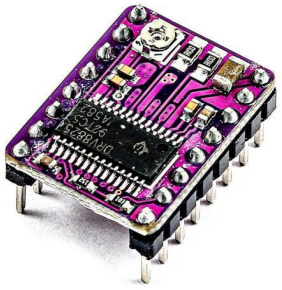


Photo: DRV8825 Driver

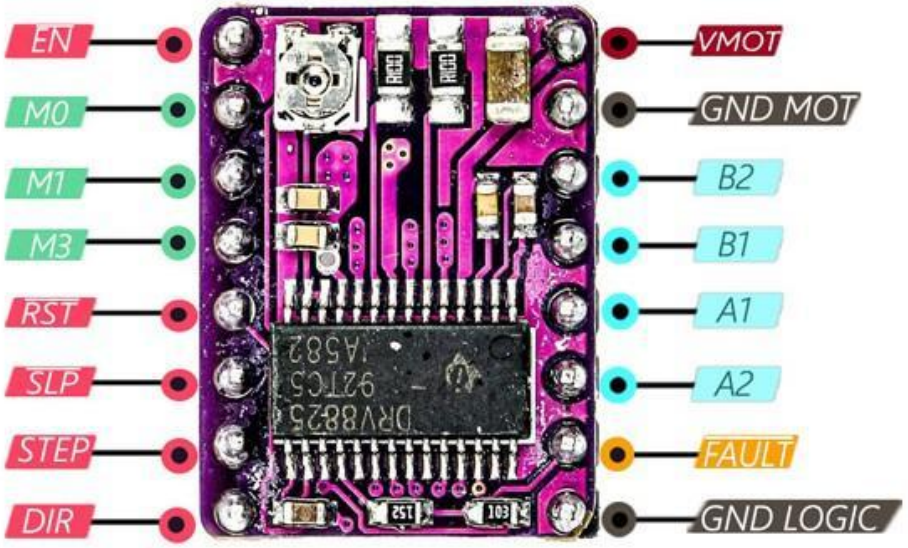


Diagram : Labelling all the pins of a DRV8825 driver.

- Here, EN (enable pin) is by default set to on. If want to stop working of that pin we need to connect that pin to ground which will stop the whole working process of the driver.
- M0, M1, M2 are the pins for the micro stepping mode.
- Enabling RST (reset pin) will reset all the previous setup. We should connect the pin with raspberry pi's 3.3 volt pin.
- SLP (sleep pin) pauses all the works on the driver for the time we will set in our program. To use this pin we need to connect this pin with the raspberry pi's 3.3 volt pin.
- STEP pin is used to run the stepper motor's single step (in our stepper motor this step is 1.8 degree).
- DIR (direction pin) give direction that the motor will rotate clockwise or, anticlockwise.

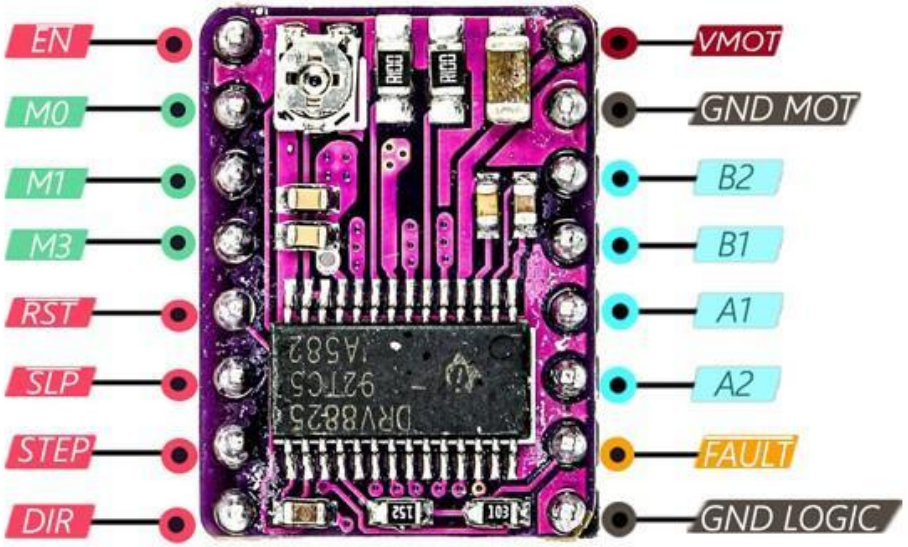
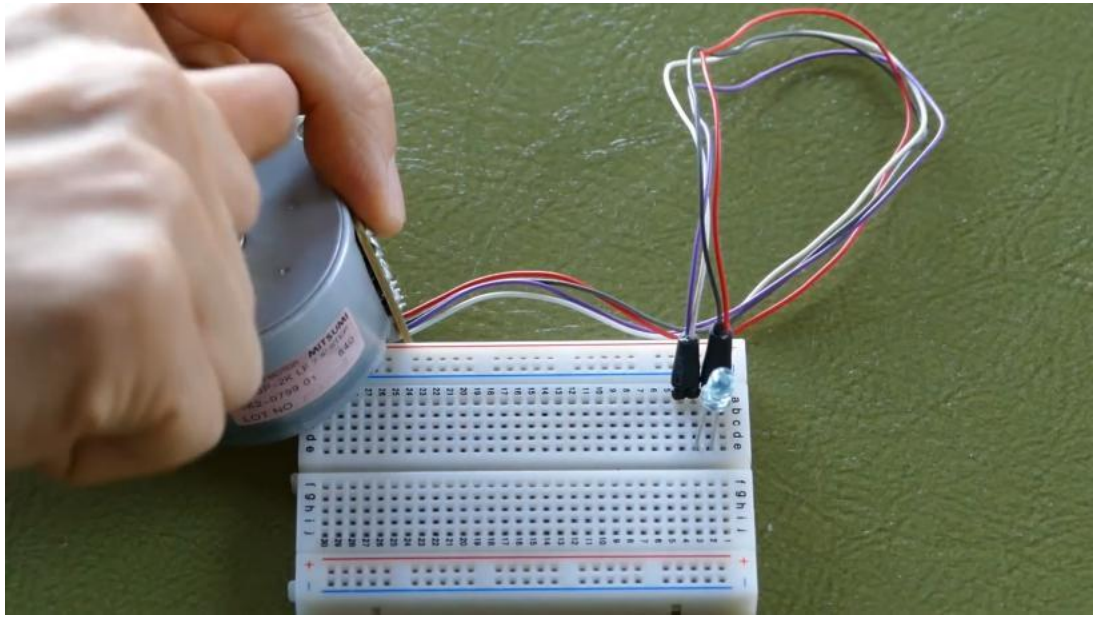


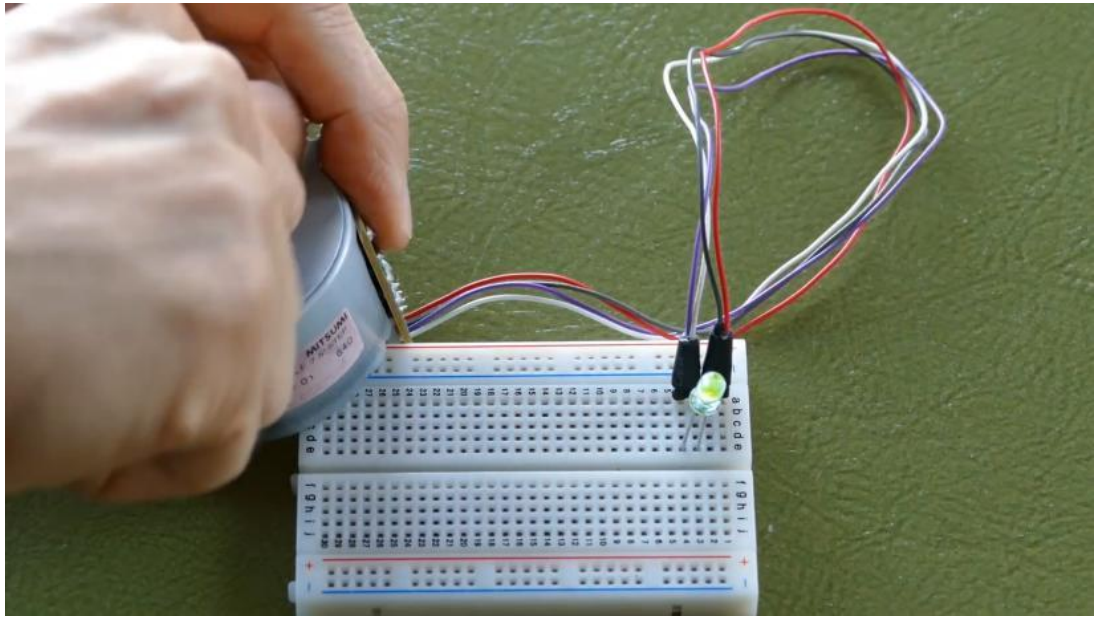
Diagram : Labelling all the pins of a DRV8825 driver.

- VMOT is the positive connection part of our external voltage source.
- GND MOT is the negative connection part of our external voltage source. We should make a rail with the raspberry pi's ground.
- A1, A2 and B1, B2 are the pins for connecting a stepper motor's 2 coils 4 wires.
- FAULT pin is not used in general cases.
- GND LOGIC pin should be connected with the raspberry pi's ground pin.
- We can also notice that there is a screw in the top left side of the driver. By rotating clockwise it lowers down the current flow and rotating anticlockwise increases the current flow.
- To continuously supply current drv8825 driver can provide maximum 1.5 ampere current and over 2.2 ampere current can destroy the driver. This driver supports at most 45 volt current over which will destroy our driver.

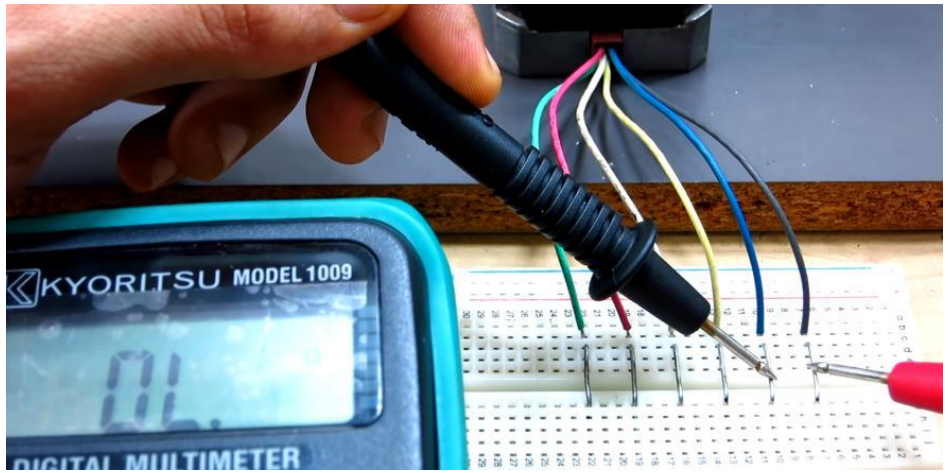
How to identify wires from the same coil of the stepper motor

- We can easily find it with a small led light. After connecting it with the 2 wires when we will rotate the motor with hand if the led turns on it means those coils are from the same coil.





- This process works well with the 4 wire bipolar stepper motor but for the 6 wire bipolar stepper motor we also need to identify the 2 neutral wires. To do this we need to use a multi meter to measure wires resistances.



- Wires from the same coils will show reading on the multi meter. Along the three wires from the same coil we will get 3 values on the multi meter. Middle valued wire is our one neutral wire.

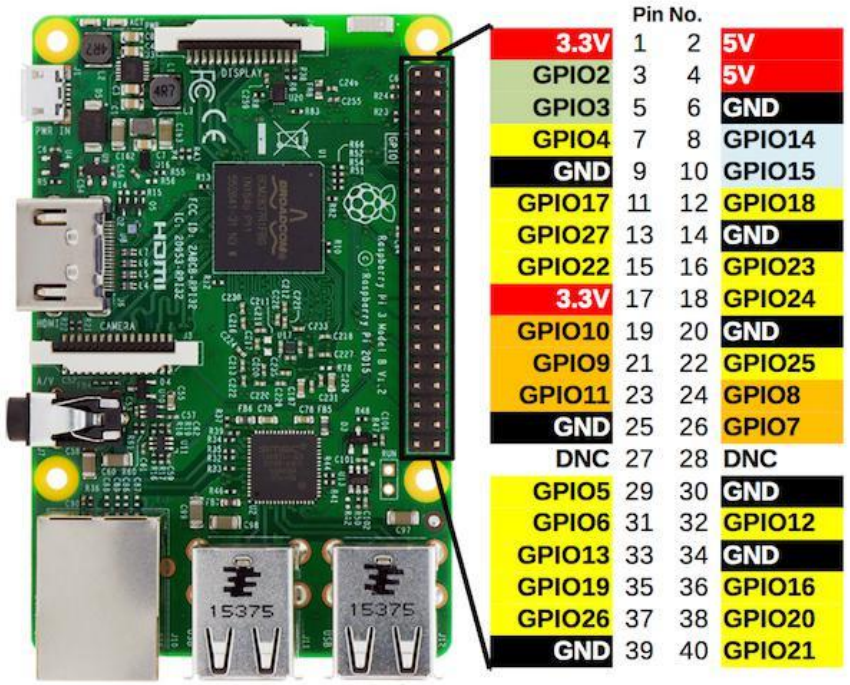
- Among the other 3 wires we will get 3 values on the multi meter. Here, also the middle valued wire is the neutral wire.

- In most of the cases white and yellow wires are the neutral wires.

Raspberry pi setup

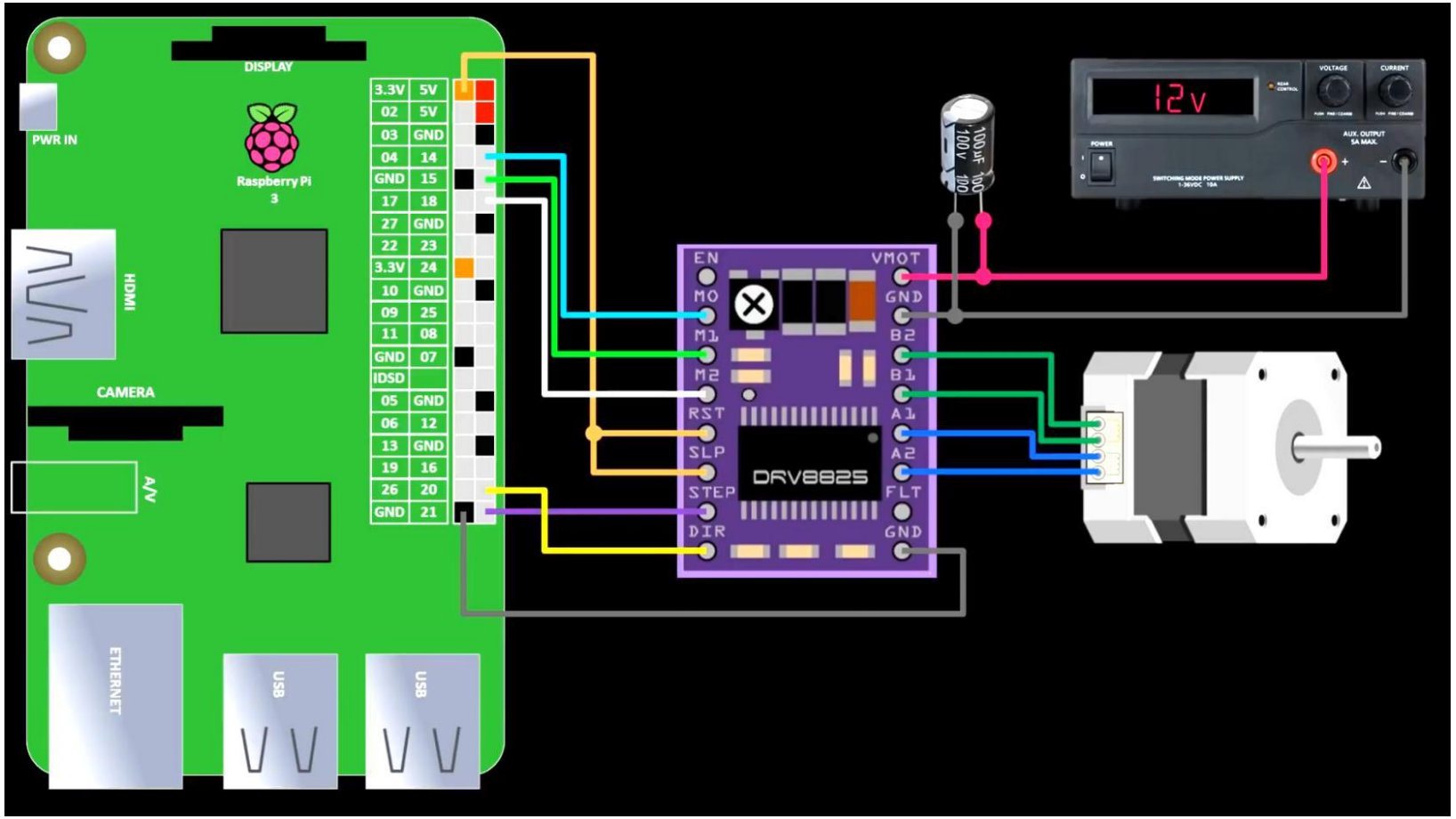
- There are 40 i/o pins in the raspberry pi. But not all the pins can be used as i/o pin because there are some ground pins, two 3.5v pins, two 5v pins and some other pins.

- More importantly we need to know the GPIO pin numbers to write the code but it is not labeled on the raspberry pi.

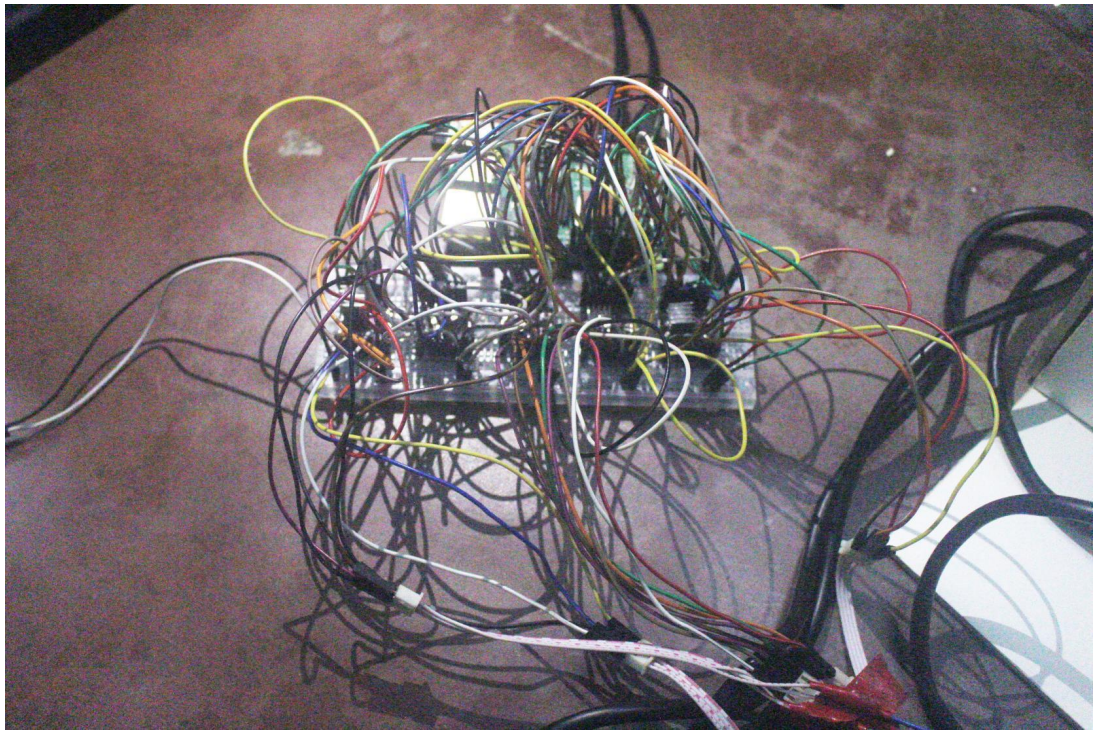


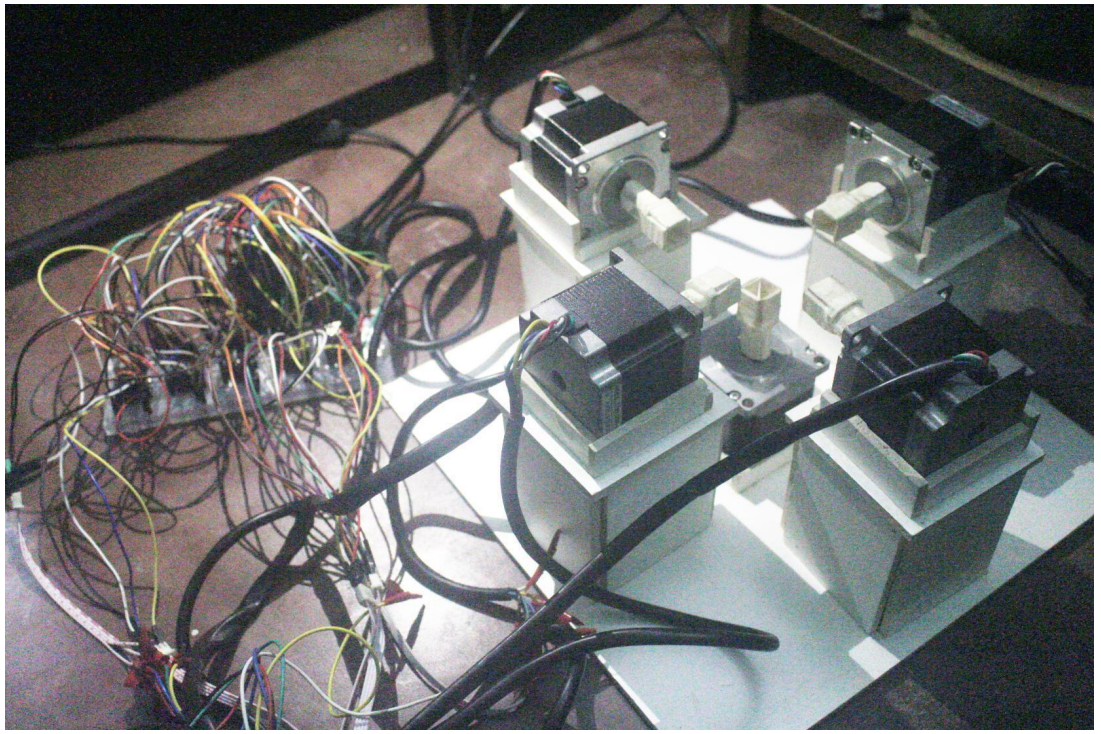
Picture : Labelling of raspberry pi's i/o pins.

Our total setup for running one stepper motor is like the diagram below-

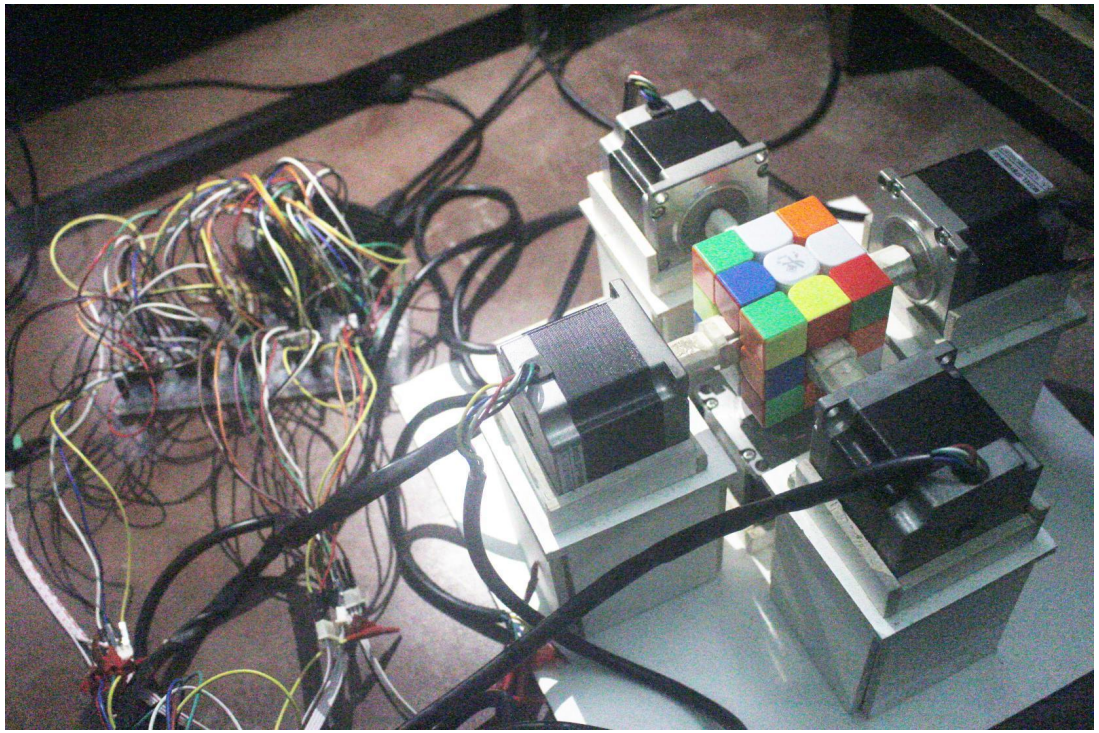


Now, we can setup 5 stepper motors which we are needed to make the rubik’s cube solver machine-

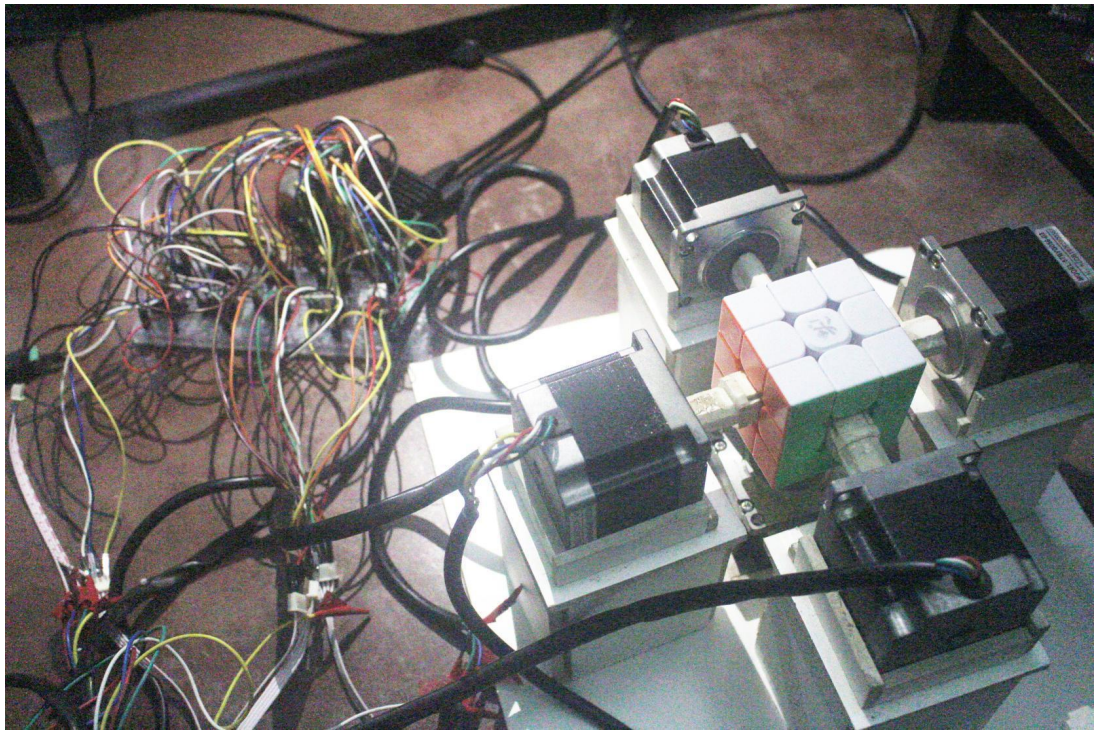




And finally making a frame for setting our stepper motors properly and adding a scrambled rubik’s cube we can run our rubik’s cube solver machine.



Picture : Before solving



Picture : After solving

Thank You