VR TUTORIAL (WIP)

New version, by Lorenzo Mauro

Introduction

Currently the best method for FMC is DR. Assuming that the DR is on U/D, which is an assumption that will be valid throughout this document by default, then the current meta is to ignore the E-slice, until you've solved everything else, to get a Leave Slice (LS). Once an LS has been achieved, E-slices can be inserted in the post-DR section in order to solve it. This is often very efficient, and can often be done in +0 or +1. Before VR there wasn't really any good way to find the optimal slice solution, and the only thing you could do is brute force, with some tricks, such as rNISSing, the fact that E2s will never solve 3e, and some intuition-based strategies, which however aren't as systematic as VR, and aren't as good for cases where the optimal isn't +0.

VR (standing for Vierergruppe Reduction) is a way I invented to find the optimal slice consistently. It consists of two steps:

- Reduce to Vierergruppe, meaning insert E/E' moves so that the slice gets reduced to a skip, a 2e2e, a 2x2x or a 2e2e2x2x. In analogy with DR, we will call any such insertion a VR.
- Solve the slice by inserting E2 moves.

I have to address a misconception here though: the final process to find the slice might have more than two actual steps, as you really should go through multiple VR's some of the time. I'll describe the full algorithm below (spoiler: if you've heard of the Kociemba method for 3x3, this is the same idea for using it to find optimal solutions to a 3x3).

If you're interested here's the link of my original proposal of VR: https://docs.google.com/document/d/1SDdEm4dDDv7jJ3TRjj1QxI-6j7iG0BekJmNRaqaM3kE/edit?usp=sharing . This isn't a tutorial, and I'm looking to do a better job of explaining VR here. Furthermore I've since found a better way to do step 1, which will be described here.

How to use VR to find the optimal slice

- Find *all* VRs in +0. Even if you already have a VR skip, you must find all others.
- For each of the VRs you found above, insert E2 moves to solve the slice. As soon as you find a slice in +0, stop.
- If you found a +0 or a +1, stop.
- Find all VRs in +1.
- For each of these new VRs, insert E2 moves to solve the slice. As soon as you find a slice in +1,stop.
- If you found a +1 or +2, stop.
- Find all VRs in +2.
- …

In practice for +2 and worse, a direct 3e insertion might be the better way to solve the slice, especially if you have very few U moves to work with.

How to find VRs

The first thing that you need to do, is write the full post-DR on normal. Then take a cube, and go to the end of the post-DR. Now you'll need to assign a number to the edge permutation of the slice, which is our goal number:

- If you have a 2e2e, then the goal number is 0.
- If to solve the edges of the slice you need to do F2 R2, F2 L2, B2 R2 or B2 L2, then the goal number is 1.
- If to solve the edges of the slice you need to do R2 F2, R2 B2, L2 F2 or L2 B2, then the goal number is -1.

Note that the centers are allowed to be an E2 off at the end of your skeleton. This doesn't affect anything.

Now you need to isolate a section of your skeleton. This depends on whether you're looking for VR in +0, +1, or more. If you're looking for VR in +0, then consider the section between the first qt and the last qt. If you're looking for VR in +1, then consider the section between the first U/D move, and the last U/D move (both can be qt's or U2/D2 moves). Otherwise the entire skeleton is the section.

Now we will be assigning numbers to each move. Here’s a table explaining how that works.

Moves\Parity of Placement (explained below)	Even	Odd
F2/B2	+1	-1
R2/L2	-1	+1
U/D moves	0	0

Where parity of placement (POP) refers to the number of side turns that come after the considered move. An example is in order

Example scramble: R' U' F D F2 L2 U F2 R B' D' F L U' F2 U2 R2 L2 U F2 U2 R2 U' R' U' F.

Example skeleton: F2 U' R2 U' R2 F2 U' L2 U R2 U2 F2 B2.

Our goal number is -1, since you need to do L2 F2 to solve the slice.

Let’s start from the end of the skeleton:

- B2 has even POP, since it’s the very last move, so it’s +1.
- F2 has odd POP, so it’s -1.
- The U2 is 0, and can be ignored.
- R2 has even POP (note: U/D moves do NOT contribute to POP), so it’s -1.
- The U can be ignored (but not quite, we need to know where it is).
- L2 has odd POP, so it’s +1.
- …

But actually, you need NOT do this for all moves. You only need to do this for moves in your section.

Now, let’s say we’re looking for VRs in +0. We want to isolate segments that are between quarter turns, as those moves are the only ones where we can insert E/E’ while adding 0 moves. I like to use pipes (the “|” symbol) to separate sections, although with this new method you can just do everything in your head for VRs in +0. Nevertheless I will write everything, so that you can understand what your thought process should be.

Let’s replace all qt’s in the skeleton with pipes:

F2 | R2 | R2 F2 | L2 | R2 U2 F2 B2.

Note that our section of interest is the one between the first pipe and the last pipe. We are completely free to ignore everything before the first pipe, but as far as moves after the last pipe are concerned, we need to keep track of how many side moves there are (otherwise we can't determine the POPs). So all we need to consider is:

| R2 | R2 F2 | L2 | [odd number of side turns].

Now convert each of these moves into numbers using are special rules above (start from the right, and go left):

| -1 | +1 +1 | +1 |.

Within each segment, take the sum modulo 3: this means that +1+1=2=-1 (mod 3).

| -1 | -1 | +1 |.

Here's how we're going to find our VRs in +0: find all ways to pick segments so that their sum (mod 3) is equal to our goal number. In our case our goal number is -1, so we can pick either of the -1's, or all three segments.

Now, given a choice of which segments to pick, how do we extract a VR? Remember that the pipes stand for qt's. If a pipe is between a picked segment and an unpicked segment (for our purposes, everything outside of all the pipes is unpicked), then you need to insert an E/E' move that cancels 2 there. For example, suppose we pick the first -1. This corresponds to widening the first and second qt, meaning we get this VR:

F2 Uw' R2 Uw' R2 F2 U' L2 U R2 U2 F2 B2.

Note that centers are not solved. This is ok. If centers are unsolved, they will always be an E2 away, and this works for us.

Had we picked the second -1, we would have gotten this other VR:

F2 U' R2 Uw' R2 F2 Uw' L2 U R2 U2 F2 B2.

And finally, picking all three segments, we get:

F2 Uw' R2 U' R2 F2 U' L2 Uw R2 U2 F2 B2.

These are all three VRs in +0. I like to denote these VRs by writing a square bracket under each picked segment, so that I don't have to write the whole skeleton again.

Now, for this specific example we most likely will not need to find VRs in +1, but let me show you what changes. Since we can still insert an E on the U2 and only add one move, we need to expand our section to include that U2. In the following I will denote U2/D2 moves with dollar signs. Remember that picking a segment bounded by a dollar sign adds one move, and picking a segment bounded by two dollar signs adds two moves. In our case we only have one U2 move to work with. So we have:

| R2 | R2 F2 | L2 | R2 $ [even number of side turns].

Convert moves to numbers, and add everything that's not separated by pipes or dollar signs:

| -1 | -1 | +1 | -1 $

Note that because there aren't U2 moves between qt's, the numbers between pipes are unaffected. If we had had something like U R2 U2 F2 U [even], then we'd have | +1 $ +1 |.

Now we can pick the last -1 alone, or together with the +1 and any other -1. We obtain the following VRs in +1:

F2 U' R2 U' R2 F2 U' L2 Uw R2 Uw U F2 B2.
F2 U' R2 Uw' R2 F2 U' L2 U R2 Uw U F2 B2.
F2 Uw' R2 Uw' R2 F2 Uw' L2 U R2 Uw U F2 B2.

We can even look for VRs in +2. I will not write them all, but now we are not allowed to ignore any move. So we have:

F2 | R2 | R2 F2 | L2 | R2 $ F2 B2.

Convert to numbers:

-1 | -1 | +1 +1 | +1 | -1 $ -1 +1.

Now we just need to pick some segment, but this need not be bounded by pipes and dollar signs on both sides. For example, we can pick the leftmost -1. This gives the following VR in +1:

Uw U' F2 Uw' R2 U' R2 F2 U' L2 U R2 U2 F2 B2.

As another example, we can pick the segment starting from the third +1, and ending at the last -1:

F2 U' R2 U' R2 F2 Uw' L2 U R2 U2 F2 Uw U' B2.

However, note that F2 and B2 commute, so we get another VR in +2 for free:

F2 U' R2 U' R2 F2 Uw' L2 U R2 U2 B2 Uw U' F2.

As you can see, there are tons of VRs in +2 on this scramble. However, if you need to find VR in +2, then the situation is already desperate, which on 4qt is unlikely, especially with no segments between qt's evaluating to 0.

What to do if the DR is not on U/D?

Unlike my previous approach for VR, here it's not quite as obvious what to do. If DR is on F/B, then let the goal number be +1 if you need to do U2 R2, U2 L2, D2 R2 or D2 L2 to solve the slice, and -1 for the inverses. If DR is on R/L, then +1 corresponds to U2 F2, U2 B2, D2 F2 and D2 B2.

The rule for assigning numbers to moves becomes:

- For DR in F/B and even POP: U2, D2=+1, R2, L2=-1, odd POP is the opposite.
- For DR in R/L and even POP: U2, D2=+1, F2, B2=-1, odd POP is the opposite.

The rest is very easy to translate.

If you find this hard to remember, just remember U>F>R. So if your DR is on F/B, then for the goal number to be +1, you need the slice to be solved with U2 R2. Also U2 with even POP will be +1, and R2 with even POP will be -1.

But POP is confusing!

What I described above is something that is close to the inner workings of the method. This makes it not very easy to do in practice, the way I described it, but other people have found ways to not have to count POP, and write the translated skeleton faster. I will describe the two that I've seen, but it's definitely possible that some people have come up or will come up with other ways to write the translated skeleton in practice.

The “R F R F R F…” way

This was found by Rodney immediately after I shared about this mod 3 way of doing step 1. There are two variations of this too. One of them is explained here (doc by discord user tseitsei89): https://docs.google.com/document/d/1zMUmAQR0pMxKWbdTmOIYrbU1bvDbPo7p5H9EabJfkrI/edit?tab=t.0 .

The idea is that you say “R F R F R F…” while reading the non-U/D moves. If the move you say is on the same axis as the one you read, you write a +, otherwise you write a -. Moreover you extend the skeleton to include some moves you need to do to solve the slice (without regard to the rest), and continue with that. This can be hard to understand without an example, so let’s go back to our own example skeleton.

F2 U' R2 U' R2 F2 U' L2 U R2 U2 F2 B2 // L2 F2,

Where I’ve written the solution to the slice after the skeleton.

Now say “R F R F…” while reading all the non-U/D moves.

- The first move is an F2, which is not on the same axis as R, so this is a -;
- The second move is an R2, which is not on the same axis as F, so this is a -;
- The third move is an R2, which is on the same axis as R, so this is a +,
- …

So we get:

- | - | + + | + | - $ - + // + +

Now, the goal number is the sum of all the numbers after the // (note that I’ve abbreviated +1 into + and -1 into -), so it’s -1. Note that, if you do it like this, then half of the time you’ll get all the signs opposite of what I’ve described above (so both the signs of the skeleton and the goal number would be flipped). This is completely fine.

Earlier, I mentioned another variation of this. This just entails reading everything backwards, and saying “R F R F…”, while reading right to left (including the moves to the right of the # sign), rather, rather than left to right. This gives the same result as the rules I gave above. Now, the only reason to prefer this over the “forwards approach” is that some experimental techniques to predict the 2e2e and doing step 2 without having to perform VRs on the cube become easier. But it’s very unlikely that such techniques I’m alluding to are worth it.

The “switch sign or not” way

This was found by discord user comp.lex, also very shortly after I published this. The idea is very simple: POP changes at literally every side move, so you don’t have to recompute it every time. So you determine the goal number just as I stated above, then you go to the right of the skeleton, take the rightmost side turn, and translate it with my rules, remembering that at this point POP is trivially even. Now you read right to left all the side turns, and follow these rules to translate them into numbers:

- If the current side turn is on the same axis as the previous side turn, you switch the sign;
- Otherwise you keep the same sign.

So if you’ve just written a + under an R2, and the side move just on the left of it is an F2, you would write a + under the F2 too, but if that was an L2, you’d write a - under that.

How to do step 2

Pick the VR you want to finish off of, and perform it on the DR state. You should have a skip, a 2e2e, a 2x2x or a 2e2e2x2x. Anything else means that you’ve made a mistake in step 1.

Now you need to determine two things: whether centers are solved, and what’s the absolute edge permutation. The latter has 4 possible answers:

- You can have a FR-FL, BR-BL swap. We call this f, as there’s a 2-swap of green edges, which belong to the F face.
- You can have a FR-BR, FL-BL swap. We call this r, as there’s a 2-swap of red edges, which belong to the R face.
- You can have a FR-BL, FL-BR swap. We call this g, as there are diag swaps, and g is the last letter of “diag”.
- You can have solved edges. We call this 0.

IMPORTANT NOTE: if centers are unsolved, then the edge permutation is to be seen NOT relative to centers, BUT with respect to the rest of the cube. So if the slice needs R2 E2 R2 to be solved, then this is f, NOT r.

Now we look at the following table.

EP \ Centers	Solved	Unsolved
0	-	f+r+g

f	r+g	f
r	f+g	r
g	f+r	g

This table should be very easy to memorize: notice that, discarding the 0 row, the rightmost column is the copy of the leftmost column, and the middle column just includes the other two 2e2e's. NOTE: the order of the addends does NOT matter, so that's not an excuse for not being able to remember the table!

Ok, but what does this table mean? To understand this, we first need to understand that an E2 move does a 2e2e (+2x2x) permutation. If you insert an E2 somewhere inside the VR, you will perform a 2e2e (and swap centers). This 2e2e is determined by what edges are opposite of each other at the site where you want to insert that E2. Looking at the colors of the involved pieces, you can see that this swap will be either f, r or g. The table tells you where to insert E2 moves.

Let's do an example. Consider our three VRs in +0 we found earlier:

1) F2 Uw' R2 Uw' R2 F2 U' L2 U R2 U2 F2 B2.
2) F2 U' R2 Uw' R2 F2 Uw' L2 U R2 U2 F2 B2.
3) F2 Uw' R2 U' R2 F2 U' L2 Uw R2 U2 F2 B2.

As a reminder, the scramble was R' U' F D F2 L2 U F2 R B' D' F L U' F2 U2 R2 L2 U F2 U2 R2 U' R' U' F.

Let's do the scramble and then the first VR. We get solved edges, and unsolved centers. As a rule of thumb, this is the worst case, as it requires 3 E2 insertions. Here we only have one U2 to work with, so the best that we can do with this VR is +2. In practice, you should skip this, and only come back to it if you found nothing better than +2 on the other VRs, which is unlikely. However, let's see how you would finish this anyway. We need an f, an r and a g.

Let's backtrack to the U2 move. You see that the FR and BL edges already belong to opposite spots on the cube, so inserting an E2 results in g. Now we need an f and an r.

Now let's backtrack to the U move. You see that FR and BL are both blue edges, so we have f here. We only need an r now.

Let's backtrack to the U' move. We have g again, no good, we don't want a g.

Let's backtrack to the second Uw': we get f again.

Let’s backtrack to the first Uw’: we get g again. Thus from here we can only hope for +3. As an exercise, try to find one.

The first VR didn’t work out so well. Let’s try the second one. We get dots again. However this time there is a +2: insert an E2 on the U2 (g), on the U (f), and on the first Uw’ (r).
Let’s try the third VR now. We have solved centers, and the edge permutation is f. Thus we need a g and an r. We get a +1 by inserting an E2 on the U2 (g), and on the second U’ (r).

We have the final slicing:

F2 Uw' R2 U' R2 F2 U Uw2 L2 Uw R2 Uw2 F2 B2.

This is +1, and thus we don’t even need to look for VRs in +1. This is already optimal, as we’ve exhausted all VRs in +0.

Just for fun, let’s look at the first VR in +1 we found:

F2 U' R2 U' R2 F2 U' L2 Uw R2 Uw U F2 B2.

Centers are unsolved, and the edge permutation is r (NOT f). We need an r. We get r on the first Uw, thus finding another +2.