

The Roux Method

a complete guide to solving the Rubik's cube

Aayush Bajaj

October 19, 2026

the web version of this guide, with the same pictures, is at <https://abaj.ai/wiki/cubing/roux/>; the PDF is <https://abaj.ai/wiki/cubing/roux/the-roux-method.pdf>.

this is a working sheet for solving a 3x3 cube with the Roux method: hold the cube, find the picture that matches it, do the moves. everything here is drawn from [Gilles Roux's own description of his method](#), with his sequences machine-checked and every case drawn. where a sentence is his it is quoted; the rest is paraphrase.

the sheet assumes nothing beyond “I can turn the faces”. every symbol used is defined in the next section, and every word of cubing jargon is defined where it first appears and again in the glossary at the end.

1 notation

1.1 how to hold the cube

hold the cube with the **yellow centre on top**, the **green centre facing you**, so the **orange centre is on your right** and the **red centre on your left**. white is then on the bottom and blue at the back. the six faces are named by where they are, not by their colour:

letter	face	centre colour (the way we hold it)
U	up	yellow
D	down	white
F	front	green
B	back	blue
R	right	orange
L	left	red



the cube as every diagram on this page shows it: U on top, F at the front-left, R at the front-right



the same cube seen from the left: F at the front-right, L at the front-left (used only for the first block)

centres never move relative to each other, so “the green face” and “the front face” are the same thing **only while you hold the cube still**. the letters always mean positions. a **piece** is named by the faces it touches: the **UF** edge is the edge between the up and front faces, the **URF** corner is the corner where up, right and front meet. every piece has a home: the **UF** edge belongs where the up and front centres meet (yellow and green).

1.2 the turns

a letter on its own is a **quarter turn of that face, clockwise as you look straight at that face**. a prime (R') is the same quarter turn anticlockwise. a 2 ($R2$) is a half turn, direction irrelevant.

symbol	what turns	direction
U	the top layer	clockwise looking down on it
U'	the top layer	anticlockwise looking down on it
R	the right layer	clockwise looking at it from the right: its front edge goes up
R'	the right layer	its front edge comes down
L	the left layer	clockwise looking from the left: its front edge comes down
L'	the left layer	its front edge goes up
F	the front layer	clockwise as you look at it
B, D	the back layer, the bottom layer	clockwise looking at that face (rare in roux)
M	the middle slice between L and R, centres included	follows L : its top comes towards you and down
M'	the middle slice	follows R : its front goes up and away from you
M2	the middle slice	half turn
r	the right layer and the middle slice together	like R: the front goes up
r'	the right layer and the middle slice together	like R'
l	the left layer and the middle slice together	like L
U2, R2, M2, ...	any letter followed by 2	half turn of that layer, direction irrelevant



U: the top layer turns clockwise (seen from above)



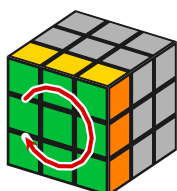
U': the top layer turns anticlockwise



R: the right layer turns clockwise (seen from the right): its front goes up



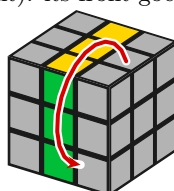
R': the right layer turns the other way: its front comes down



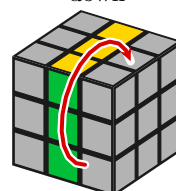
F: the front layer turns clockwise as you look at it



L: the left layer turns clockwise seen from the left: its front comes down



M: the middle slice follows L: its top comes towards you and down



M': the middle slice follows R: its front goes up and away



r: the right layer and the middle slice together (R and M' at once)



l: the left layer and the middle slice together (L and M at once)

in each picture the coloured pieces are the ones that move; the grey ones stay put.

1.3 the traps

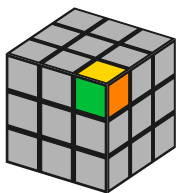
- **M goes the “wrong” way.** M follows L, not R. on a cube held yellow-up, green-front, M brings the top of the middle slice **towards you and down**, so the yellow centre ends up at the front. this is the single most common slip; when a sequence “doesn’t work”, check your **M** first.
- **r is not R.** r (lower case) turns two layers: the right face **and** the middle slice. R (upper case) turns only the right face. r is the same as doing R and then M'.
- **letters are positions, not pieces.** UF means “the slot at the top front”, whatever piece is sitting there now. when a case says “the UL edge is at DF”, it means the yellow-red edge piece is sitting in the

bottom-front slot.

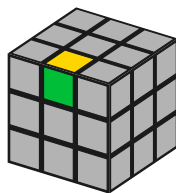
- **AUF** (“adjust the U face”) means: turn the top layer, by **U**, **U'** or **U2**, until the cube matches the diagram. the case diagrams on this page are drawn for one particular turn of the top layer; you find the matching turn by rotating the top only. never rotate the whole cube unless a step says so; nothing in roux needs it.
- **the whole-cube rotations** **x**, **y**, **z** (turn the entire cube like **R**, **U**, **F**) exist in cubing notation but are never used on this page.

2 the words

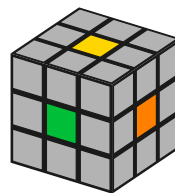
the method is simple but cubers describe it with words that mean nothing until you have seen them. every term is listed again in the glossary at the end.



a corner piece: three stickers



an edge piece: two stickers



the centres never move relative to each other

- **corner**: a piece with three stickers. there are eight.
- **edge**: a piece with two stickers. there are twelve.
- **centre**: the single-sticker piece in the middle of a face. the six centres never move relative to each other; they define which colour belongs on which face.
- **solved** (of a piece): sitting in its home slot with every sticker matching the centre it touches.



first block: the 1x2x3 on the bottom left (seen from the left)



second block: the mirror 1x2x3 on the bottom right



F2B (first two blocks) done: only the top layer and the middle slice are left

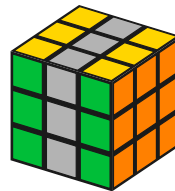
- **block**: a set of pieces that are solved relative to one another, so they move as a unit. a **1x2x3 block** is a centre, three edges and two corners filling one face's lower two rows.
- **first block (FB)**: the 1x2x3 built on the bottom-left: the red centre, the **DL**, **FL**, **BL** edges and the **DFL**, **DBL** corners.
- **second block (SB)**: the mirror block on the bottom-right: the orange centre, **DR**, **FR**, **BR**, **DFR**, **DBR**.
- **F2B** (“first two blocks”): both of them done. everything in the middle slice and the top layer is still free.



a square (1x2x2): a corner with its two edges around a centre; here the back-right one



a pair (1x1x2): a corner joined to an edge; here the front-right one



the M slice is free: the grey pieces are not solved until the very end

- **pair** (1x1x2): a corner joined to one edge, the two stickers they share on each face matching.
- **square** (1x2x2): a corner with both its edges around a centre: a pair plus one more edge. a 1x2x3 block is a square plus a pair.
- **the M slice**: the middle layer between L and R, holding four centres and four edges. while you build the blocks nothing in it matters, which is why roux can be done with R, r, U and M alone: “the M slice is free”.

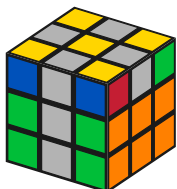


all four corners oriented: yellow on top

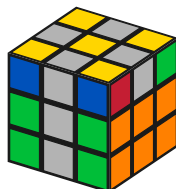


front-right corner misoriented: its yellow sticker faces the side

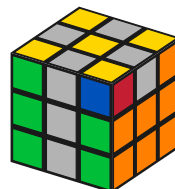
- **oriented** (of a top corner): its yellow sticker faces up. **misoriented**: its yellow sticker faces a side. a corner can be oriented in the wrong slot, and in the right slot but twisted.



headlights: the two corner stickers on one face have the same colour (front face here)



adjacent swap: the two back corners must trade places.
The front face shows headlights (a matching pair)



diagonal swap: two corners opposite each other must trade places. No face shows headlights

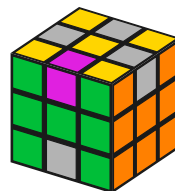
- **headlights**: on one side face, the two corner stickers have the same **non-yellow** colour. they tell you those two corners are correctly placed relative to each other.
- **bar**: on one side face, both corner stickers are **yellow**. a bar says those two corners are twisted so that their yellow stickers face the same side; the case names use it (“left bar”, “right bar”).
- **adjacent swap**: of the four top corners, two that sit **next to each other** need to trade places. with all four oriented you see headlights on exactly one face.
- **diagonal swap**: two corners **opposite each other** (across the top face) need to trade places. no face shows headlights.
- **permutation**: which slot each piece is in. **orientation**: which way round it sits in that slot. CMLL fixes both for the corners at once.



a good edge: its yellow (or white) sticker is on the top or the bottom face



a bad edge: its yellow sticker faces the side



how the case diagrams draw it: bad edges are magenta

- **good / bad edge** (edge orientation, “EO”): once the yellow centre is on top, an edge is **good** when its yellow or white sticker is on the top or the bottom face, and **bad** when that sticker faces a side. the six edges left at the end are all yellow-or-white edges, so every one of them can be read this way.
- **LSE** (“last six edges”): the final step, **UL**, **UR**, **UF**, **UB**, **DF**, **DB**, done with only **M** and **U**.
- **CMLL**: the one algorithmic step: “corners of the last layer”, solved while the **M** slice is ignored.

3 the method in four steps

1. **first block**: build the 1x2x3 block on the bottom-left, by looking and thinking. no algorithms.
2. **second block**: build the mirror block on the bottom-right, using only **R**, **r**, **U** and **M** so the first block survives.
3. **CMLL**: the four top corners, one algorithm from a set of 42. the middle slice is ignored.
4. **last six edges**: **M** and **U** moves only, in three small steps: make every edge good (4a), put the red and orange edges in (4b), finish the middle slice (4c).

Roux’s own summary of the idea: the first block “leaves the cube in a clean state”, and once it is built on the left “you can have an eye on every other piece without rotating the cube”. the two blocks cost about 10 moves each in practice, CMLL is one memorised sequence, and the last six edges flow.

4 step 1: the first block

4.1 what you are building

the red centre, the three edges around it in the bottom two rows (**DL**, **FL**, **BL**) and the two corners (**DFL**, **DBL**). the picture in the glossary shows it from the left. you look at it from the left while building, with the finished block ending on the bottom-left.

4.2 how to think about it

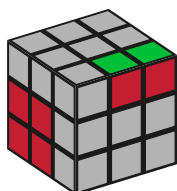
there are no algorithms here, and that is the point. Roux: “There are no strict rules, it completely depends on the situation.” his advice, paraphrased:

- **start with a square**. “Starting with a 1x2x2 block often is the best strategy”: pick a corner and the edge that goes with it, join them into a pair, add the centre’s neighbouring edge, and you have a square; then add the last pair.
- **look before you move**. “Try to imagine what it would do to the two other pieces. It’s important to know before where to expect them after the move.” in competition you get 15 seconds of inspection; this step is designed to use them.
- **always the same colours**. “my recommendation is to always take the same colors, because you don’t have enough time to explore many sets of five pieces”: red on the left, white below, every time. colour-neutral solving comes much later.

- **three edges first is a bad idea** “unless you can position them in one or two moves.”
- **pair pieces before placing them.** get the corner and its edge next to each other with their shared colours lined up, then insert them as one unit. this is what the four examples below do.
- while the right side is empty it is a free working area: pieces can pass through the bottom-right without cost.

4.3 worked examples

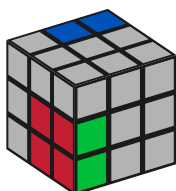
each card shows a cube in which only the first-block pieces are coloured; the moves put them home. these are the smallest cases; a real solve chains three or four of them.



front edge and front corner
are out

L' U L

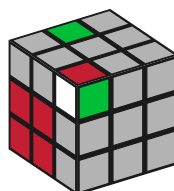
seen from the left



back edge and back corner
are out

L U' L'

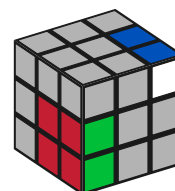
seen from the left



front edge and front corner
are out

L' U' L U

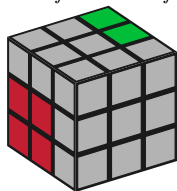
seen from the left



back edge and back corner
are out

l U2 l'

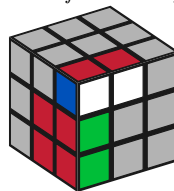
seen from the left



front edge and front corner
are out

L' U2 L

seen from the left



bottom edge and back edge
and back corner are out

F' L F

seen from the left

5 step 2: the second block

5.1 what you are building and the one rule

the orange centre with DR, FR, BR, DFR and DBR: the mirror image of the first block, on the bottom-right. Roux: “as soon as you do F, B or D, the L-side block becomes an endangered species.” so from now on you use only R, r, U and M (and their inverses). these four moves never touch the left block, and they are enough.

5.2 the plan

Roux’s strategy, which he calls the way to “always try to track and combine two elements”:

1. make a corner-edge **pair** (a corner with one of its edges).
2. add the missing edge: now you have a **square** (1x2x2) and you drop it into place on the right.
3. solve the last corner and edge, the **last pair**.

the pair in step 1 is brought to the right side “only when combined with the missing edge”: you make the square somewhere convenient (usually up top) and then one R turn places it.

below, his tables for steps 2 and 3, redrawn. in every card only the pieces that matter are coloured: the first block (green stickers on the left of the front face), the orange centre, and the pieces being placed. grey means “whatever is there”. the edge position is named by **where its orange sticker faces, then where**

its other sticker faces: “orange facing right / other sticker up” is the edge in the top-right slot with orange on the right face.

5.3 build the square

Roux’s table solves the **front** square (DFR corner, FR and DR edges). start from the pair you have made, which lies in one of four positions, then find the card for where the missing edge is. his sequences leave the square built in the top layer; the final R' or R2 on each card (not in his table) drops it into place, so the whole card leaves the front square solved. if you cannot find your case, “rotate the U-layer or the R-layer (a layer the pair does not belong to), or consider a symmetrical case.”

5.3.1 pair lying along the front-bottom, orange facing front; missing the orange-white edge



edge at top-right, orange facing right

r2 U' R U2 R'

Roux: r2 U' R U2; R' places the square



edge at top-right, orange facing up

U' r' U' r' U R'

Roux: U' r' U' r' U; R' places the square



edge at front-right, orange facing right

r2 U' R2 U2 R'

Roux: r2 U' R2 U2; R' places the square



edge at front-right, orange facing front

R U' r2 U r U' R2

Roux: R U' r2 U r U'; R2 places the square



edge at back-right, orange facing right

r2 U R'

Roux: r2 U; R' places the square



edge at back-right, orange facing back

r' U' r' U' R' U2 R'

Roux: r' U' r' U' R' U2; R' places the square



edge at bottom-right, orange facing right

r U R' U' r U R'

Roux: r U R' U' r U; R' places the square



edge at bottom-right, orange facing down

r2 U M' U R' U' R'

Roux: r2 U M' U R' U'; R' places the square



edge at bottom-back, orange facing down

r' U r' U R'

Roux: r' U r' U; R' places the square



edge at bottom-back, orange facing back

r2 U' R' U2 R'

Roux: r2 U' R' U2; R' places the square

5.3.2 pair lying along the front-bottom, white facing front; missing the orange-green edge



edge at top-right, orange
facing right

r U' R2

*Roux: r U'; R2 places the
square*



edge at top-right, orange
facing up

U r U R U2 R2

*Roux: U r U R U2; R2 places
the square*



edge at front-right, orange
facing right

r2 U' R U r' U' R2

*Roux: r2 U' R U r' U'; R2
places the square*



edge at front-right, orange
facing front

r U r U R U R2

*Roux: r U r U R U; R2 places
the square*



edge at back-right, orange
facing right

r U R' U2 R2

*Roux: r U R' U2; R2 places the
square*



edge at back-right, orange
facing back

r' U r U r U' R2

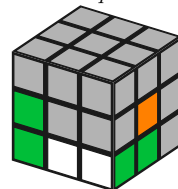
*Roux: r' U r U r U'; R2 places
the square*



edge at bottom-right,
orange facing right

r U R2 U2 R2

*Roux: r U R2 U2; R2 places
the square*



edge at bottom-right,
orange facing down

R2 U r U' R' U2 R'

*Roux: R2 U r U' R' U2; R'
places the square*



edge at bottom-back,
orange facing down

M U M' r U' R2

*Roux: M U M' r U'; R2 places
the square*



edge at bottom-back,
orange facing back

r U' M' U' R U R2

*Roux: r U' M' U' R U; R2
places the square*

5.3.3 pair in the top layer at the front-left, orange facing up; missing the orange-green edge



edge at bottom-right,
orange facing right

U2 r' U' R2

*Roux: U2 r' U'; R2 places the
square*



edge at bottom-right,
orange facing down

U R F' U' F R'

*Roux: U R F' U' F; R' places
the square*



edge at top-left, orange
facing left

U2 r' U R2 U2 R2

*Roux: U2 r' U R2 U2; R2
places the square*



edge at top-left, orange
facing up

r' U' r2 U r' U R'

*Roux: r' U' r2 U r' U; R'
places the square*



edge at top-back, orange
facing up

U R' F' U' F R'

*Roux: U R' F' U' F; R' places
the square*



edge at top-back, orange
facing back

U R2 U r' U' R2

*Roux: U R2 U r' U'; R2 places
the square*



edge at bottom-front,
orange facing down

r U2 R' U R U2 R2

*Roux: r U2 R' U R U2; R2
places the square*



edge at bottom-front,
orange facing front

U2 r2 U r U' R2

*Roux: U2 r2 U r U'; R2 places
the square*



edge at bottom-back,
orange facing down

U2 r' U R U2 R2

*Roux: U2 r' U R U2; R2 places
the square*



edge at bottom-back,
orange facing back

U2 r2 U' r U' R2

*Roux: U2 r2 U' r U'; R2 places
the square*

5.3.4 pair in the top layer at the front-left, green facing up; missing the orange-white edge



edge at bottom-right,
orange facing right

R U' R'

*Roux: R U'; R' places the
square*



edge at bottom-right,
orange facing down

R2 U' r U M2 U' R2

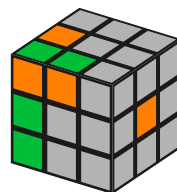
*Roux: R2 U' r U M2 U'; R2
places the square*



edge at top-left, orange
facing left

U' R' U' R U R2 U' R'

*Roux: U' R' U' R U R2 U'; R'
places the square*



edge at top-left, orange
facing up

U' M' U R' U' R'

*Roux: U' M' U R' U'; R'
places the square*



edge at top-back, orange
facing up

U2 r U r' U R'

*Roux: U2 r U r' U; R' places
the square*



edge at top-back, orange
facing back

U R' U2 R'

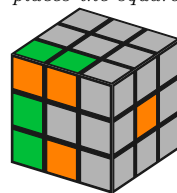
*Roux: U R' U2; R' places the
square*



edge at bottom-front,
orange facing down

U' R U M' U' R2

*Roux: U' R U M' U'; R2 places
the square*



edge at bottom-front,
orange facing front

U' M2 U R' U' R'

*Roux: U' M2 U R' U'; R'
places the square*



edge at bottom-back,
orange facing down

U' M U R' U' R'

*Roux: U' M U R' U'; R' places
the square*



edge at bottom-back,
orange facing back

U' r U M' U' R2

*Roux: U' r U M' U'; R2 places
the square*

5.4 the last pair, at the back

with the front square in, the remaining pieces are the DBR corner (orange, white, blue) and the BR edge (orange, blue). Roux's table for this is written for the last pair at the **front**; the cards below are its mirror image (every move reversed, front and back exchanged), checked by the same simulator, so they follow on from the square table above. Roux considers "six cases for the last corner (two locations and three orientations)", and for the edge only the positions you cannot trivially fix with a U or M turn: "reducing this number is only a matter of applying one or two moves."

5.4.1 corner in the top layer, orange facing up



edge at top-right, orange
facing right

U R' U R U' R' U' R

mirror of Roux: U' R U' R' U

R U R'



edge at top-front, orange
facing front

R' U' R

mirror of Roux: R U R'

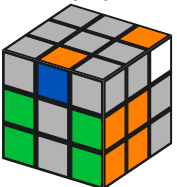


edge at top-right, orange
facing up

U' M' U M r' U' R

mirror of Roux: U M U' M' r

U R'



edge at top-front, orange
facing up

r' U2 M2 U R

mirror of Roux: r U2 M2 U' R'



edge at top-left, orange
facing left

U R' U' R U' R' U' R

mirror of Roux: U' R U R' U

R U R'



edge at back-right, orange
facing right

R' U' M' U' M U R

mirror of Roux: R U M U M' U' R'



edge at top-left, orange
facing up

U M' U' M r' U' R

mirror of Roux: U' M U M' r

U R'



edge at back-right, orange
facing back

R' U' M' U M2 U' R

mirror of Roux: R U M U' M2 U R'

5.4.2 corner in the top layer, orange facing right



edge at top-right, orange
facing right

U' R' U R

mirror of Roux: U R U' R'



edge at top-front, orange
facing front

M U M' U R' U2 R

mirror of Roux: M' U' M U' R U2 R'



edge at top-right, orange
facing up

U M U2 M r' U R

mirror of Roux: U' M' U2 M' r

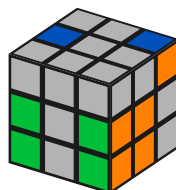
U' R'



edge at top-front, orange
facing up

M U' M r' U R

mirror of Roux: M' U M' r U' R'



edge at top-left, orange
facing left

U M' U2 r' U R

mirror of Roux: U' M U2 r U

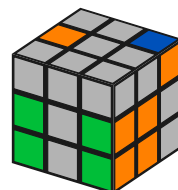
R'



edge at back-right, orange
facing right

U R' U R U R' U2 R

mirror of Roux: U' R U' R' U' R U2 R'



edge at top-left, orange
facing up

B U B'

mirror of Roux: F' U' F



edge at back-right, orange
facing back

U2 R' U' M U' R

mirror of Roux: U2 R U M' U R'

5.4.3 corner in the top layer, orange facing back



edge at top-right, orange
facing right

R' U2 R U R' U' R

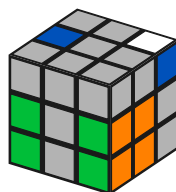
mirror of Roux: R U2 R' U' R



edge at top-right, orange
facing up

U' M U' R' F R' F' R2

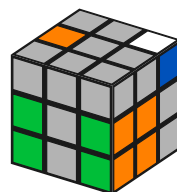
mirror of Roux: U M' U R B'



edge at top-left, orange
facing left

R' F' U2 F R

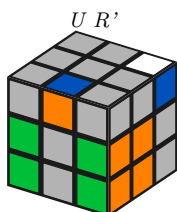
mirror of Roux: R B U2 B' R'



edge at top-left, orange
facing up

R' U r U2 r' M U' R

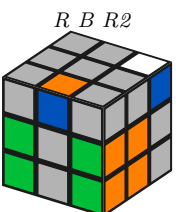
mirror of Roux: R U' r' U2 r



edge at top-front, orange
facing front

U' R' U' F R' F' R2

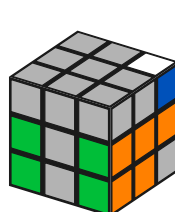
*mirror of Roux: U R U B' R B
R2*



edge at top-front, orange
facing up

B L U2 L' B'

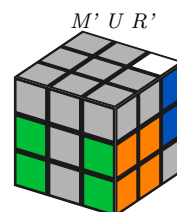
mirror of Roux: F' L' U2 L F



edge at back-right, orange
facing right

R' U' R U R' U' R U R'

U' R
*mirror of Roux: R U R' U' R
U R' U' R U R'*



edge at back-right, orange
facing back

R' U2 R U r' U' R

*mirror of Roux: R U2 R' U' r
U R'*

5.4.4 corner in its slot, solved



edge at top-front, orange
facing front

r' U M' U' R

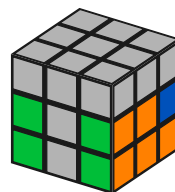
mirror of Roux: r U' M U R'



edge at top-front, orange
facing up

U R' U M U' R

*mirror of Roux: U' R U' M' U
R'*



edge at back-right, orange
facing back

R' U' M' U2 M2 U' R

*mirror of Roux: R U M U2 M2
U R'*

5.4.5 corner in its slot, twisted: orange facing down



edge at top-front, orange
facing front

U' R' U R U' R' U R

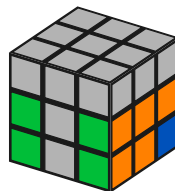
*mirror of Roux: U R U' R' U
R U' R'*



edge at top-front, orange
facing up

R' U R U' M' R' U R

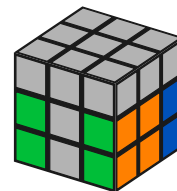
*mirror of Roux: R U' R' U M
R U' R'*



edge at back-right, orange
facing right

R' U' R U R' U2 R U R'

U' R
*mirror of Roux: R U R' U' R
U2 R' U' R U R'*



edge at back-right, orange
facing back

R' U r M' U' r' U R

*mirror of Roux: R U' r' M U r
U' R'*

5.4.6 corner in its slot, twisted: orange facing back



edge at top-front, orange facing front

U' R' U' R U R' U' R

mirror of Roux: U R U R' U' R U R'



edge at top-front, orange facing up

r' U2 r U2 r' U' R

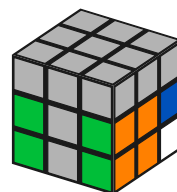
mirror of Roux: r U2 r' U2 r U R'



edge at back-right, orange facing right

R' U' r U' r2 F R F' R

mirror of Roux: R U r' U r2 B' R' B R'



edge at back-right, orange facing back

R' U' r U M r' U' R

mirror of Roux: R U r' U' M' r U R'

Roux's note on all of this: "you don't need to learn anything by heart, because most of the configurations are really easy to solve. Just look at them a few times, one by one, and you'll realize it becomes intuitive fast."

6 step 3: CMLL, the four top corners

6.1 what it does

after two blocks, the unsolved pieces are the four corners of the top layer and the six edges that are left (four in the middle slice, two on the top). CMLL solves the four corners completely, position **and** twist, with one sequence, and does not care what happens to the six edges. Roux: "In this method, U-edges and M-slice are free." he describes it as "only" 24 unique sequences plus their mirrors and inverses; the modern count, with one sequence per case, is 42.

6.2 how to find your case

look only at the four top corners and their twelve stickers. ignore every edge and every centre except that yellow is on top.

1. **count the yellow stickers on top** and note where they are. that gives one of eight **orientation patterns** (Roux's rows A to H). the eight pictures below show them; the other stickers are greyed out because they do not matter for this step.
2. **within the pattern, look at the side stickers** of the four corners: where the remaining yellow stickers are (two on one face make a **bar**), and whether any face shows **headlights** (two matching non-yellow corner stickers). the cue under each card says what to look for in words.
3. **AUF**: turn the top layer until your corners match the card, then do the moves. the corners are done; the top layer may need a final **U** turn at the very end of the solve, not now.



O: all four oriented (Roux: A)



H: none oriented, yellows in two opposite pairs (Roux: H)



Pi: none oriented, two yellows on one face (Roux: G)



U: two adjacent oriented, the other yellows on one face (Roux: E)



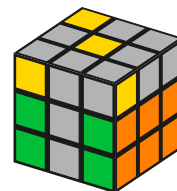
T: two adjacent oriented,
the other yellows on
opposite faces (Roux: F)



L: two diagonal oriented
(Roux: D)



S: one oriented, sune (Roux:
C)



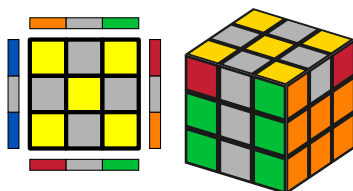
AS: one oriented, anti-sune
(Roux: B)

6.3 how to read a card

every case has two drawings of the **same** cube. the flat card on the left is the top face seen from above with the four side faces folded out as thin bars: the bar above the square is the back face, below it the front, left and right as you would expect. the 3D view on the right shows the top, front and right faces as you see them in your hands; the back and left stickers are only in the flat card. grey stickers do not matter. the line in *italics* is the cue; the **bold** line is the algorithm; where Gilles Roux gives a sequence for the same case that works without re-holding the cube it is listed as “Roux:” underneath. the two sequences solve the same case; use whichever your fingers prefer.

every card is generated by a cube simulator that applies the inverse of the printed algorithm to a solved cube and reads off the stickers, so a drawing cannot disagree with its algorithm; the same program checks that every sequence leaves both blocks intact.

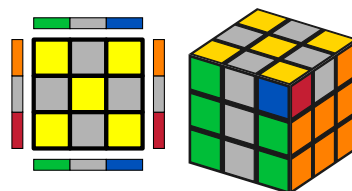
6.4 the 42 cases



O1: adjacent swap

*all four yellow on top; headlights on the
left*

R U R' F' R U R' U' R' F R2 U'
R'

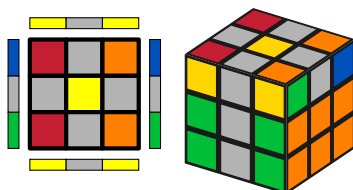


O2: diagonal swap

all four yellow on top; no headlights

F R U' R' U' R U R' F' R U R' U'
R' F R F'

Roux: R2 U' R F' R' U r2 U F U F'

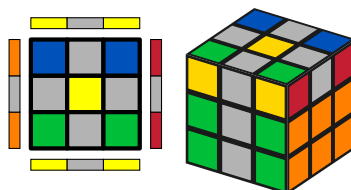


H1: columns

*no yellow on top; yellow side stickers:
both on the back, both on the front; no
headlights; left and right faces show the
same colours*

R U2 R' U' R U R' U' R U' R'

Roux: U R' F R U R' F R U' R' F' R

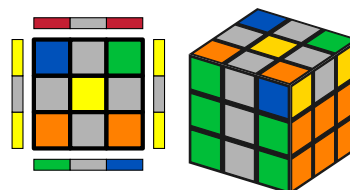


H2: rows

*no yellow on top; yellow side stickers:
both on the back, both on the front;
headlights on the left and right*

F R U R' U' R U R' U' R U R' U'
F'

Roux: U F R U R' U' R U R' U' R U R'
U' F'

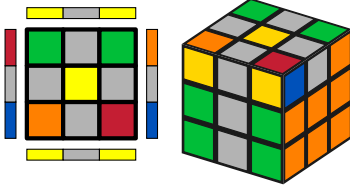


H3: columns alt

*no yellow on top; yellow side stickers:
both on the left, both on the right;
headlights on the back*

R U2 R2 F R F' U2 R' F R F'

Roux: F U R U' R' U F' U' R' F' R

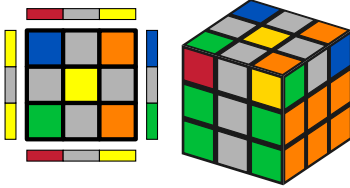


H4: rows alt

*no yellow on top; yellow side stickers:
both on the back, both on the front; no
headlights; front and back faces show the
same colours*

r U' r2 D' r U' r' D r2 U r'

Roux: **R U R' U R U L' U R' U' L**

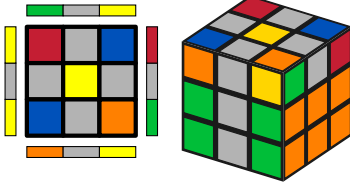


Pi1: right bar

*no yellow on top; yellow side stickers: one
on the back, both on the left, one on the
front; no headlights; front and back faces
show the same colours*

F R U R' U' R U R' U' F'

Roux: **F R U R' U' R U R' U' F'**

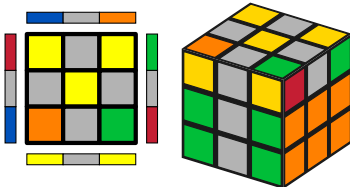


Pi4: forward slash

*no yellow on top; yellow side stickers: one
on the back, both on the left, one on the
front; no headlights; right face:
neighbouring colours*

R U2 R' U' R U R' U2 R' F R F'

Roux: **R' U L' U R U' L U2 R' U R**

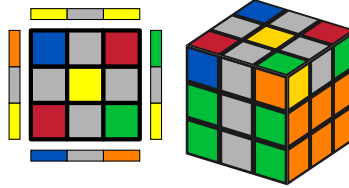


U1: forward slash

*two yellow on top: the back two; yellow
side stickers: both on the front; no
headlights; back face: neighbouring colours*

R2 D R' U2 R D' R' U2 R'

Roux: **R' U L' U R U' L U2 R' U R**

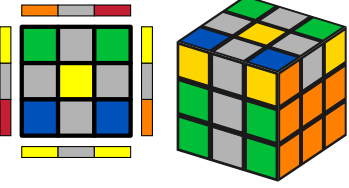


Pi2: back slash

*no yellow on top; yellow side stickers:
both on the back, one on the left, one on
the right; no headlights*

F R' F' R U2 R U' R' U R U2 R'

Roux: **R' F R' F' R2 U2 B' R B R'**

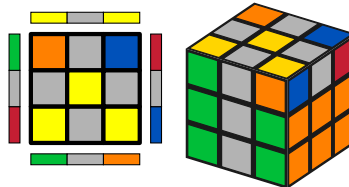


Pi5: columns

*no yellow on top; yellow side stickers: one
on the left, one on the right, both on the
front; no headlights*

r U' r2 D' r U r' D r2 U r'

Roux: **R' F R' F' R2 U2 B' R B R'**

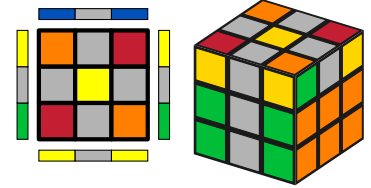


U2: back slash

*two yellow on top: the front two; yellow
side stickers: both on the back; no
headlights*

R2 D' R U2 R' D R U2 R

Roux: **R' F R U R' F R U F U2 F'**

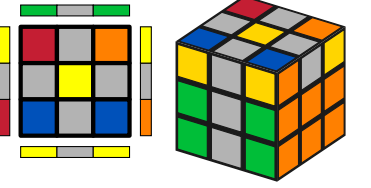


Pi3: x-checkerboard

*no yellow on top; yellow side stickers: one
on the left, one on the right, both on the
front; headlights on the back; left and right
faces show the same colours*

R' F R U F U' R U R' U' F'

Roux: **U' R' F R U F U' R U R' U' F'**

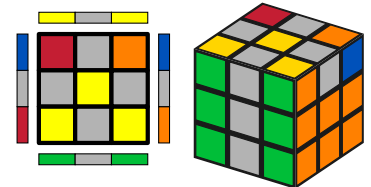


Pi6: left bar

*no yellow on top; yellow side stickers: one
on the left, one on the right, both on the
front; headlights on the back; back face:
green left and green right*

R' U' R' F R F' R U' R' U2 R

Roux: **U2 R' U' R U' R' U F' U F R**

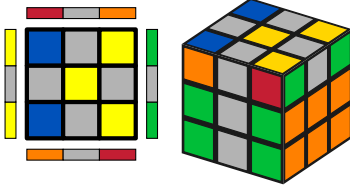


U3: front row

*two yellow on top: the front two; yellow
side stickers: both on the back; headlights
on the front*

R2 F U' F U F2 R2 U' R' F R

Roux: **R2 F' U F U' F' U' F2 R F' R**

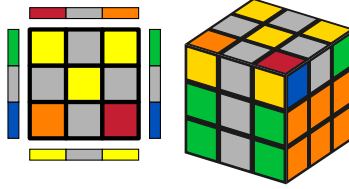


U4: rows

two yellow on top: the right two; yellow side stickers: both on the left; headlights on the right

F R2 D R' U R D' R2 U' F'

Roux: U' F R2 D R' U R D' R2 U' F'

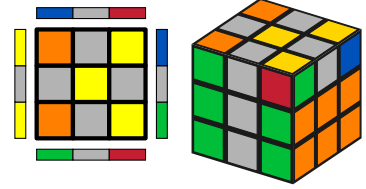


U5: x-checkerboard

two yellow on top: the back two; yellow side stickers: both on the front; no headlights; left and right faces show the same colours

r U' r' U r' D' r U' r' D r

Roux: R B' U R' B' R U' R' B

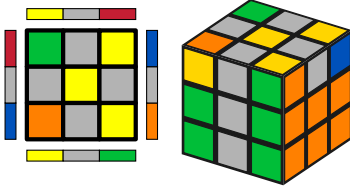


U6: back row

two yellow on top: the right two; yellow side stickers: both on the left; no headlights

F R U R' U' F'

Roux: U' F R U R' U' F'

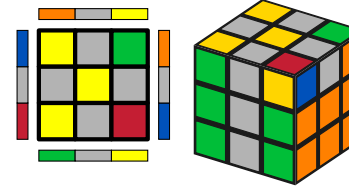


T1: left bar

two yellow on top: the right two; yellow side stickers: one on the back, one on the front; no headlights

R U R' U' R' F R F'

Roux: U' R U R' U' R' F R F'

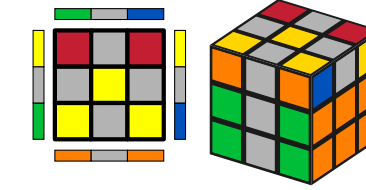


T2: right bar

two yellow on top: the left two; yellow side stickers: one on the back, one on the front; no headlights

L' U' L U L F' L' F

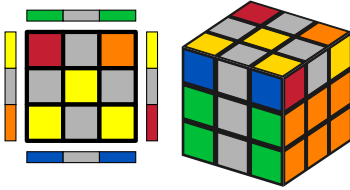
Roux: U' R' U' R U R B' R' B



T3: rows

two yellow on top: the front two; yellow side stickers: one on the left, one on the right; headlights on the front

F R' F R2 U' R' U' R U R' F2

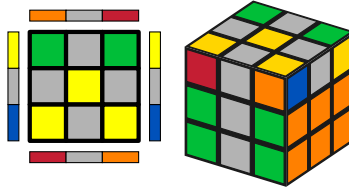


T4: front row

two yellow on top: the front two; yellow side stickers: one on the left, one on the right; headlights on the back and front

r' U r U2 R2 F R F' R

Roux: R' U r U2 R2 F R F' R

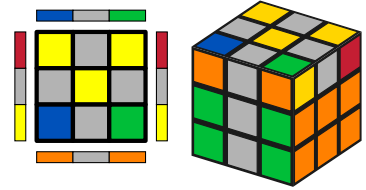


T5: back row

two yellow on top: the front two; yellow side stickers: one on the left, one on the right; no headlights

r' D' r U r' D r U' r U r'

Roux: F R' U' R F' R' U F' R

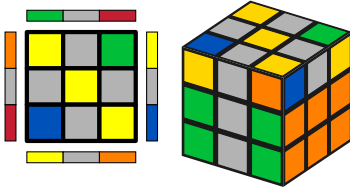


T6: columns

two yellow on top: the back two; yellow side stickers: one on the left, one on the right; headlights on the front

r2 D' r U r' D r2 U' r' U' r

Roux: R' F R F' R U2 R' U' F' U' F

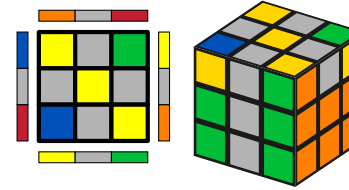


L1: mirror

two yellow on top: diagonal (back-left and front-right); yellow side stickers: one on the right, one on the front; no headlights; back face: green left and red right

F R U' R' U' R U R' F'

Roux: R B L B' R' B L' B'

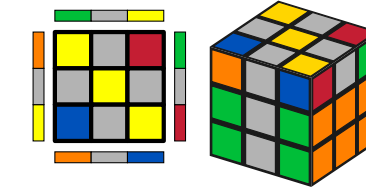


L2: inverse

two yellow on top: diagonal (back-left and front-right); yellow side stickers: one on the right, one on the front; no headlights; back face: orange left and red right

F R' F' R U R U' R'

Roux: R U' r' U' F' U F

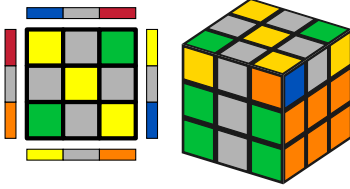


L3: pure

two yellow on top: diagonal (back-left and front-right); yellow side stickers: one on the back, one on the left; no headlights; back face: green left and yellow right

R U2 R' U' R U R' U' R U' R U' R

Roux: R' D R' D R' U R D' R D' R



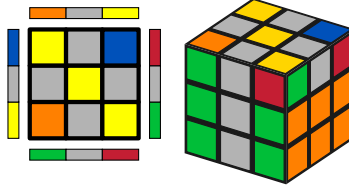
L4: front commutator

two yellow on top: diagonal (back-left and front-right); yellow side stickers: one on the right, one on the front; no headlights; back face: blue left and red right

R U2 R D R' U2 R D' R2

Roux: U' L' B' R B2 L' B R' B' L B'

L



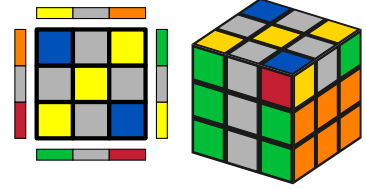
L5: diagonal

two yellow on top: diagonal (back-left and front-right); yellow side stickers: one on the back, one on the left; no headlights; back face: orange left and yellow right

R' U' R U R' F' R U R' U' R' F

R2

Roux: R U R2 F2 r F R' F2 R

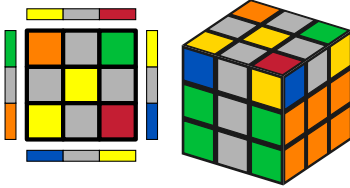


L6: back commutator

two yellow on top: diagonal (back-right and front-left); yellow side stickers: one on the back, one on the right; no headlights

R' U2 R' D' R U2 R' D R2

Roux: R B L' B2 R B' L B R' B R'

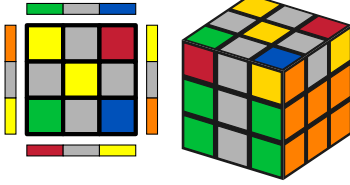


S1: left bar

one yellow on top (front-left); yellow side stickers: one on the back, one on the right, one on the front; no headlights; left face: green back and orange front

R U R' U R U2 R'

Roux: R U R' U R U2 R'

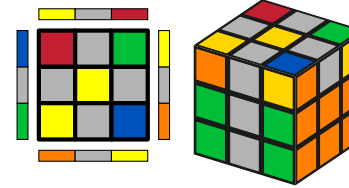


S4: columns

one yellow on top (back-left); yellow side stickers: one on the left, one on the right, one on the front; no headlights

R' U' R U' R2 F' R U R U' R' F
U2 R

Roux: L U' R' U F' R B L' U L B' L'

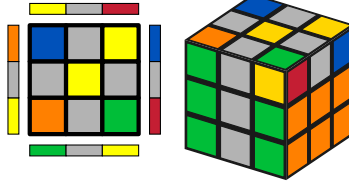


S2: x-checkerboard

one yellow on top (front-left); yellow side stickers: one on the back, one on the right, one on the front; no headlights; left face: blue back and green front

L' U2 L U2 L F' L' F

Roux: L' U2 L U2 L F' L' F

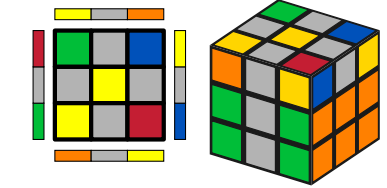


S5: right bar

one yellow on top (back-right); yellow side stickers: one on the back, one on the left, one on the front; no headlights

R U R' U R' F R F' R U2 R'

Roux: R' U2 R B' R B R' U R' U R

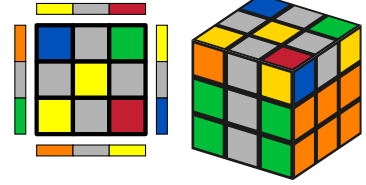


S3: forward slash

one yellow on top (front-left); yellow side stickers: one on the back, one on the right, one on the front; no headlights; left face: red back and green front

F R' F' R U2 R U2 R'

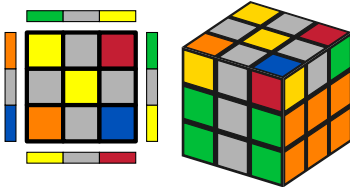
Roux: F R' F' R U2 R U2 R'



S6: back slash

one yellow on top (front-left); yellow side stickers: one on the back, one on the right, one on the front; no headlights; left face: orange back and green front

R U' L' U R' U' L

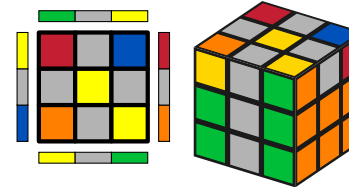


AS1: right bar

one yellow on top (back-left); yellow side stickers: one on the back, one on the right, one on the front; no headlights

R' U' R U' R' U2 R

Roux: R' U' R U' R' U2 R

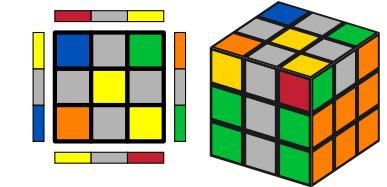


AS2: columns

one yellow on top (front-right); yellow side stickers: one on the back, one on the left, one on the front; no headlights; right face: red back and orange front

R2 D R' U R D' R' U R' U' R U'
R'

Roux: L' U R U' B R' F' L U' L' F L

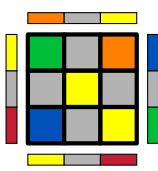


AS3: back slash

one yellow on top (front-right); yellow side stickers: one on the back, one on the left, one on the front; no headlights; right face: orange back and green front

F' L F L' U2 L' U2 L

Roux: B' R B R' U2 R' U2 R

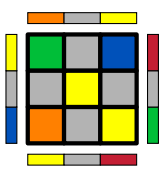


AS4: x-checkerboard

one yellow on top (front-right); yellow side stickers: one on the back, one on the left, one on the front; no headlights; right face: blue back and green front

R U2 R' U2 R' F R F'

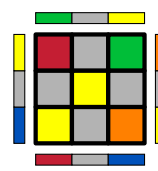
Roux: L U2 L' U2 L' B L B'



AS5: forward slash

one yellow on top (front-right); yellow side stickers: one on the back, one on the left, one on the front; no headlights; right face: red back and green front

L' U R U' L U R'



AS6: left bar

one yellow on top (front-left); yellow side stickers: one on the back, one on the left, one on the right; no headlights

R' U' R U' R' U R' F R F' U R

Roux: R U2 R' F R' F' R U' R U' R'

7 step 4: the last six edges

six edges remain: UL, UR, UF, UB, DF, DB, and the four centres of the middle slice may be turned. from here **every move is M or U**. Roux: “You should need less than 15 moves on average in this step, but don’t be afraid of MU!” the step has three parts: make every edge good (4a), put the red and orange edges home (4b), finish the middle slice (4c).

7.1 4a: edge orientation

7.1.1 what good and bad mean

turn the middle slice (**M** or **M'**) until the **yellow centre is on top** and white at the bottom. now look at each of the six edges:

- an edge is **good** if its yellow or white sticker is on the top face or on the bottom face;
- an edge is **bad** if its yellow or white sticker faces a side.

that is the whole definition. the colour of the other sticker does not matter, and whether the edge is in the right slot does not matter: a white edge sitting on top is good, it is just in the wrong place, and 4b or 4c will move it. Roux’s own wording is the same rule stated through the centres: a middle-slice edge “is correctly oriented if a facelet matches the adjacent M-center’s color or the opposite M-center’s color”, and for the two edges with a red or orange sticker, that sticker must be “on F/B/L/R-side”.

three facts make the counting easy:

- bad edges always come in an **even** number: 0, 2, 4 or 6.
- **U** turns never change whether an edge is good or bad; they only move the top ones around.
- every **M** or **M'** quarter turn flips all four edges of the middle slice between good and bad (the two side edges **UL** and **UR** are not in the slice and are unaffected).

so: count the bad edges on top, count the bad edges below, and if the total comes out odd you have missed one (usually the hidden one at **DB**).

7.1.2 the twelve orientation cases

the case is decided by **how many bad edges are on top, how many below, and how they sit**. the cards are drawn after an **AUF** that puts the bad top edges where the card shows them; bad edges are magenta, good edges grey. the bottom-back edge cannot be seen in the drawing; the note under the card says when it is bad. every sequence ends with the yellow centre back on top and all six edges good.



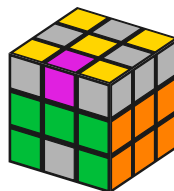
all good: skip
nothing to do



one on top, one below at
the front

U M U M U2 M' U M'

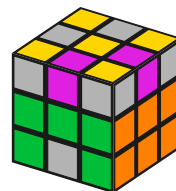
bad: *DF UF*



one on top, one below at
the back

M U M' U M U M'

bad: *DB UF* (*DB is hidden
behind*)



two on top, next to each
other

M U M U2 M U M

bad: *UF UR*



two on top, opposite

M U M' U' M U M'

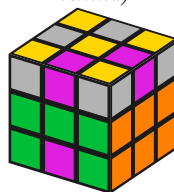
bad: *UB UF*



two below, none on top

M U M U' M' U M'

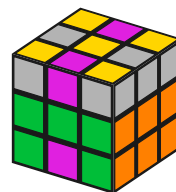
bad: *DB DF* (*DB is hidden
behind*)



two on top next to each
other, two below

M2 U' M U M

bad: *DB DF UF UR* (*DB is
hidden behind*)



two on top opposite, two
below

U M U2 M U2 M' U M'

bad: *DB DF UB UF* (*DB is
hidden behind*)



three on top, one below at
the front (arrow)

M' U M

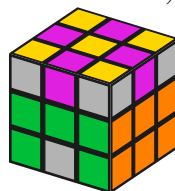
bad: *DF UF UL UR*



three on top, one below at
the back

U2 M U M'

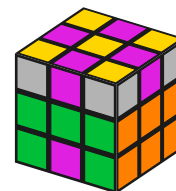
bad: *DB UF UL UR* (*DB is
hidden behind*)



four on top

M U2 M U2 M U M

bad: *UB UF UL UR*



all six bad

M U M U M' U M U2 M U M

bad: *DB DF UB UF UL UR*
(*DB is hidden behind*)

7.1.3 a worked example

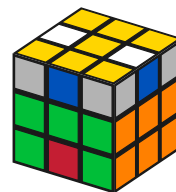
the cube looks like this: seven yellow stickers on top, one white, one red; seven white below, one yellow, one blue. that is a 4a case: the red-showing edge on top is bad (its yellow faces the side), the blue-showing edge below is bad (its white faces the side), and the white edge on top and the yellow edge below are good but misplaced, which is not this step's problem.



the cube as it is (corner side
stickers greyed): seven
yellow on top with one
white and one red; the red
edge is at the front; below it
the front edge shows white
at the side and blue
underneath



read as a 4a case: two bad
edges, one on top (front)
and one below (front)



after **U M U M U2 M' U M'**:
all six edges good; the white
edge on top is now a 4b job

turn the top (U moves only) until the red-showing edge is at the front, look whether the blue-showing edge is at the front or the back of the bottom, and take that card: “one on top, one below at the front” or “one on top, one below at the back”.

7.1.4 Roux's way of doing it

Roux does not enumerate the cases. his instruction is iterative: "Adjust U and/or rotate the whole cube, your pattern belongs to one of three categories below ... Apply the sequence and the pattern will change. Do it again, as long as wrong orientations remain." the three seeds are $M' U2 M'$ for two bad edges on top opposite each other, $M2$ for the pattern where the four middle-slice edges are bad, and $M' U M'$ for three bad on top with one below (the "arrow"). every row of the table above reduces to a short run of these; once the table is in your fingers you will find yourself steering towards the arrow rather than looking things up.

7.2 4b: the red and orange edges

7.2.1 what to do

after 4a all six edges are good. now put the yellow-red edge (UL) and the yellow-orange edge (UR) into their slots on the left and right of the top layer, **without making any edge bad again**. Roux: "U and M2 moves are okay, but if you perform a M/M' move, U2M/M' must follow." his cases are "very few possible combinations, zero recognition time".

first **line up the top layer with the blocks**: turn U until the four top corners sit over the matching blocks (the corner colours match the red and orange sides). now find the red edge and the orange edge: each is in one of the six slots, and that pair of positions is the case. the cards colour only those two edges; both are good, so you see yellow on top or bottom and red or orange on a side. a sequence ending in U or U' is part of the case: it puts the corners back in line.

7.2.2 the thirty cases



red edge top-left; orange
edge top-right
already there



red edge top-left; orange
edge top-front
M2 U' M' U2 M' U'



red edge top-left; orange
edge top-back
M2 U M U2 M U



red edge top-left; orange
edge bottom-front
U M U2 M U M2



red edge top-left; orange
edge bottom-back
U' M' U2 M' U' M2

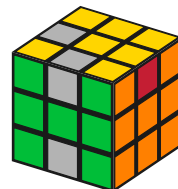
hidden behind



red edge top-right; orange
edge top-left
U M2 U2 M2 U



red edge top-right; orange
edge top-front
M2 U M U2 M' U M2



red edge top-right; orange
edge top-back
M2 U' M' U2 M U' M2



red edge top-right; orange
edge bottom-front
U' M' U2 M U'



red edge top-right; orange
edge bottom-back
U M U2 M' U

hidden behind



red edge top-front; orange
edge top-left
M2 U' M U2 M' U' M2



red edge top-front; orange
edge top-right
M2 U M' U2 M' U



red edge top-front; orange
edge top-back

M2 U' M2 U



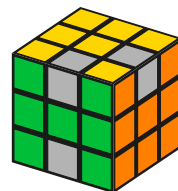
red edge top-front; orange
edge bottom-front

U2 M' U2 M' U M2 U'



red edge top-front; orange
edge bottom-back

M U2 M U M2 U



red edge top-back; orange
edge top-left

M2 U M' U2 M U M2



red edge top-back; orange
edge top-right

M2 U' M U2 M U'



red edge top-back; orange
edge top-front

M2 U M2 U'



red edge top-back; orange
edge bottom-front

M' U2 M' U' M2 U'



red edge top-back; orange
edge bottom-back

U2 M U2 M U' M2 U



red edge bottom-front;
orange edge top-left

U M' U2 M U



red edge bottom-front;
orange edge top-right

U' M U2 M U' M2



red edge bottom-front;
orange edge top-front

U2 M' U2 M' U' M2 U



red edge bottom-front;
orange edge top-back

M' U2 M' U M2 U



red edge bottom-front;
orange edge bottom-back

U M2 U' M2

hidden behind



red edge bottom-back;
orange edge top-left

U' M U2 M' U'

hidden behind



red edge bottom-back;
orange edge top-right

U M' U2 M' U M2

hidden behind



red edge bottom-back;
orange edge top-front

M U2 M U' M2 U'

hidden behind



red edge bottom-back;
orange edge top-back

U2 M U2 M U M2 U'

hidden behind



red edge bottom-back;
orange edge bottom-front

U' M2 U M2

hidden behind

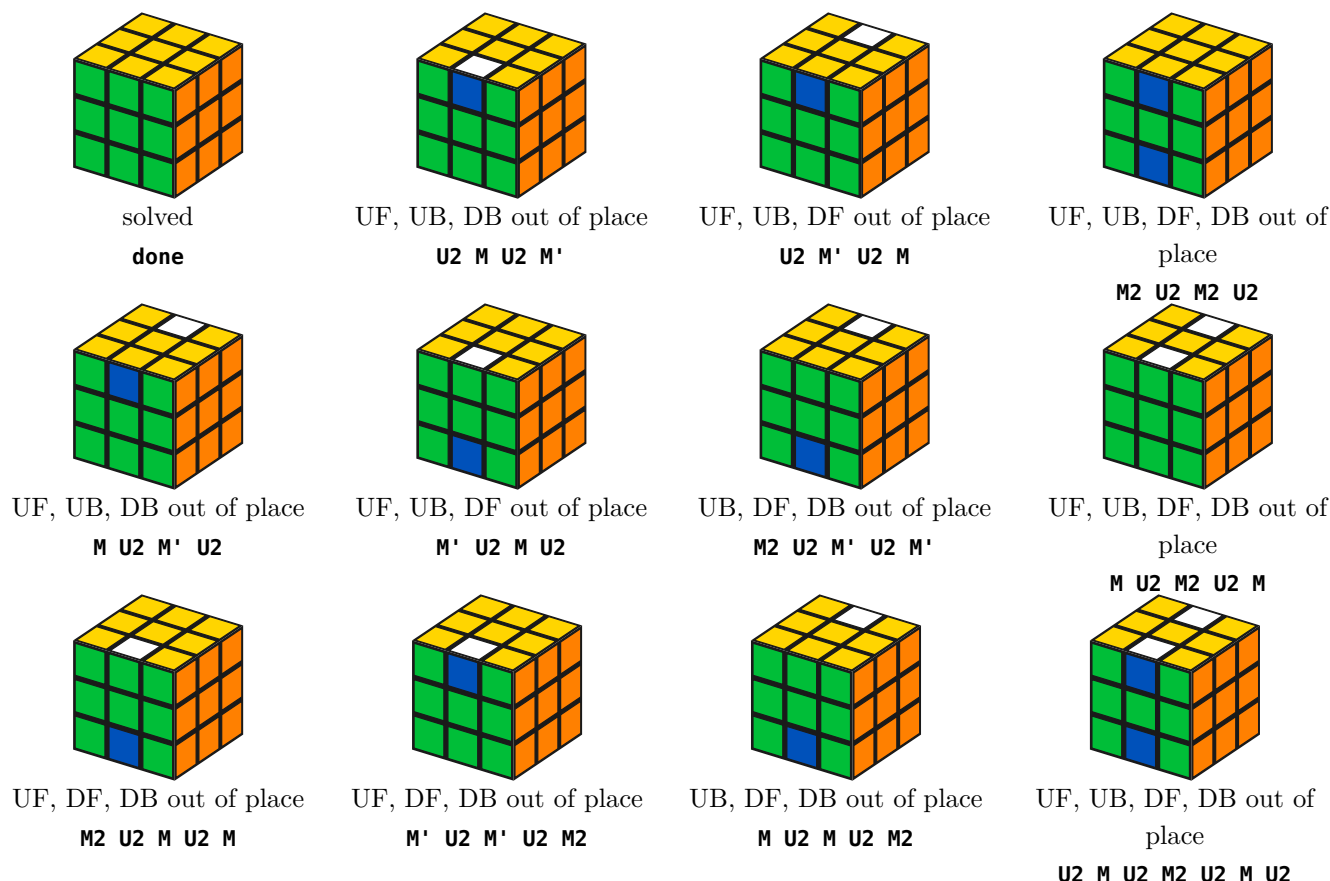
7.3 4c: the last four edges

7.3.1 what to do

only the four middle-slice edges (UF, UB, DF, DB) and the middle-slice centres remain. with the yellow centre on top, look at which edges are out of place: there are twelve possibilities, and each is a short M and U2 sequence. Roux solves the same thing with three moves he does from memory, “U2MU2, U2M2U2 and E2ME2”; the table gives you the exact one for what you see.

7.3.2 the twelve permutation cases

the cards show the true colours of every piece, so match the whole picture.



8 glossary

- **AUF**: “adjust the U face”: turn the top layer until the cube matches a diagram.
- **adjacent swap**: two neighbouring top corners need to trade places; one face shows headlights.
- **bad edge**: an edge whose yellow or white sticker faces a side (with yellow centre on top).
- **bar**: two yellow corner stickers on one side face; not headlights.
- **block**: pieces solved relative to one another, moving as a unit. 1x2x3: a centre, three edges and two corners.
- **centre**: the one-sticker piece in the middle of a face.
- **CMLL**: the algorithmic step that solves the four top corners, ignoring the middle slice.
- **corner**: a three-sticker piece.
- **diagonal swap**: two top corners opposite each other need to trade places; no headlights anywhere.
- **edge**: a two-sticker piece.
- **EO**: edge orientation; the state of every edge being good or bad; step 4a makes them all good.
- **F2B**: first two blocks, the state after steps 1 and 2.
- **first block, FB**: the 1x2x3 built on the bottom-left (red centre).
- **good edge**: an edge whose yellow or white sticker is on the top or bottom face.

- **headlights**: two corner stickers of the same non-yellow colour on one side face; those corners are correctly placed relative to each other.
- **LSE**: last six edges; step 4.
- **M slice**: the middle layer between left and right; “free” during the blocks and CMLL.
- **misoriented**: of a corner, its yellow sticker faces a side; of an edge, see bad edge.
- **oriented**: of a corner, yellow on top.
- **orientation / permutation**: which way a piece sits in its slot / which slot it is in.
- **pair** (1x1x2): a corner joined to one of its edges.
- **second block, SB**: the 1x2x3 built on the bottom-right (orange centre).
- **slot**: a position on the cube, named by the faces it touches (UF, DFR).
- **square** (1x2x2): a corner with both its edges, around a centre.
- **sune, anti-sune**: the CMLL patterns with exactly one oriented corner, named after the classic sequence $R\ U\ R'\ U\ R\ U^2\ R'$ and its mirror.

9 credits and references

the method, the tables of step 2, the orientation rows of step 3 and the edge-orientation definition of step 4 are Gilles Roux’s, from his site grrroux.free.fr (Introduction, Step 1 to Step 4, Miscellaneous), fetched 2026-10-18. his pictures are redrawn here rather than copied; his words are quoted where they are his.

- [Gilles Roux](#), “Introduction” and [Step 2](#), [Step 3](#), [Step 4](#): the source.
- [Kian Mansour’s roux guide](#): the one-sequence-per-case CMLL set used for the 42 cards, and the modern naming (O, H, Pi, U, T, L, S, AS).
- [speedsolving wiki: roux method](#): history and overview.
- [the CTAN rubik bundle](#): draws the flat cards in the PDF.
- [tools/](#) next to this page: the cube simulator ([gen_cmll.py](#)), the drawing code ([cube3d.py](#)) and the case generator ([gen_cases.py](#)) that produced every diagram and checked every sequence; [cases.json](#) has all of it as data.