Introduction to Roux

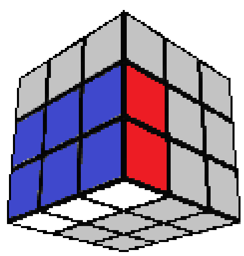
Overview

The Roux method, invented by Gilles Roux, is an alternative to the CFOP method. It begins by building two 1x2x3 blocks on opposite sides of the cube. This leaves the U and M slices free to turn. Next, the top layer corners are solved. The remaining six edges are oriented and positioned using U and M turns.

Roux is a fast, intuitive method. It requires fewer turns than CFOP, uses fewer algorithms than CFOP, and requires no cube rotations after the first step. Beginners may struggle with the block building involved in Roux, and the heavy reliance on M turns may be unappealing to some.

Step 1: Solve the First Block

We can begin making a 1x2x3 block of any color, but this tutorial will start with the blue block shown below.

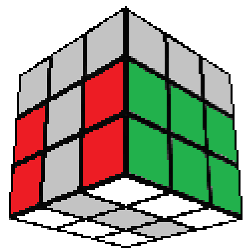


There are no algorithms to make a 1x2x3 block, only ideas. One approach is to insert the white/blue edge followed by each of the two CE pairs around it. Solving the block in 8-10 moves is a good goal.

The first block will remain on the lower left of the cube for the remainder of the solve. Left-handed solvers may prefer to keep the first block on the lower right.

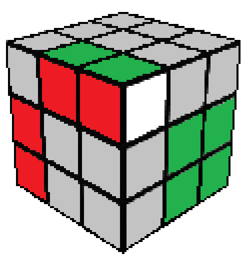
Step 2: Solve the Second Block

Next, we form another block on the opposite side of the cube without disrupting the first block.

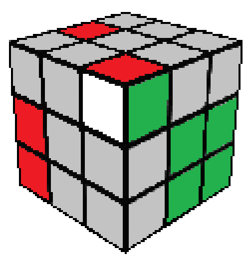


As with the first block, there are no set algorithms. Solving the second block in 10-12 moves is a good goal.

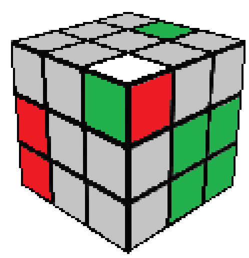
F2L techniques can be helpful in forming the second block while preserving the first one, but there may be more efficient ways to form the block utilizing the M slice. Here are some example ideas.



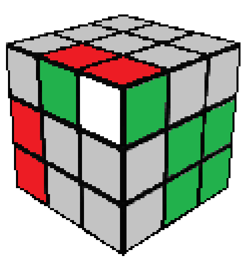
(rUr’)



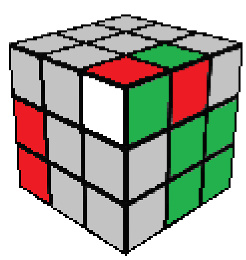
(U’MU2) (rU’r’)



(rU’r’) U2 (rUr’)



(MU) (rU’r’)



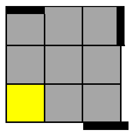
(U’M2U2) (rU’r’)

Step 3: Solve the Top Corners

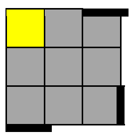
Once the two 1x2x3 blocks are in place, the top corners are solved. CMLL algorithms do this in one step (see next section), but here we will orient and position the corners separately.

3a: Orient corners

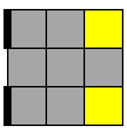
There are seven cases for orienting corners. These corner algorithms are in some cases simpler than those in CFOP because the LL edges and center do not matter.



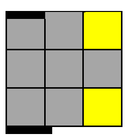
Sune
RUR'URU2R'



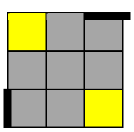
Anti-Sune
R'U'RU'R'U2R
U2 L’U’LU’L’U2L



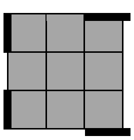
Headlights
F (RUR’U’) F’



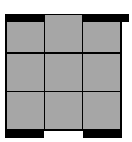
Blinkers
(RUR'U') (R'FRF')



Bowtie
(RU2R'U') (RUR'U')*2 (RU'R')
U2 (FR'F'R) (URU'R')



Pi
F (RUR’U’)*2 F’



Double Headlights
(RU2R'U') (RUR'U') (RU'R')
F (RUR’U’)*3 F’

3b: Position corners

There are two cases to position the top corners. Both can be solved with familiar PLL algorithms.

If there are matching corners (corners with the same color sticker on the same side), then turn U to put the matching corners on the left and use a J Perm:

(RUR'F') (RUR'U') (R'F) (R2U'R') (the final U' is unnecessary)

If there are no matching corners, use a Y Perm with any U position:

(FRU') (R'U'RUR'F') (RUR'U')(R'FRF')

Step 4: Solve the Last Six Edges

Solving the last six edges (LSE) can be done in 3 steps.

Step 4a: Orient the Last Six Edges

In this step, we orient the remaining 6 edges (the 4 on top plus the bottom 2 between the blocks). First, turn the M slice so that the white or yellow center is facing up. Only then can we easily see if an edge is oriented (good) or disoriented (bad). Second, count the bad edges:

- *An edge is oriented correctly if its yellow or white sticker is on the top or bottom of the cube.* Otherwise, it is a bad edge that needs to be flipped. In other words, when all the edges are oriented, the top and the bottom of the cube will have only yellow or white stickers. Edges are always flipped in pairs, so there will be 0, 2, 4, or 6 bad edges.
- *Don't look at the DB edge.* Five of the six edges are immediately visible, so the state of the hidden DB edge can be inferred without rotating the cube to look at it. If there is an odd number of bad edges visible, DB is also bad. If there is an even number of bad edges visible, DB is good.

Third, turn the U layer until your edges match one of the 11 cases below. You can orient the edges in two ways: cycles or direct algorithms. Cycles require less memorization, but algorithms are optimal for speed. Since algorithms are straightforward, I will focus on clarifying cycles.

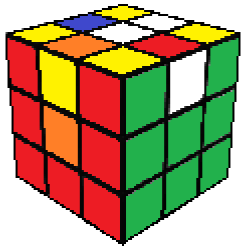
There are two algorithmic tools used in cycles:
- (M'UM') flips the orientation of the UL, UF, UR, and DF edges (the Arrow F case)
- (M'U2M) switches the position of the UF and DF edges

These intuitive moves are used to convert any case to an Arrow case, which is then solved with M'UM' or MU'M'.

I have divided the Arrow case into Arrow F and Arrow B to clarify the location of the bad bottom edge. In each case, you turn the U slice so that the top "arrow" points to the bad bottom edge, bring the bad bottom edge up with M or M’, then orient all edges with any U turn move followed by any M turn.

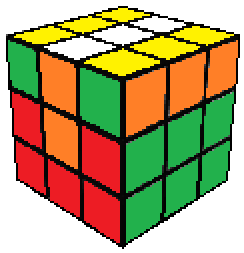
Here is an example.

Before:

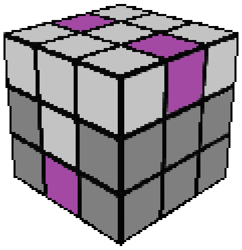
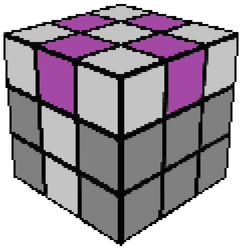
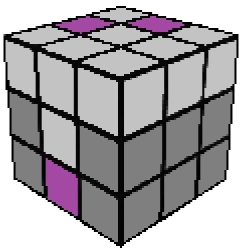
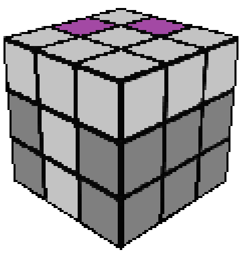
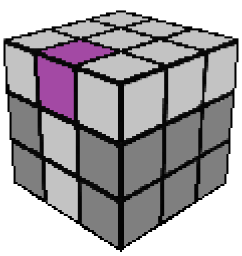


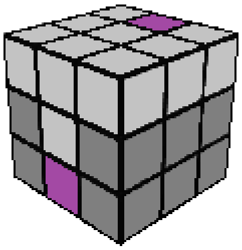
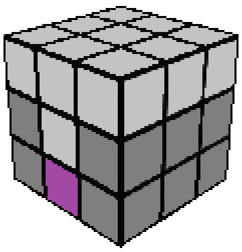
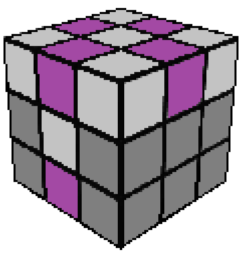
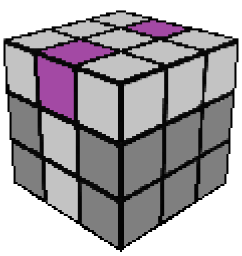
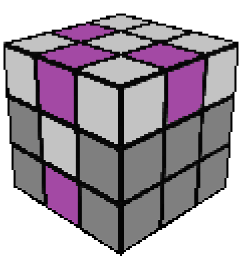
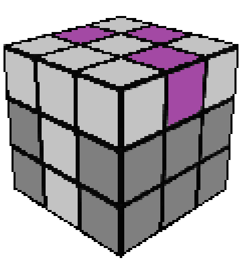
We can see that there are 3 bad edges on the top (UL, UF, UR). We can also see that the UB and DF edges are good. Since the total number of bad edges must be even, then we know that the DB edge is bad even without seeing it. This is an Arrow B case which could be solved (U2 MU’M’). The result is shown below.

After:



Our five visible edges are all good, so we know the DB edge is also good. This completes edge orientation.

Case	**Name (Edges/Layer)**	**Cycle**	**Direct Algorithm**
	Cradle (2o/2)	M'U2M' → Arrow B	(MU2) (M'U2) (M'U'M')
	Cross (4/0)	M'U2M' → Arrow F	(M'U2) (M'U2) (M'U'M')
	Stinger (2a/2)	M2 → Arrow F	(M2U') (M'U'M')
	Neighbors (2a/0)	M'UM' → Arrow F	U (M'U'M') U2 (M'U'M') (M'UM') U2 (M'U'M') U2 (rUR'U) (MU') (RU'R')
	Diagonal B (1/1)	M'UM' → Arrow B	(M'U'M') U' (MU'M')

	Diagonal F (1/1)	MU’M → Arrow F	(M’U’M) U’ (M’U’M’)
	Lower Pair (0/2)	M'UM' → Arrow B	(M’UM’) U’ (MU’M’) (M’U’M’) U (MU’M’)
	All Bad (4/2)	M'UM' → Diagonals	RU’r’U’ M’ UrUr’ M’U’M’-U2-M’U’MU’M’U’M’
	Upper Pair (2/0)	M’UM’ → Stinger	(M’UM) U' (M'U’M') (RUR'U') M' (URU'r')
	Arrow F (3/1)	M’UM’ (solved)	M’U’M’
	Arrow B (3/1)	MU’M’ (solved)	MU’M’

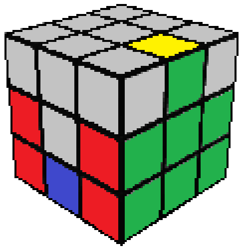
Step 4b: Solve the UL and UR Edges

For this tutorial, the UL and UR edges are the blue/yellow and green/yellow edges, respectively. To solve these two edges, we first move them both to the bottom layer. As we do this, we must make moves that preserve the edge orientation. Permitted moves are:

- U, U’, U2, and M2
- M and M’ only if they are followed with U2M or U2M’

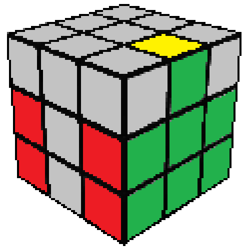
There are not many cases in this step. We will use moves like M’U2M to move edges between layers. The basic idea is to get one edge on the bottom, then insert the other edge across from it.

1 on top, 1 on DF



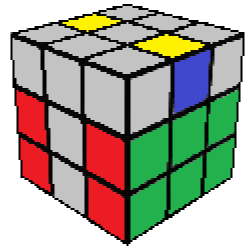
U’MU2M’

1 on top, 1 on DB



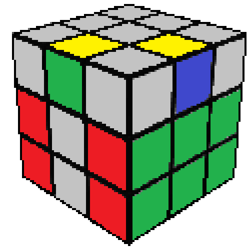
UM’U2M

2 on top, opposite



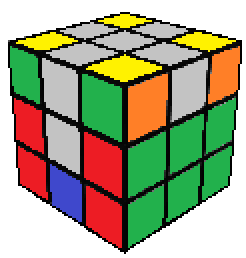
UM2

2 on top, adjacent

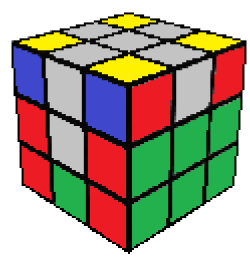


M2UM’U2M

Once the UL and UR edges are in the bottom layer, you will be able to see one of them on the front face. Note the color of this edge (in our case, blue or green), then turn the top layer so that the corner stickers facing you are the *opposite* color from the edge. In other words, if you can see a blue edge, you want to see green corners, and vice versa.

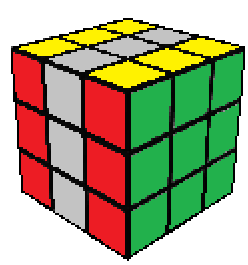


OR



Now simply use M2 to bring the UL and UR edges to the top layer, then use U or U’ to solve the L and R sides

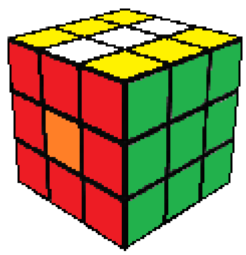
All that remains is to solve the last 4 edges around their centers, which only takes 3 or 4 turns.



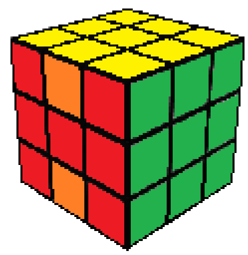
Step 4c: Solve the Last Four Edges

There are 3 basic cases for this last step: 2 dots, 4 dots, and a 3-edge cycle.

2 Dots

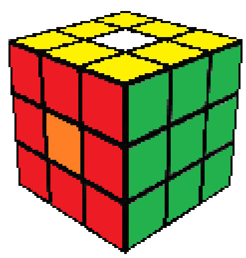


(U2 M2 U2)

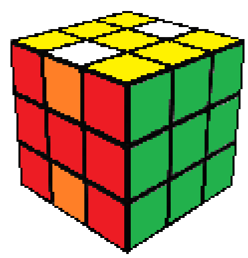


(M2 U2 M2 U2)

4 Dots



(E2 M’ E2 M)



(E2 M’ E2 M’)

3-Edge Cycle

3-edge cycles are a common case requiring only 4 or 5 turns to solve, but recognizing their best solution takes practice.

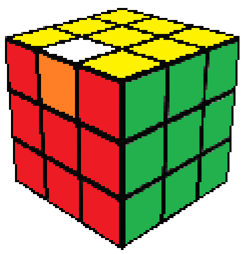
First, find the “unique edge” whose stickers do not align with its neighboring centers in the M slice. Make sure that the unique edge is in the top layer, using M2 if necessary.

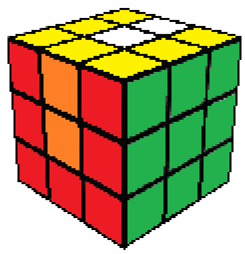
Next, look at the FU and BU stickers. They will be either the same color (e.g. red) or opposite colors (e.g. red and orange).

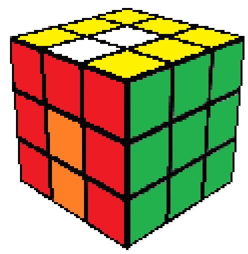
- Same color: do (MU2 MU2) or (M’U2 M’U2), whichever keeps the unique edge on top. A final M2 may be required to solve the cube.
- Opposite color: do (U2 M U2) or (U2 M’ U2), whichever keeps the unique edge on top. A final M or M’ turn solves the cube.

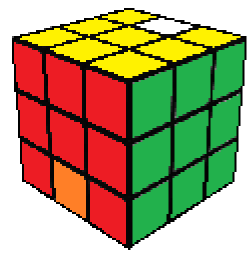
3-Cycle Recognition

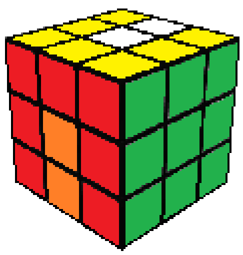
Some may prefer a more algorithmic approach to 3-cycles, so here are the eight possible 3-cycle cases in this color scheme. The solution to last four cases depends on the color (orange or red) of the BU edge sticker, which should be tracked in advance.

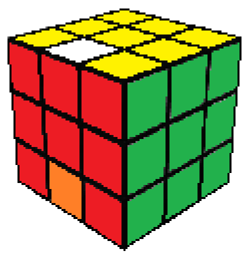

R: U2MU2M'

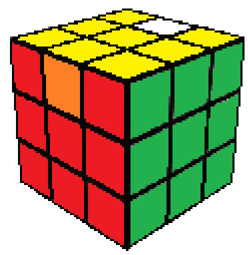

R: M2U2MU2M'

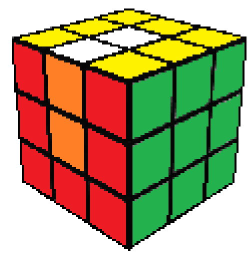

O: U2M'U2M'


O: M2U2M'U2M'


O: U2MU2M
R: M'U2M'U2


O: M2U2MU2M
R: M'U2MU2


O: MU2M'U2
R: U2M'U2M


O: MU2MU2
R: M2U2M'U2M