LMCF 3x3 Rubik's Cube Method (Revision 4.5)

(Low Movecount Corners First, by Eric Fattah [freediver991@gmail.com], March 18/2017)

In modern speedcubing, CFOP and Roux are accepted as the fastest methods and little attention is paid to corners first variants such as Waterman. This has been a grave mistake. For example, at the 2003 Rubik's cube world championships, all the cubers were using CFOP and yet no one managed to get under 20 seconds; the world record average at the time was 17.70 seconds by Marc Waterman from 1987 using his corners first method. So even after 21 years of developing CFOP (1982-2003), the best CFOP solvers were still not able to beat the corners first record time. The LMCF (low movecount corners first) method I have developed combines modern developments in 2x2 solving, bringing those developments into the 3x3 and mixing them with Waterman's techniques. The result is a method with an average movecount of 41-45 (based on how many algorithms you learn), and best of all it has the highest statistical chance of sub-30 move solves compared to any method. Indeed 27-29 move solutions are not that uncommon and happen in around 1 in 500 solves. Sub 36 move solves can happen as many as 1 in 6. Although I am not a fast turner (I average 3.80 TPS), I routinely get 8 to 10 second solves using LMCF simply because of the low movecount. A 27-29 move LMCF solve could be executed in less than 3 seconds by an expert cuber.

PLEASE WATCH MY LMCF TUTORIAL VIDEO ON MY YOUTUBE CHANNEL:

https://www.youtube.com/channel/UCc438H4nNG991tJBGo9OC9A

The basic steps in the LMCF method are:

1. SOLVE THE CORNERS LIKE A 2x2

Solve the corners using full EG as if you are solving a 2x2. Like a master 2x2 solver, you should be able to see the entire corner solve during the 15 second inspection and execute it blindfolded. If you use EG1, EG2 and CLL there are 127 algorithms. If you learn TCLL as well there are 167 algorithms for this step. If you are a beginner you can use Ortega to solve this step which is only about 12 algorithms but is slower.

2. TRANSITION PHASE

The transition phase attempts to transition from solving the corners into solving the edges. This step is very short, doesn't really have any algorithms (see Appendix), but is actually one of the biggest secrets and most confusing parts of the method. During this phase you solve 1-2 edges of the top face while looking at the E-slice and the rest of the cube to find your first E2L pair before you do a z/z' rotation and start the E2L phase. This phase is explained in high detail in my tutorial video. This phase is CRITICAL and is the key to a smooth solve without pauses.

3. SOLVE 6 L/R EDGES, SOLVING THEM IN PAIRS OR TRIPLETS (E2L PHASE)

The goal of this step is to solve a TOTAL of SIX edges of the L/R faces (you might have already solved one of them in the previous step). At this point we adopt Waterman's terminology; the M-Slice becomes the 'ring', the edges of the middle slice are called 'midges', the edges of the left face are called 'ledges', the edges of the right face are called 'redges.' The entire step is called E2L (edges of the 2 layers) and these pairs are called E2L pairs. This step has a lot of similarities to the CFOP F2L phase, with some big differences. In about a quarter of cases it is possible to solve THREE edges at the same time instead of just two, this is called an E2L triplet. In this phase the goal is to complete 6 edges of the L/R faces. In some cases you already solved 1 edge in step 2. So there are several cases:

→ solve (1) edge in step 2, then solve one E2L triplet (3) and one E2L pair (2) = total SIX edges
→ solve (1) edge in step 2, then solve two E2L pairs (4) and one single edge on its own = total SIX edges
→ solve (0) edges in step 2, then solve three E2L pairs (6) = total SIX edges
→ solve (0) edges in step 2, then solve two E2L triplets (6) = total SIX edges

At the end of this step, you have solved 6 edges of the L/R faces and you will have just 6 edges left to solve the rest of the cube.

4. LAST SIX EDGES (L6E)

At this start of this step 2 edges of the L/R faces are still unsolved. There are three cases:

→ one edge of the L face is unsolved (UL), and one edge of the R face is unsolved (UR) (similar to Roux LSE except L/R faces may be misaligned)
OR
→ two edges of the L face are unsolved, and all the edges of the R face are already solved
OR
→ two edges of the R face are unsolved, and all the edges of the L face are already solved

The goal here is to solve the last two edges of L/R, while simultaneously orienting the midges (edges of the M-slice), then permute the midges and finish the solve. In some cases it is possible to use the Roux LSE method to finish this step. However the LMCF method has special algorithms for this step, of which about half are 'new' and the other half are borrowed from Marc Waterman's method. This step can be finished in as little as 8 algorithms or as many as 453 algorithms if you learn all the cases with all the reflections. While you execute the algorithm to solve the last L/R edges and permute the midges, you must 'look-ahead' to the L/R slice adjustments, as usually you need to adjust L or R (L', R', R, R2) to get the L and R slices to match. The final part of this step is to permute the midges (edges of the M-slice), which is trivial and has only 3 cases.

IMPORTANT: HYBRID METHODS

As speedcubing evolves, more and cubers are becoming METHOD NEUTRAL. This means you choose your method based on the scramble that you are presented with. Some scrambles heavily favour one method over another. For example if you see a ready-made cross, it would be a good idea to use CFOP. If you see a 3x2x1 block, you could start with Roux. If you see a face of corners, you could start by solving the corners and use LMCF. My personal suggestion for method neutral solving is this:

→ Do you see a face of 3 or 4 corners, even if incorrectly permuted? If so, full EG with 1-look is easy, so use LMCF
→ Do you see the possibility of making an entire slice minus one edge? If so, use Waterman (which is actually a subset of LMCF)
→ Do you see the possibility of making a 3x2x1 block easily? If so, use Roux
→ Do you see a 0-1 move cross? If so, use CFOP

Algorithm Counts

LMCF has drastically varying algorithm counts, depending on what you want to learn. The more algorithms you learn, the lower the movecount of the method. Here are the basic breakdowns:

	LMCF Basic	LMCF Intermediate	LMCF Full (Advanced)	LMCF Roux Simplified
Corners Algorithms	12 (Ortega 2x2)	CLL+EG1 (87)	CLL+EG1+EG2+TCLL (167)	CMLL (42)
E2L Algorithms	3	84 (E2L only)	156 (E2L + pure triplets)	42 (E2L front reflections only)
L6E Algorithms	11 (8 DFL + 3 midge permutations)	30 (DFL, BFR, iDFR, +special cases)	78 basic LMCF cases, plus 186 R-face Waterman plus 186 L-face Waterman, plus 3 midge permutations = 453	15 (Roux LSE)
Average Move-count per Solve	59	46-49	41-45	50 (estimate)
Total Algorithms	**26**	**201**	**776**	**99**

At the time of writing this document (Feb 25/2017) I am using the LMCF intermediate set (201 algorithms), and I am in the process of learning the full set. It is worth noting that the LMCF beginner's method (26 algorithms) gives a similar movecount as CFOP with full OLL/PLL (78 algorithms). From the standpoint of a beginner, LMCF beginner's method can easily get you to 16-17 seconds with good enough TPS and lookahead, making it possibly the fastest method for such a low count of algorithms (26). It is also very intuitive for a beginner, and allows easy transition from the basic algorithm set into larger algorithm sets with corresponding increases in speed.

Statistics

LMCF intermediate set (201 algorithms) has a similar move count as advanced Roux (46-49), with two major differences. In LMCF you identify the 'big' algorithm (CLL/EG1) during the inspection time; in Roux you must identify the CMLL algorithm during the solve, so in Roux, the recognition time counts as part of the solve. In LMCF the recognition happens during the inspection. Both LMCF and Roux allow for CLL/CMLL skips (1 in 162) giving very low move count solves. Unlike Roux, LMCF has a huge probability of pre-solved edges after the corner solve, with half of solves having at least 1 solved edge piece. The full LMCF set (776 algorithms) has a similar algorithm count as CFOP with full ZBLL. The move count average for these methods is somewhat similar, but again LMCF has a higher probability of low movecount singles. The high chance of low movecount singles in LMCF comes from several factors:

1. Chance of CLL skip (1 in 162) or EG2 'skip'
2. High (50%) chance of at least 1 L/R edge pre-solved after doing the corners, in many cases you have 2 or even 3 pre-solved edges, making the E2L phase very fast
3. High chance of midge permutation skip
4. High chance (>50%) that an E2L pair can solve three edges pieces instead of two, which happens almost every solve if you learn the full E2L triplet set. This does not add any moves to the solve but diminishes the number of E2L pairs you need to solve. In lucky solves you can finish the entire six E2L edges with a single E2L triplet (with 3 pre-solved edges). This is equivalent to a CFOP solve where you only need to solve one F2L pair.

Move Counts per Step

Full EG: Average 13 Moves
Full EG + TCLL: Average 12 Moves
Six edges E2L with 84 algorithm E2L set: 17.85 moves
Six edges E2L with 156 algorithm E2L set: 16.00 moves
L6E with intermediate algorithm set (30 algs): 16 moves
L6E with full advanced set (453 algs): 13 moves

Step-by-Step Algorithms & Instructions

1. CORNERS

The first step involves solving the corners as if solving a 2x2 cube. For this step you can use the Ortega method which is very simple and basic, but not very fast and it does not have a low movecount. Having said that, if you can solve a 2x2 in less than 4 seconds using the Ortega method, it is possible to still solve the 3x3 very fast using LMCF. However, the recommended method is to use CLL, EG1 and EG2. Advanced cubers can also use TCLL. I recommend watching 2x2 tutorial videos by Chris Olson (www.cyotheking.com and associated YouTube channel). The main differences when using LMCF versus classic 2x2 are as follows:

1. The 2x2 does not have center pieces. In LMCF you ideally will solve one edge piece at the same time as you solve the centers. If using an EG1 algorithm, the center of the 'bar' is usually untouched by the algorithm so you can (if possible) place a valid edge piece into the bar prior to the algorithm. Sometimes it is better to solve the 2 centers (L/R) before executing the corner algorithm; other times you solve the 2 centers AFTER the corners algorithm, solving one edge piece at the same time.

2. In LMCF you do not need to AUF at the end of the corners algorithm if you choose the U and D faces to be your L/R faces. For example, I am able to solve the corners color-neutral, but in the E2L phase I am not color neutral and I solve green and blue on the left and right (either green/blue or blue/green, I don't care which). If I create a blue or green face, then my EG algorithm will solve the blue/green corners. After solving the corners I solve the blue/green centers on U/D while solving 1 edge piece. I do not need to AUF because I do not care if the blue/green faces are aligned. I then rotate the cube so blue/green are on the left and right. In other cases if I solve the yellow/white or orange/red corners, then I might need to AUF. This is more clear in the example solves.

The general summary here is in LMCF, normally you do NOT need to AUF after the EG algorithm, but you DO need to solve 2 centers (of the L/R faces).

The 3x3 moves a bit different than the 2x2. I have selected an ideal set of CLL and EG1 algorithms optimized for 3x3, and in some cases made modifications to existing algorithms for better turning style. The graphical sheet on the next page shows my algorithm selections for CLL and EG1 for 3x3 LMCF. In classic 2x2 style, all of the EG1 algorithms require the 'bar' on the D face to be at the back.

Now, as fast singles are a critical part of LMCF and a main reason why people choose this method, there are some words of warning on how to get fast singles in terms of the corners step.

If you choose to make the fastest 'face' possible of four corners (with random permutations), there is a 66% chance (2/3) that it will be an EG1 face, and a 16.6% (1/6) chance that it will be a CLL face, and a 16.6% chance (1/6) that it will be an EG2 face. However, if you choose a CLL face, then there is a 1 in 162 chance of a CLL skip which is a major savings, not only in moves, but quite often you can see the CLL skip in the inspection and actually lookahead to EDGE SOLVING while still in the inspection. If you can lookahead to edge solving in the inspection that you are SURE to have an INCREDIBLY LOW MOVECOUNT solve with extreme speed.

If you make an EG2 face then you have a 1 in 162 chance of an R2-F2-R2 permute on the top layer which is essentially an EG2 skip, almost as good as a CLL skip.

If you make an EG1 face then there is essentially no skip possible. There is no case which solves in less than 7 moves. It is very tempting to make EG1 faces all the time since they are usually the easiest face to make, and indeed this is what I do. However if you are aiming for fast singles, making CLL and EG2 faces will give a greater statistical chance of a 'skip' and extremely fast solve. In some cases the CLL skip is so obvious you can see it right away in the inspection. This is the best case of all.

SUNE	ANTI-SUNE	L CASES	T CASES	U CASES	PI CASES	H CASES
CLL: R U R' U R U2 R' EG1: z' R' U' R U2 B2 l U'	R' U' R U' R' U2 R U' B U' r2 F2 U' F	F R' F' R U R U' R' U R' F R U' R' F R2 U R' F'	U R' U R U2 R2 F R F' R R U R' F R U R' F U ' R U' R'	U' R2 F2 R U R' F U' R U R2 x B2 U' R' U R U' F R U R U'	F R U R' U' R U R' U' F' U' F U' R' F R U' F2 R U R'	R' U2 R y R' U R' U' R U' R R U R' F' R U R' U' R U R'
U' R' F R2 F' R U2 R' U' R2 R U R' F2 U F R U R'	R U2 R' F R' F' R U' R U' R' R U' F2 R U2 R U' F	F R U' R' U' R U R' F' U R U' R2 U' F U' R2 U2 R	U F U' R U2 R' U' F2 R U R' L' U L2 F' L' U' L' U2 L	F U' R U R' U' F' U2 F U' F' U2 R' F R F' R' F R2 U' R'	U2 R U R' U R D' R U' R' F' U' R' F R2 U' R2 F R	U' R U R' U R U R' F R' F' R U' R' U' R' F2 x B U' R U'
R U' R' F R' F' R F' U R U' R' U F R U R'	x' R' F R U' R U R' R U' R' F' U' R U R' U' F	U2 R2 U R' U2 R U2 R' U R2 R' F R2 U R' F' R U2 R'	U2 R' F' R U R U' R' F U' R U' R' F R U' R' F R U R' F'	U' R2 U R' U' R2 D' R' F2 R U' R' F R2 U' R' U y' R U R'	F U' R' F2 R U' F2 R U2 R' F R U' R' F R U2 R' U F'	R2 U2 R U2 R2 R' F R2 U' R' F R U R' F'
F R' F' R U2 R U2 R' U' R U' R' F2 U2 F U' F2	x' U' R U R' F2 R' F2 R R U2 R' y' R2 U R' U2 R'	R U2 R2 F R F' R U2 R' R' U R2 U' R2 U' F R2 U' R'	R U R' U' R' F R F' U2 R2 U R U' R2 F R U2 R' F	U' R U2 R' U l U2 R U' R' U2 R x B U B' R' U2 R U' B2 U'	R U2 R' U' R U R' U2 R' F R F' U2 F' L' U L F' L' U2 L U' F	F (R U R' U')x3 F' R2 U R U' R F2 U' R U R
R' F2 R U2 R U' R' F R' F R2 U' R' U R U' R' F	l U2 R' F2 R' F R U' x' R' F D R D F2 l'F U' F2	U R' F2 R U y' R2 U R U' R2 R U' R' y' R' U2 R' U R2	U' R' U R' U2 R U2 R' U R2 U' R' R2 B2 U' R' U' R U' R' U R'	F R U' R' U' F2 U' R U R' D R U F' U2 R U2 R' U2 F	R U' R' F R' F R U R' F R U' F R' F U' F2 R U R	
U' R' U' R D' R' U R' U' R U' R U R U' R' U R U' R' U F R U' R'	U R U R' D R U' R U R' U R' U' R U R' F' U' R U R' U' R U R'	R U2 R' U' y' R2 U' R' U R2 U L' U L y' R U2 R U' R2	U' F R U R' U' R U' R' U' R U R' F' L' U' L2 x U L' U' L U L'	F R U R' U' F' R U R' U R U R' - R' F R2 U' R	U' R U' R U' R' U R' F R2 F' U' F2 R U R' U2 F2 R U' R'	

CLL and EG1 algorithms Optimized for 3x3 LMCF method

by Eric Fattah (Feb 2017)

(First algorithm is CLL, second algorithm is EG1)

Permutation Algs:

R2 F2 R' U' R F2 R' U R'
R2 F2 R D R' F2 R D' R

R2 U' B2 U2 R2 U' R2

R2 F2 R2

R U R F2 R' U R'
R' D R' F2 R D' R

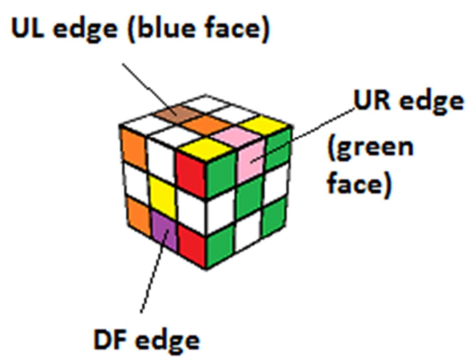
R U' R' U' F2 U' R U R' U F2

2. TRANSITION

The transition step is one of the most challenging steps of the LMCF method. In this step you are trying to make a smooth transition from solving the corners, to solving the edges. If you did not already solve the L/R centers before your corners algorithm, you will be solving the L/R centers during this stage with (hopefully) one edge piece as well. Even if the centers are already solved, you can usually solve one edge piece with M U M', or M D' M' or similar. See the example solves for a better understanding of the transition phase. When solving the center and/or one edge piece and possibly rotating the cube during this transition phase, you are looking ahead for your first E2L pair which is really the most important part of this transition phase, to ensure smooth TPS with no pauses between solving the corners and rolling directly into your first E2L edge pair.

3. E2L PHASE

During this phase we are trying to solve a total of 6 edges of the left and right faces. In an ideal solve you have solved one edge piece during the transition phase and you will therefore need to solve 5 edge pieces during the E2L phase. Again, in an ideal solve, these five edge pieces will be solved in two sets, one E2L pair and one E2L triplet. For the purpose of clarity I will assume that you have the blue face on the left, and the green face on the right, meaning we are solving blue and green edges in this phase:



As shown in the above diagram, the PRIMARY method of solving edges in this phase is to attempt simultaneously solve the DF edge, the UR edge and the UL edge, ALL AT THE SAME TIME. In the above example, the DF edge could be the green edge that fits into UR; or it could be the blue edge that fits into UL. If the UR slot contains ANY blue or green edge, then it can be solved as well; if the UL slot contains ANY blue or green edge, then it also can be solved at the same time in a single short algorithm. The general sequence to set up this E2L pair/triplet solve is as follows:

1. Search the ring (M-Slice) for any blue or green edge and place it at the DF position by doing M' or M.

2. Examine this edge at the DF position. If it is a green-yellow edge (for example) then adjust the green face [R] so that the green-yellow slot is at UR.

2. Now examine the UR slot. Does the UR slot contain a blue edge? If so, then adjust the L face so that UR fits into UL. If instead UR contains a green edge, then ignore the L face and you will be solving an E2L pair both on the L face.

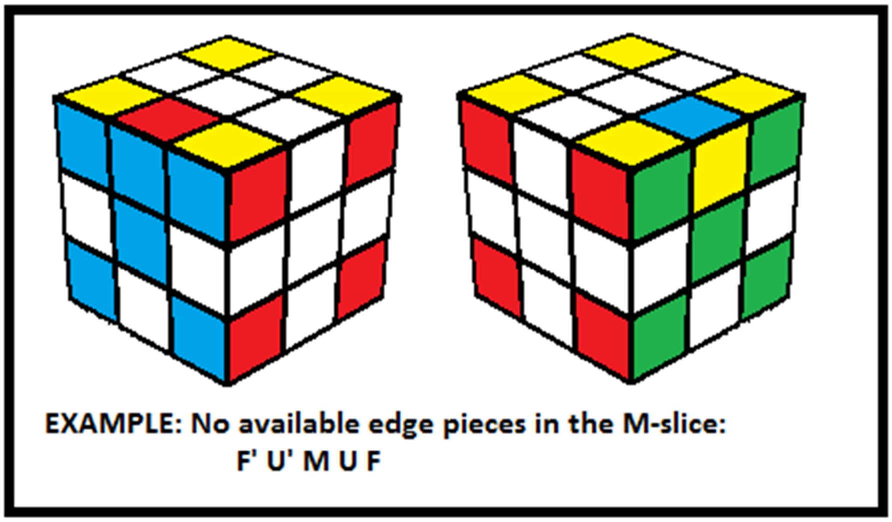
3. If the UR slot contained a BLUE edge, then look at the UL slot. Does the UL slot contain EITHER a blue or green edge? If so then you can solve this as well, possibly making the solve a triplet.

This will become more clear in the algorithm lists and the examples. The general idea is to find an edge in the ring (M-slice), place it at DF, then adjust the L and R faces to set up your E2L pair or triplet, where we are solving DF, UL and UR at the same time. Note that most of the algorithms are assuming that the DF edge has the blue/green facelet on the D face. If the blue or green facelet is on the F face, then do an M move and place the DF edge into BD. In this case you are solving the BD edge, the UL and UR edges. This uses the same algorithms but essentially acts as a front/back reflection. If you turn the cube (y2) then you are still using the same algorithm sets. Of course for full mastery it is strongly recommended to learn the reflections of the algorithms so you never need to do y2 rotations.

SPECIAL CASES

In extremely rare cases, after you solve the corners there will not be ANY blue or green edges in the M-slice. In this case the normal setup for an E2L pair or triplet fails. This case can only happen if EVERY SINGLE edge on the L/R faces is already filled with blue/green edges. In that case it is very easy to solve an E2L pair. You ignore the M-slice and adjust L/R so that you are solving the UL and UR edges at the same time, without solving any midges (M-slice edges). The same E2L triplet algorithms work (that normally solve DF, UL, UR), except since DF is 'empty', you only solve UL and UR.

This is just one of many examples where you have no available edges to solve in the M-Slice but you can still use existing E2L algorithms to solve both UL and UR:



E2L Algorithm List

The E2L pairs set contains 21 algorithms (84 with reflections). The E2L triplets set contains an additional 18 unique algorithms (72 with reflections). Therefore the entire E2L set consists of 156 algorithms.

The following E2L pairs list contains the 21 crucial E2L algorithms that can be used to solve pairs (some of which also solve triplets). This set is the DF into UR set. There are four reflections:
DF->UR = 21 algorithms
DF->UL (reflection L/R) = 21 algorithms
BD->UR (reflection F/B) = 21 algorithms
BD->UL (reflection both F/B and L/R) = 21 algorithms

Therefore this set has 21 primary algorithms and a total of 84 algorithms if you include the front back and left-right reflections. This is just the algorithm list, for a graphical depiction of the situation that each algorithm solves, look at the graphical images that follow.

E2L Pairs Set (21 algorithms)

Solve DF->UR while simultaneously solving UL

UL is oriented

1. UL->DL: U M' U' l' L' U M' U'
2. UL->FL: U l' U M U' L U'
3. UL->BL: B U M' U' B'
4. UL->DR: U M' U' R2 [Link to next pair]
5. UL->FR: U r U' M U R' U'
6. UL->BR: B' U M' U' B

UL is disoriented

7. UL->UL: U' M U2 M2 U'
8. UL->FL: M F' U M2 U' F
9. UL->BL: B' U M' U' M B
10. UL->DL: M' D' M U M' [U'+D]
11. UL->FR: M F U M2 U' F'
12. UL->BR: B U M' U' B'
13. UL->DR: M' D M U M' [U'+D']

Solve DF->UR while simultaneously solving UR

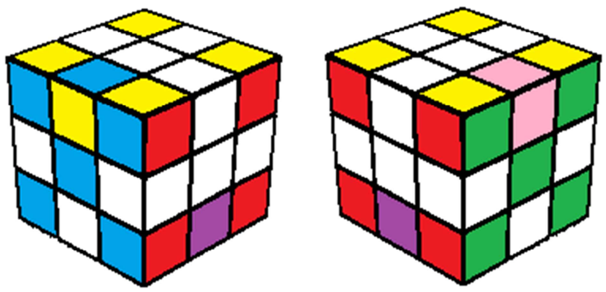
UR is oriented

14. UR->UL: M U M U2 M' U
15. UR->DR: U M' U R2 U' M' U
16. UR->FR: U M' U R U' M' U
17. UR->BR: U M' U R' U' M' U

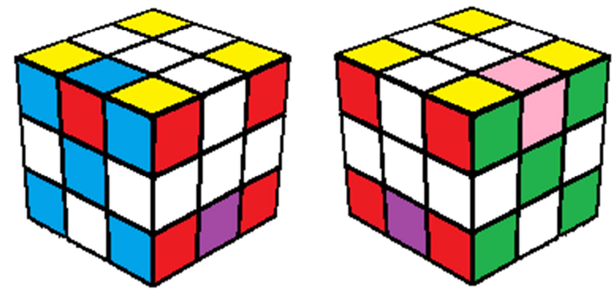
UR is disoriented

18. UR->UL: U M' U'
19. UR->DR: U2 R2 U' M' U R2 U2 (or: r2 y M' U M U')
20. UR->FR: U2 R U' M' U R' U2
21. UR->BR: U2 R' U' M' U R U2

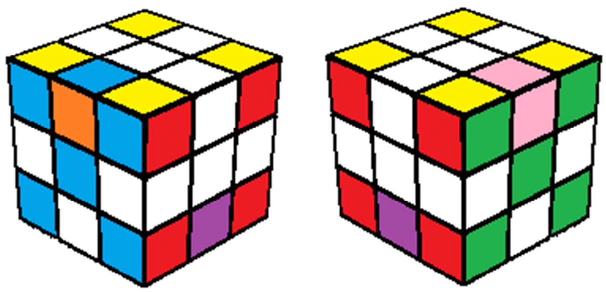
LMCF E2L algorithms, solving DF->UR, and solving UL at the same time



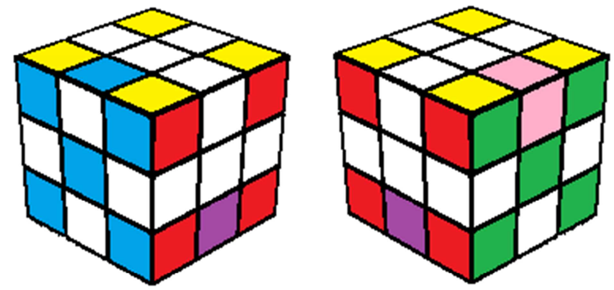
U' M U2 M2 U'



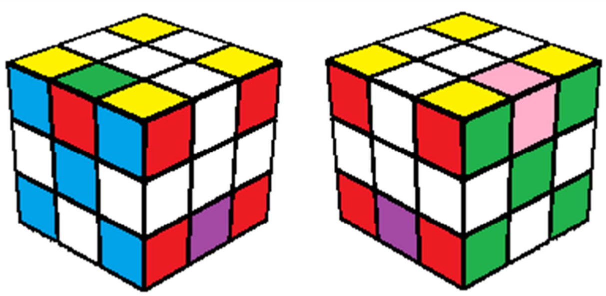
M F' U M2 U' F
Becomes a triplet if UR fits into UL



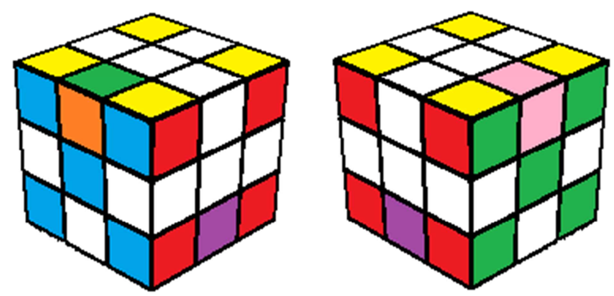
B' U M' U' M B (clever ways to finger trick)
Becomes triplet if UR fits into UL



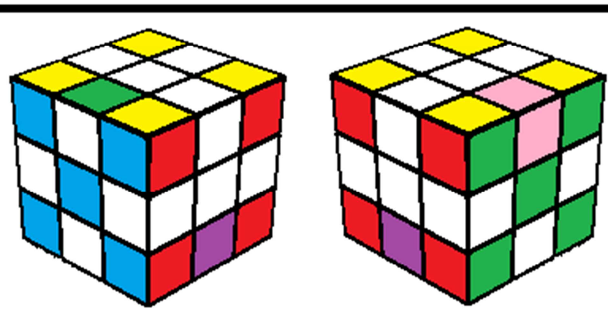
M' D' M U M' [U'+D]
Becomes triplet if UR fits into UL



M F U M2 U' F'
Becomes triplet if UR fits into UL

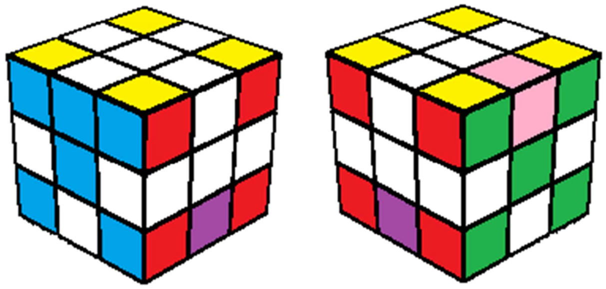


B U M' U' B' (clever ways to fingertrick)
Becomes triplet if UR fits into UL

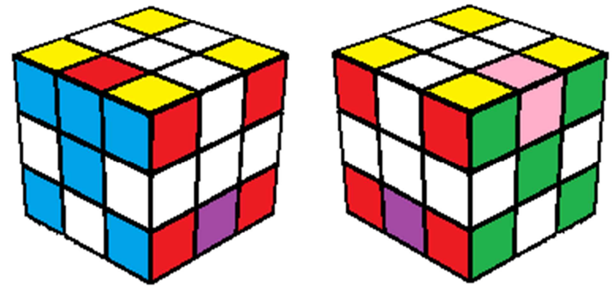


M' D M U M' [U'+D']
Becomes triplet if UR fits into UL

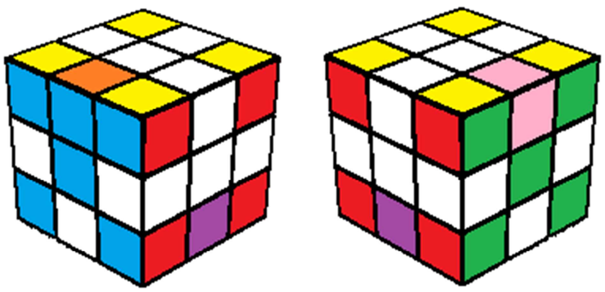
LMCF E2L algorithms, solving DF->UR, and solving UL at the same time



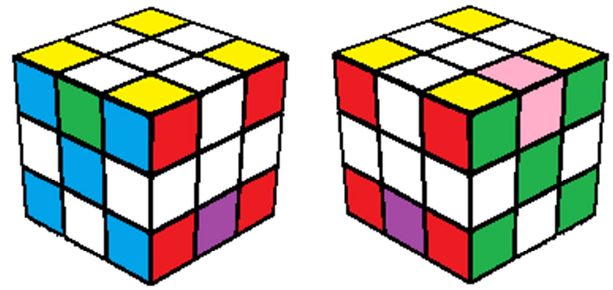
U M U' l' L' U M' U'



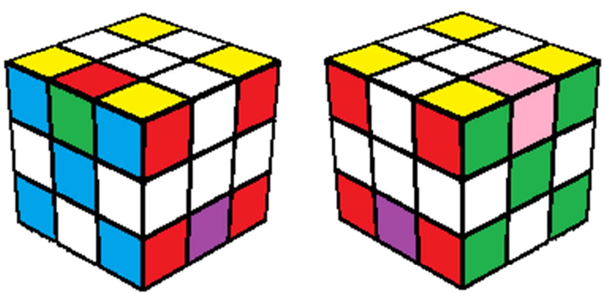
U l' U M U' L U
(becomes a triplet if UR fits in UL)



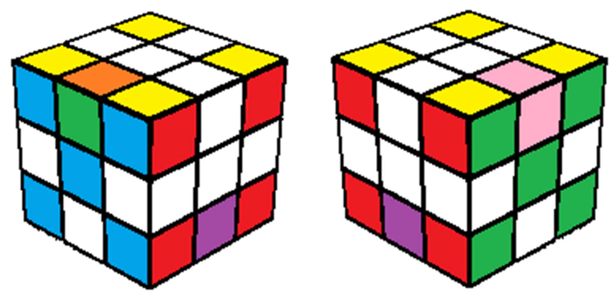
B U M' U' B' --> there are clever ways to fingertrick
This becomes a triplet if UR fits in UL



U M' U' R2 [Link to next pair]
Direct solve is not finger friendly



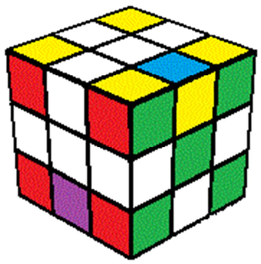
U r U' M U R' U'
Becomes a triplet if UR fits into UL



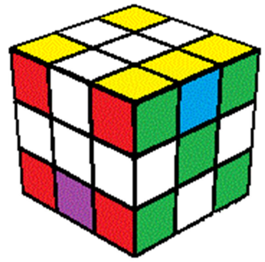
B' U M' U' B (clever ways to finger trick)
Becomes a triplet if UR fits into UL

LMCF E2L algorithms (continued)

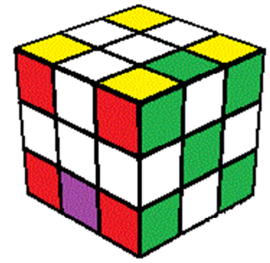
Solving DF->UR and simultaneously solving UR



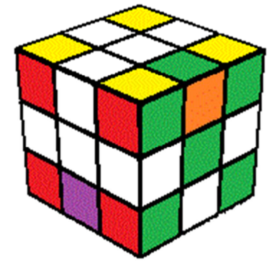
U M' U'



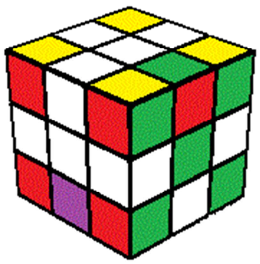
M U M U2 M' U



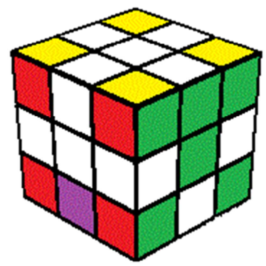
U2 R2 U' M' U R2 U2,
or r2 y M' U M U'



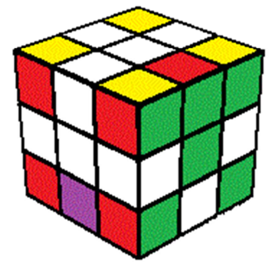
U2 R' U' M' U R U2



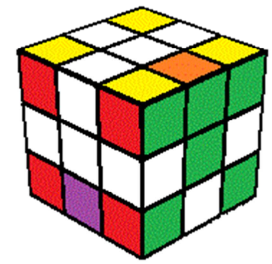
U2 R U' M' U R' U2



U M' U R2 U' M' U
Try to solve DR same time



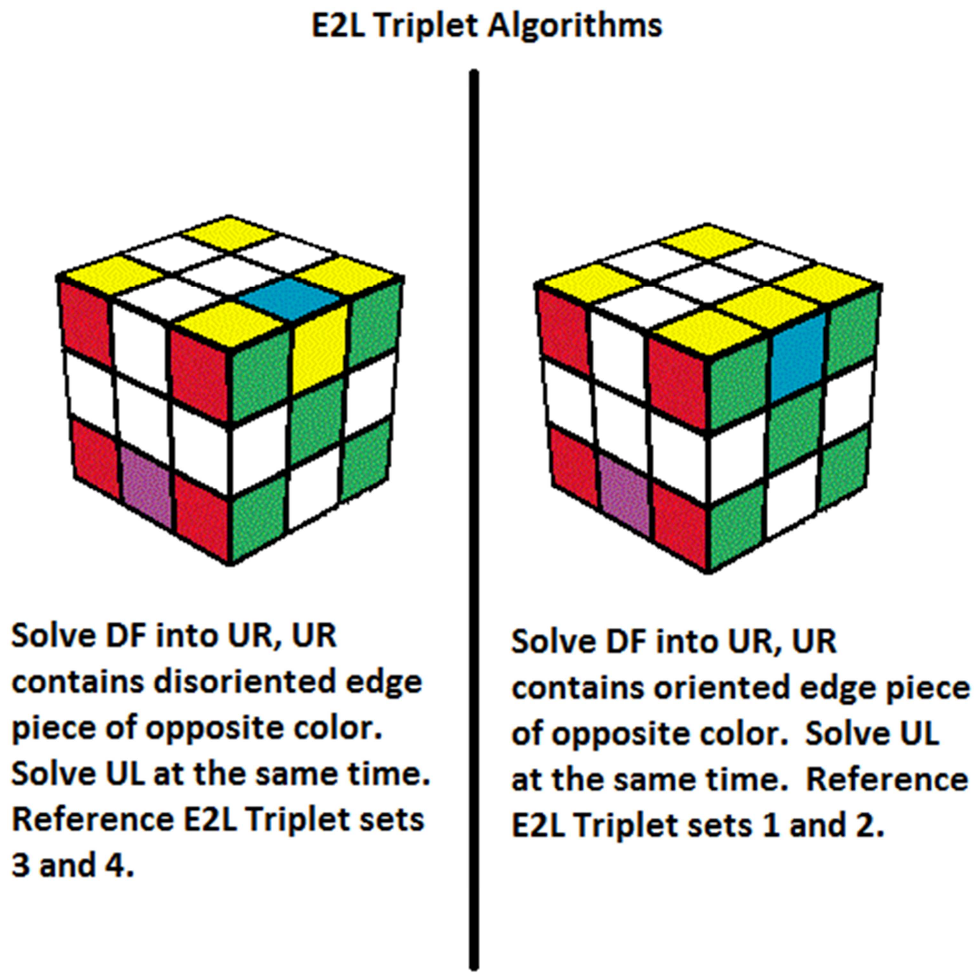
U M' U R U' M' U
Try to solve FR same time



U M' U R' U' M' U
Try to solve BR same time

E2L Triplets Set

The triplets set will solve DF, UL and UR at the same time. There is some overlap with the pairs set. There are 24 cases, but some were already covered in the pairs set, so this set introduces 16 unique algorithms (including reflections L-R and F-B it will be 72 algorithms).



Triplets Set 1:

Solve DF into UR, UR contains the oriented edge piece of opposite color, solve UL at the same time:

UL oriented

1. UL->FL: r' U F' D2 M' D2 F U' [8] or M U M2 U L' U M2 U' L U2 [10]
2. UL->BL: M U M2 U L U M2 U' L' U2 [10]
3. UL->DL: M U M2 U L2 U M2 U' L2 U2 [10]

UL disoriented

4. UL->FL: U' L' U2 M2 U M' U M' r B [9]
5. UL->BL: U' L U2 M2 U M' U R' F [9]
6. UL->DL: U M2 U2 M2 U L2 U' M U [9] or D' M U M' D M2 U2 M' U [9]

Triplets Set 2:

UL oriented
7. UL->FR: M U M2 U' R U' M2 U [8] or U M2 U2 M' U R U' M' U [9]
8. UL->BR: M U M2 U' R' U' M2 U [8]
9. UL->DR: U M2 U2 M' U R2 U' M' U

UL disoriented
10. UL->FR: L' U R2 B' M' B R2 U' [8]
11. UL->BR: L' U B' M' B U' [6]
12. UL->DR: L' U R' B' M' B R U' [8]

Triplets Set 3:
Solve DF into UR, UR contains the disoriented edge piece of opposite color, solve UL at the same time:

UL oriented
UL->FL: U l' U M U' L U → overlaps with E2L pair set
UL->BL: B U M' U' B' [5] → overlaps with E2L pair set
13. UL->DL: U l' L' U M U' L2 U' or M' D M U M' U' M' D'

UL disoriented
14. UL->FL: U' M U L' U M2 U'
UL->BL: B' U M' U' M B → overlaps with E2L pair set
UL->DL: M' D' M U M' [U'+D] → overlaps with E2L pair set

Triplets Set 4:

UL oriented
UL->FR: U r U' M U R' U' -> overlaps with E2L pair set
UL->BR: B' U M' U' B [5] -> overlaps with E2L pair set
15. UL->DR: M' D' M U M' U' M' D [8]

UL disoriented
16. UL->FR: L U M' F M F' U' [7]
UL->BR: B U M' U' M B' [6] -> overlaps with E2L pair set
UL->DR: M' D M U M' [U' D'] [6] -> overlaps with E2L pair set

4. LAST SIX EDGES (L6E)

This section covers the solving the last six edges. Numerous algorithm sets are presented. Some are more important than others in the sense that they occur FAR more often than others. I did statistical analysis of 300 solves and tabulated the frequency that each of the L6E algorithm sets occurred to show which are important to learn in which order. Remember you can always FORCE a particular set to occur by intuitive solving but this increases your move count.

Set Name	**Frequency of Occurrence**
Most Important:	
Direct L5E (DFL, DFR, BDL, BDR, 32 cases):	42.7%
Medium Importance:	
One of UL/UR disoriented, the other solved (4 cases):	9.0%
x-set (xDFR, xBDR, xDFL, xBDL, 32 cases):	8.7%
r-set (rDFR, rDFL, rBDR, rBDL, 32 cases):	8.3%
i-set (iDFR, iDFL, iBDR, iBDL, 32 cases):	8.0%
Waterman Set 2 (96 cases):	6.3%
Waterman Set 3 (96 cases):	5.7%
Waterman Set 1 (96 cases):	4.7%
UL/UR solved and disoriented (4 cases):	3.3%
Low Importance:	
Waterman Set 6 (24 cases):	1.0%
Waterman Set 7 (24 cases):	0.7%
Waterman Set 4 (24 cases):	0.7%
UL/UR swapped but oriented (4 cases):	0.7%
Waterman Set 5 (12 cases):	0.3%

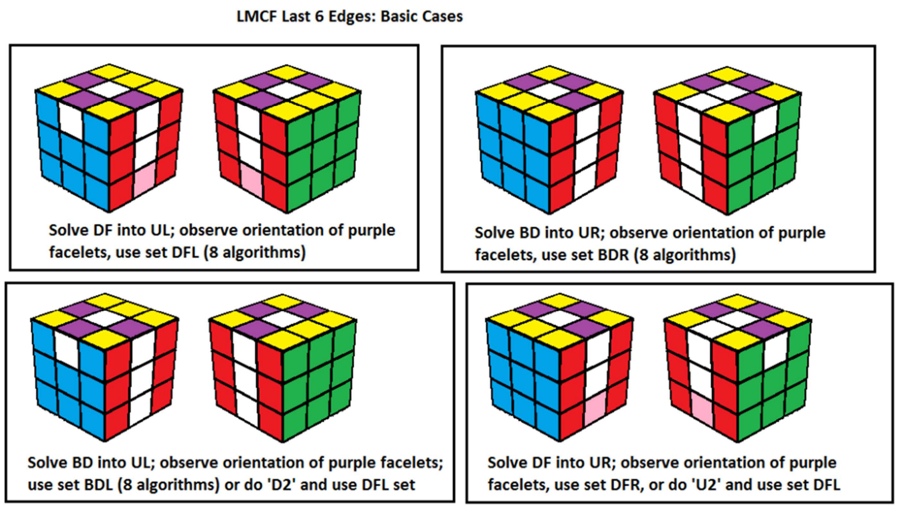
Recommended learning:

I recommend learned the direct L5E set first as well as the 4 cases of [UL/UR solved, the other disoriented], plus the pure midge flips (UL/UR both solved).
Next I recommend learning the double disoriented (UL/UR both solved & disoriented) as well as the x-set. The next is the i-set, followed by Waterman Set 2, and then Set 3. The last set I recommend is Waterman Set 6. The others I think occur too infrequently and are easily transposed. The r-set for example can be transposed in 3 moves to a direct set.

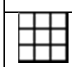
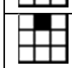
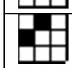
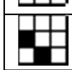
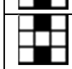
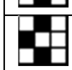
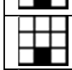
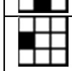
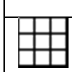
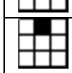
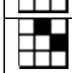
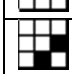
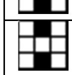
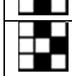
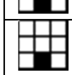
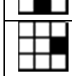
I also did statistics on how often EG1, EG2 and CLL cases arise, considering that I only know CLL and EG1 and try to avoid EG2:

EG1: 70.4%
CLL: 17.9%
EG2: 11.6%

The first L6E case is both UR and UL are unsolved. This is similar to Roux LSE except the L/R faces may be misaligned, making Roux LSE recognition difficult. The first variant (Direct set) is where you intuitively solve UR or UL, leaving just one of UL/UR unsolved as in the following diagram. When observing the orientation of the purple facelets, compare them with the U center piece; same or opposite colors compared to the center means oriented, and random color means disoriented.

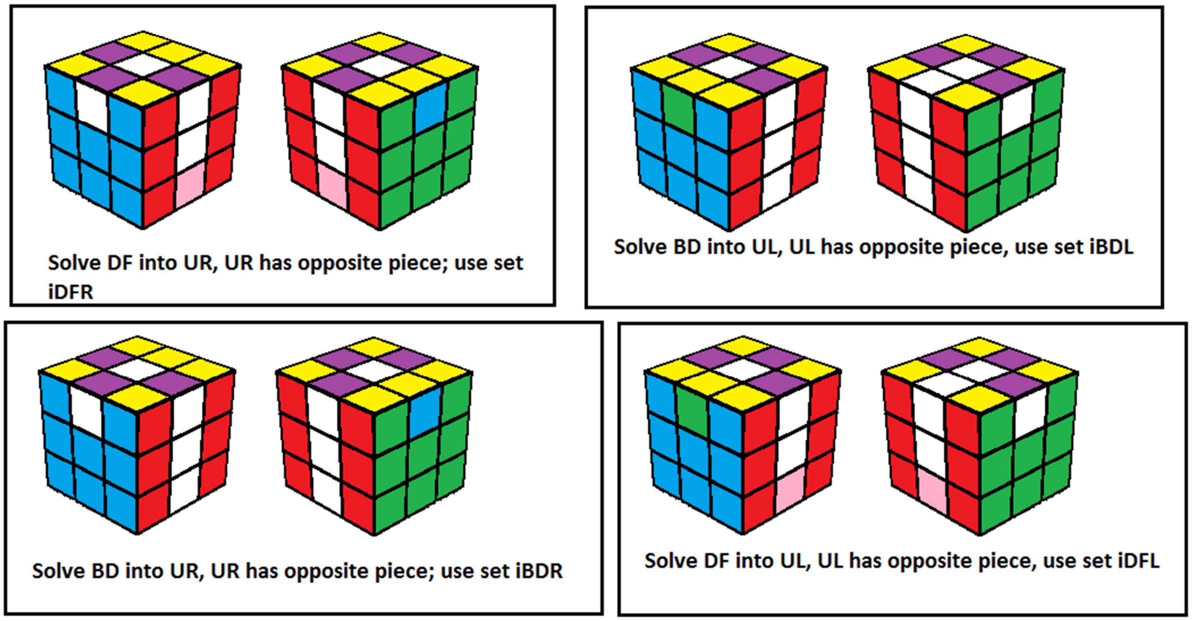


Average move count for this set is 8.12.

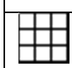
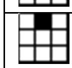
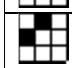
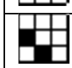
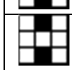
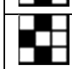
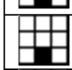
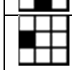
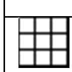
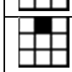
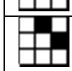
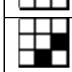
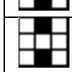
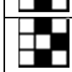
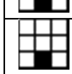
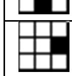
Top 3 facelet orientations (black=bad)	**DFL (DF->UL + orient midges)**	**BDL (BD->UL + orient midges)**
	M' U M' U' M U' M' U [8]	M U' M U M' U M U'
	M' U' M U' M' U' M U' [8]	U2 M U' M' U' M U' M' U
	U' M U M' U' M' U [7]	U M' U M U' M U'
	U' M U' M' U M' U [7]	U M' U' M U M U'
	U' M' U' M U' M U M' U M U [11]	U M U M' U M' U' M U' M' U'
	U' M' U' M' U' M U2 M U [9]	U M U M U M' U2 M' U'
	U2 M' U M U M' U M U' [9]	M U M' U M U M' U
	M U M' U2 M' U [6]	M' U' M U2 M U'
	DFR (DF->UR + orient midges)	**BDR (BD->UR + orient midges)**
	M' U' M' U M U M' U'	M U M U' M' U' M U
	M' U M U M' U M U	U2 M U M' U M U M' U'
	U M U' M' U M' U'	U' M' U' M U M U
	U M U M' U' M' U'	U' M' U M U' M U
	U M' U M U M U' M' U' M U'	U' M U' M' U' M' U M U M' U
	U M' U M' U M U2 M U'	U' M U' M U' M' U2 M' U
	U2 M' U' M U' M' U' M U	M U' M' U' M U' M' U'
	M U' M' U2 M' U'	M' U M U2 M U

The second important variant is the i-set where one of UL/UR contains the opposite edge piece as shown in the following diagram:

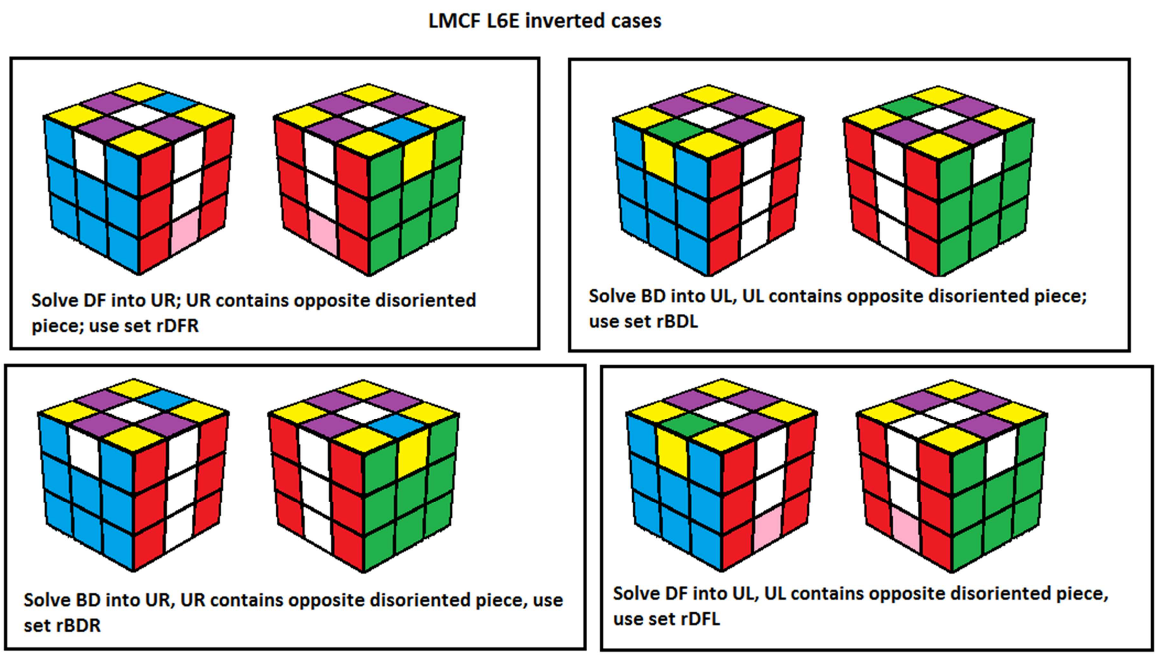
LMCF L6E inverted cases



Average move count for this set is 8.50.

Top 3 facelet orientations (black=bad)	**iDFR (DF->UR + orient midges)**	**iBDR (BD->UR + orient midges)**
	M' U M' U' M U M2 U2 M' U [10]	Use reflections
	M' U M2 U2 M U M U M' U' [10]	
	U M2 U M' U M' U [7]	
	U M2 U' M' U' M' U [7]	
	r F R U' M' U R' F' [R] [8]	
	U' M' U M' U M' U2 M U [9]	
	U2 M' U M U M' U M' U2 M2 U [11]	
	M U M U2 M' U [6]	
	iDFL (DF->UL + orient midges)	**iBDL (BD->UL + orient midges)**
	Use reflections	Use reflections
		
		
		
		
		
		
		

The third variant is the r-set where UR/UL is has the opposite edge, disoriented:

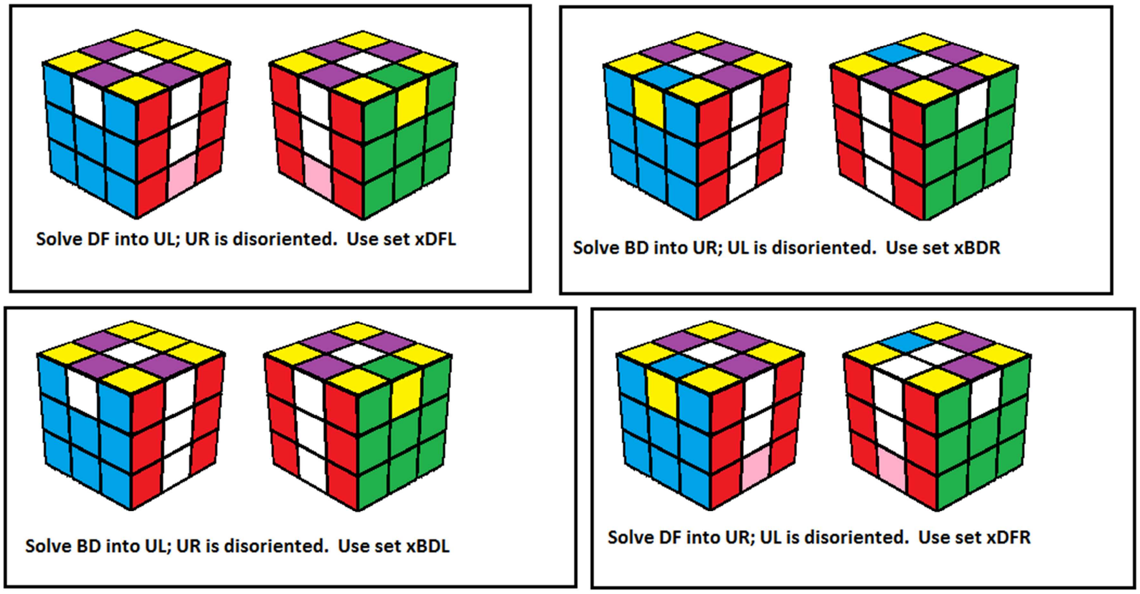


Average move count for this set is 7.87.

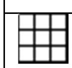
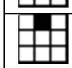
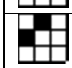
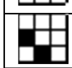
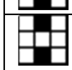
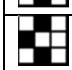
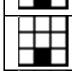
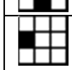
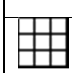
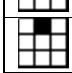
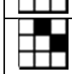
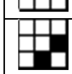
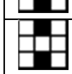
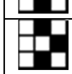
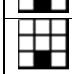
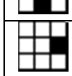
Top 3 facelet orientations (black=bad)	**rDFR (DF->UR + orient midges)**	**rBDR (BD->UR + orient midges)**
	U' M' U' M U' M' U M' U M2 U [11]	Use reflections
	U2 M' U2 M' U' M U [7]	
	U M2 U' M' U2 M U' M U' [9]	
	U M2 U M' U2 M U M U' [9]	
	U M U' [3]	
	M' U M' U2 M' U M U M' U' [10]	
	M' U2 M' U M U [6]	
	r B [M' R'] U M U' R B' [8]	
	rDFL (DF->UL + orient midges)	**rBDL (BD->UL + orient midges)**
	Use reflections	Use reflections

The fourth variant is the x-set where UR is flipped:

LMCF L6E inverted cases

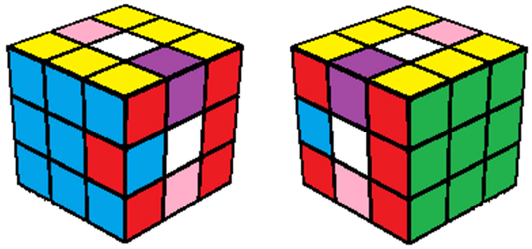


This set has an average move count of 9.00.

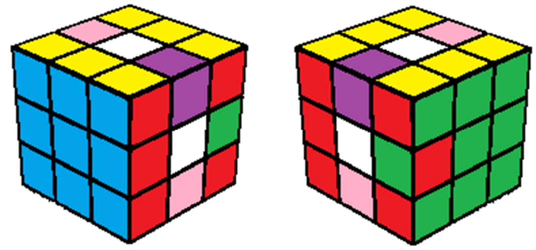
Top 3 facelet orientations (black=bad)	**xDFL (DF->UL + orient midges)**	**xBDL (BD->UL + orient midges)**
	U' M U' M' U M2 U M' U M' U' [11]	Use reflections
	U2 M' U2 M' U M2 U2 M U [9]	
	U' M2 U' M' U2 M' U' M U [9]	
	U' M2 U M' U2 M' U M U [9]	
	U M U2 M2 U [5]	
	M U M' U M U M U2 M' U' [10]	
	M' U2 M' U M' U2 M2 U' [8]	
	U' M U r U' r' U M' U' r U [11] U M2 U' M' U' M2 U M U M' U M U M' U [15]	
	xBDR (BD->UR + orient midges)	**xDFR (DF->UR + orient midges)**
	Use reflections	Use reflections
		
		
		
		
		
		
		

More special cases:

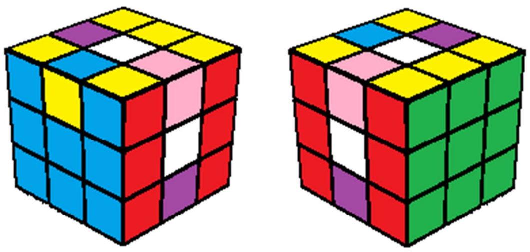
LMCF L6E One edge inverted



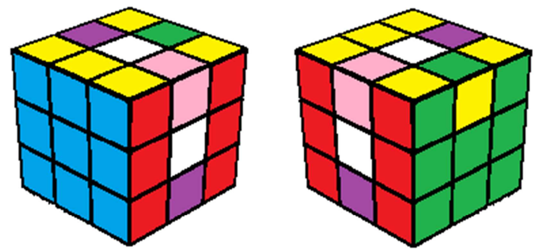
Purple edge is bad: U L' U M' U2 M2 U L U'



Purple edge is bad: U' R U' M' U2 M2 U' R' U



Pink edge is good: M' U M' U M' U M' U



Pink edge is good: M' U' M' U' M' U' M' U'

UR and UL both solved and disoriented

2 adjacent disoriented midges at top: U M' U2 M U2 M' U' [7] or U M2 F2 M' F2 U' [6]
2 diagonal disoriented midges, UF and BD: U M2 U [M' U M' U M' U] M U' [11]
0 disoriented midges: U M' U M' U M2 U M' U M' U' [11]
4 disoriented midges: R' F R U M' U2 M2 U R' F' [10] (+R)

UR and UL both swapped and oriented:

0 disoriented midges: U M2 U2 M2 U [5]

2 disoriented midges diagonal, DF and UB:

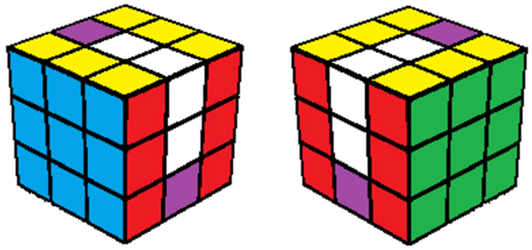
U x' U R U M2 U' R' U' F' or
x' F U R U M2 U' R' U' F' [9] or
2-gen: U M U' M' U2 M' U M' U2 M2 U' [11]

2 disoriented midges adjacent (front): U' M U' M' U2 M' U M' U2 M2 U [11]

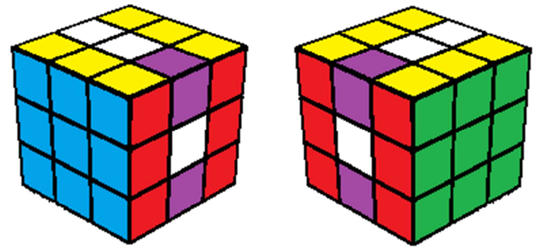
4 disoriented midges:

U M' U2 F2 M2 U M2 U M U2 F2 U [12] or
U M U2 M U r U' r' U M' U' r U [13]
(or 2-gen: U M' U' M' U' M2 - U M U M' U M U M' U [15])

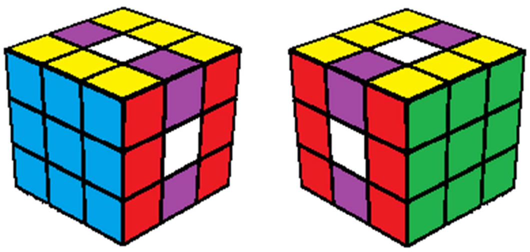
LMCF L6E: UL and UR have been accidentally solved and you must orient the midges:



2 disoriented midges at DF and UB:
U' M U M U2 M' U M' U [9]



2 disoriented midges at UF and DF:
U M U M U2 M' U M' U' [9]

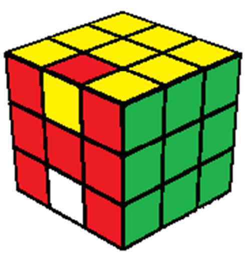


U r' U M U' M2 R U' R' U M U' [12]
U'M U2 M' U' M' r' F R U' M' U R' F' [14]
U' M U M U' M' U' M' U M U M' U M U' [15]

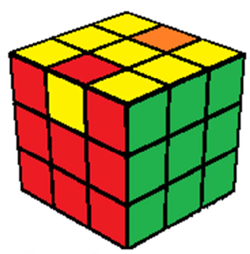
4 disoriented midges
Choose any of these

In the case that the entire cube is accidentally solved and all that is left is to flip edge pieces, these special cases should be mastered.

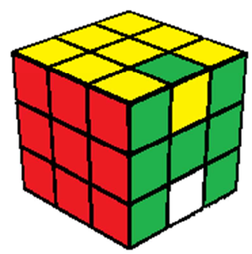
PURE EDGE FLIP CASES:



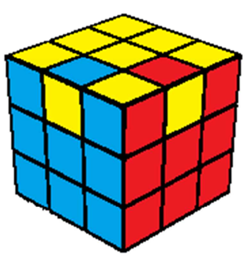
2 edges flipped at UF, DF:
U M U M U2 M' U M' U M' U2 M [12]



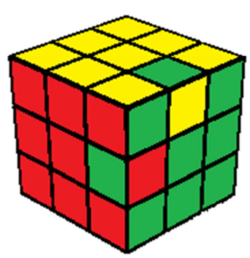
2-edges flipped at UF, UB:
M' U M' U M' U M' U2 M' U M' U M' U M' [15]



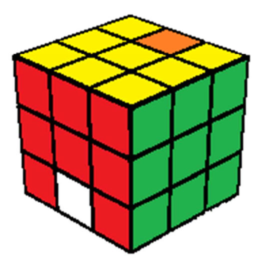
2-edges flipped at UR and DR:
U M U2 M2 U R2 U' M2 U2 M' U' R2 [12]



Adjacent edges flipped at UL, UF:
M2 U M U2 M' U M' U M' U2 M U M' [13]



Adjacent edges flipped at UR, FR:
U M U2 M2 U R U' M2 U2 M' U' R' [12]



Diagonal opposite edges are flipped at UB and DF:
U' M U M U2 M' U M' U M' U2 M U2 [13]

UR and UL both solved (by accident):

This case should only happen by accident, as recognition is slower than standard L6E, but the algorithms are below (exactly as shown in the above graphic):

0 disoriented midges: Skip and go to permute midges!

2 disoriented midges at DF and UB: U' M U M U2 M' U M' U [9]

2 disoriented midges at UF and DF: U M U M U2 M' U M' U' [9]

4 disoriented midges:
U r' U M U' M2 R U' R' U M U' [12]
U' M U2 M' U' M' r' F R U' M' U R' F' [14]
2 gen: U' M U M U' M' U' M' U M U M' U M U' [15]

If DF is already solved and you have just three edge pieces unsolved, you can solve them in a single algorithm 50% of the time, either of these two:
y' L2 U M' U' L2 U M U' [1-look solve if UF and UB are disoriented]
y' U M' U' L2 U M U' L2 [1-look solve if UF and BD are disoriented]

PURE EDGE FLIPS
If the entire cube ends up solved except for edge flips:

Diagonal opposite edges are flipped:
UB and DF: U' M U M U2 M' U M' U M' U2 M U2 [13]

2 edges are flipped (these are interchangeable based on setup)
UF and DF: U M U M U2 M' U M' U M' U2 M [12]
UF and UB: M' U M' U M' U M' U2 M' U M' U M' U M' [15]
UR and DR: U M U2 M2 U R2 U' M2 U2 M' U' R2 [12]

Adjacent edges are flipped:
UL and UF: M2 U M U2 M' U M' U M' U2 M U M' [13]
UR and FR: U M U2 M2 U R U' M2 U2 M' U' R' [12]

IMPORTANT TIPS FOR BASIC L6E

It is very important to consider the average move count of the main sets:
DFL set: 8.12
xDFL set: 9.00
rDFR set: 7.87
iDFR set: 8.50

So it is clear that if UR and UL are unsolved, then BY FAR your best choice is to insert one of the two edges in the opposite slot, disoriented, and use the rDFR set which has an average move count of 7.87, including one case with just 3 moves.

Advanced LMCF L6E (Waterman's Method)

The second case is where you have two edges unsolved on one side (L/R) and the other side is fully solved. This is the most advanced part of LMCF and the dreaded last step of the Waterman method. Please understand you can always transpose this scenario to one of the earlier L6E cases by intuitive solving, in some cases these transpositions are very fast. However for minimum move count you should use Waterman's algorithms. These algorithms are all set up for the unsolved redges on the R layer, where in LMCF the unsolved redges could be in the L layer. You can do an inefficient y2 rotation or better yet learn the L-R reflections of the algorithms.

I have normalized the set up for these algorithms to use modernized orientation recognition, making recognition much faster than Waterman's original method. The orientation instructions are included in each set.

The following diagram shows each of the cases and which Waterman algorithm set you must use to solve that case. After the diagram you will find the list of each algorithm set.

I will also mention that any case that was more than 13 moves in the original Waterman method has been recalculated to a version that is 12 moves or less while still maintaining ergonomics. In some cases multiple algorithm options have been provided.

Experts will notice that the number of moves saved using Waterman's L6E is not that many, but what is important is your TPS will be very high, not only because the algorithms are ergonomic but because you can execute the solution at algorithm speed (high TPS), leaving only the very fast midge permutation step.

Roux Application

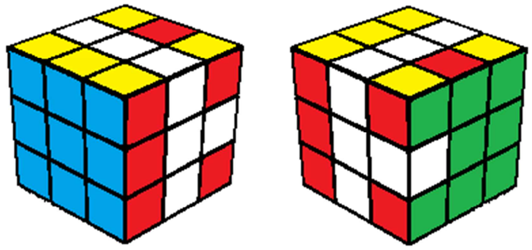
There is a significant application of Waterman L6E to Roux. Using Waterman L6E, if you are solving the second (right) block on a Roux solve, you can skip any of the second block edges if the edge is disoriented in its place, and then instead of using Roux LSE, you finish the solve with Waterman L6E which will automatically orient the bad edge during the L6E phase.

Further Advanced Variants: L7E

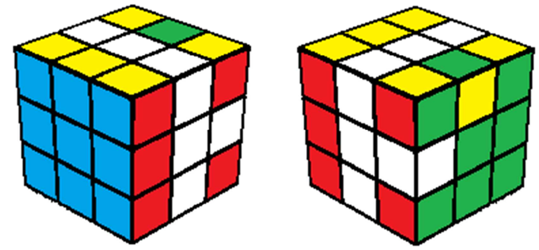
The most advanced finish to an LMCF solve is to finish with L7E (last seven edges). In this case, you are faced with Waterman set 4, 5, 6 or 7 (with two redges in the R face but incorrectly solved). However for L7E, rather than having the L face fully solved, the UL slot is unsolved. This means you have 7 edges still remaining in the solve, and you solve the last ledge and both redges while orienting the midges. Recognition is extremely fast (same top 3 facelets), but the number of cases is high. This method is extremely fast because when using L7E you only need to solve 5 edge pieces in the E2L phase (typically one pair and one triplet if you have mastered all the triplet algorithms). Move count will not only be lower but TPS increases even more because now the last 7 edges are algorithmic. You might argue that not all solves will end up with the situation where the last two redges are in the R face but unsolved; on the contrary, if you know L7E, then when solving E2L, when you see redges that are in the R face but incorrectly solved, ***<u>you consider them already solved</u>***, because you know you can do them in the L7E step. This means that almost every solve can be made to deliberately end up with an L7E case. L7E can also be used with Roux as explained above.

The L7E algorithms can be found here:
https://drive.google.com/open?id=0B2QnZ3uD6I8kbnRRM0sxSDhHbkk

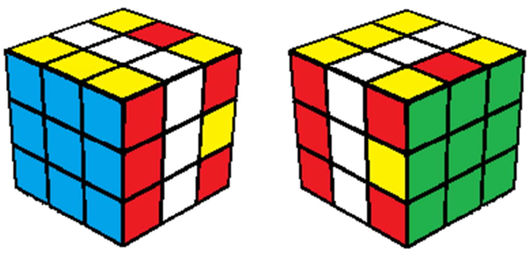
LMCF Advanced L6E: Two edges unsolved on the same side; solve both at once and simultaneously orient the midges



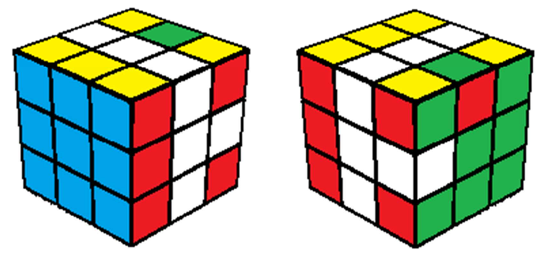
Waterman Set 1: Redge in R layer correctly oriented in the position of other redge



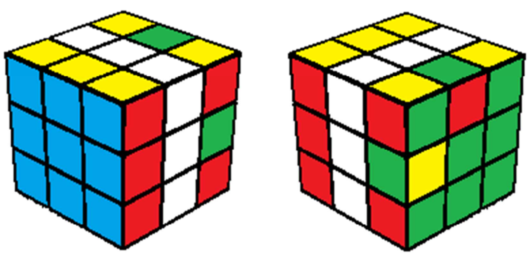
Waterman Set 3: Redge in R layer flipped in its position (OR You can easily transpose to iDFR set by U M U2 M2 U)



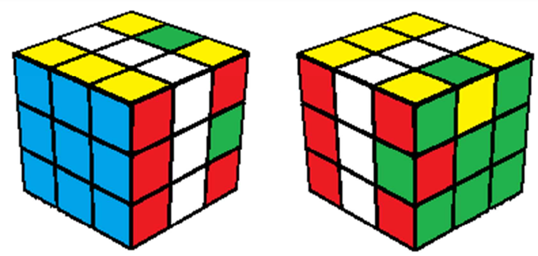
Waterman Set 4: Last two redges are oriented correctly but interchanged



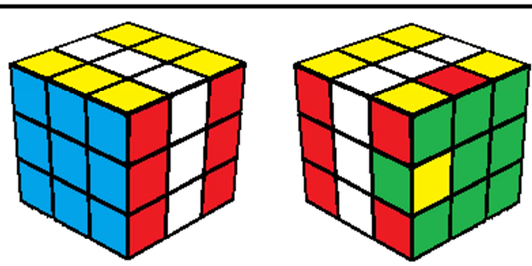
Waterman Set 2: Redge in R layer flipped in the position of the other redge (You can transpose to iDFR set by U M' U')



Waterman Set 6: Last two redges are interchanged and both redges are flipped



Waterman Set 7: Last two redges are flipped in their correct positions

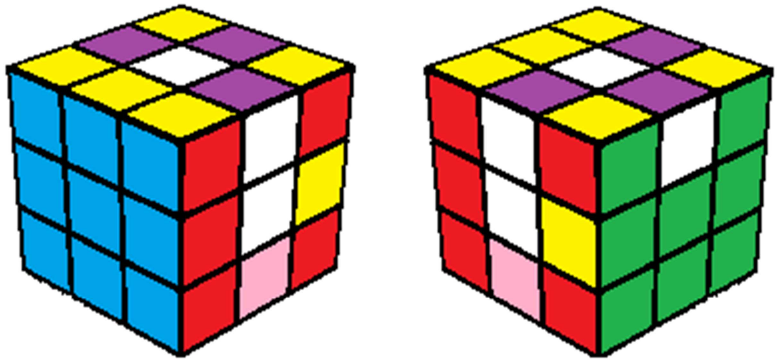


Waterman Set 5: Last two redges are interchanged and one of the last two redges is flipped

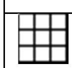
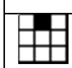
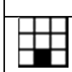
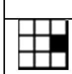
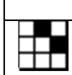
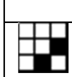
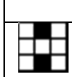
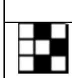
Set 1: 48 cases; Set 2: 48 cases
Set 3: 48 cases; Set 4: 12 cases
Set 5: 6 cases; Set 6: 12 cases;
Set 7: 12 cases

So Waterman L6E has a total of 186 cases when solving on the R-face, and another 186 reflections on the L-face.

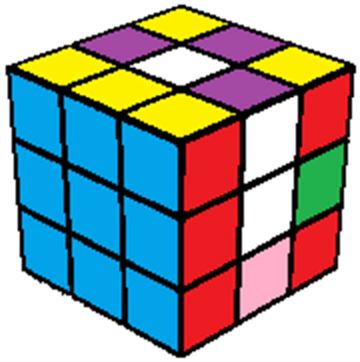
Waterman Set 1: Redge in R layer correctly oriented in the position of the other redge (48 cases):



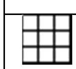
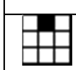
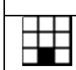
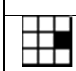
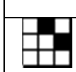
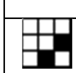
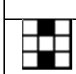
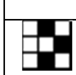
Waterman Set 1: One redge is oriented but in the wrong position. The green-red edge is in the M-slice, at either DF or BD, with its green facelet on the D-face. Examine the three purple facelets to determine the orientation case, with respect to the U center piece. There are 24 cases for green-red at DF, and 24 cases for green-red at BD, 48 cases total. The colors are just for example & clarity.

	Algorithm for Green-Red at DF position	Redge @	Algorithm for Green-Red at BD position
	U' M2 U' R' U M U' R U' M' U' [11] U' M2 U' R U M U' R' U' M' U' [11] U' M2 U' R2 U M U' R2 U' M' U' [11]	BR FR DR	Reflections coming soon!
	M' R' U M U' M U r U M' U' R' U' [13] or B' L U' M U R' U M' U M' U' [12] r U M U' M U M' R' U M' U' R U' [14] or M' U' M' U M2 F U M' U' F' [10] or U R U M' U' R' U M' U M' U2 M' U' [13] r2 U M' U' M' U r2 U M' U' R2 U' [12] or R' U R U R U2 R2 L F' R' F' [11]	BR FR DR	
	x' U' M' U' R U' M U R' U' M2 U' [11] R M U' M' U' R' U' M U R U' M2 U' [13] or M U' M' U M F U M' U' F' [10] or x M2 U' M' U' R' U' M U R U' M2 U' [12] R2 M U' M' U' R2 U' M U R2 U' M2 U'[13]	BR FR DR	
	R' U M' U' R U M U' [8] R U M' U' R' U M U' [8] R2 U M' U' R2 U M U' [8]	BR FR DR	
	R' U M' U M' U2 M R U M U' [11] R U M' U M' U2 r' U M U' [10] R2 U M' U M' U2 r' R' U M U' [11]	BR FR DR	
	R' U M' U' M R U2 M U M' U [11] R U M' U' r' U2 M U M' U [10] R2 U M' U' r' R' U2 M U M' U [11]	BR FR DR	
	R' U M R U M' U' M' R' U'[10] R U r' U M' U' r U' [8] R2 U r' R' U M' U' r R U' [10]	BR FR DR	
	M' R' U M R U M' U' R' U2 M U [12] x U r' U M' U' R U2 M U [9] M' R2 U r' R' U M' U' R2 U2 M U [12]	BR FR DR	

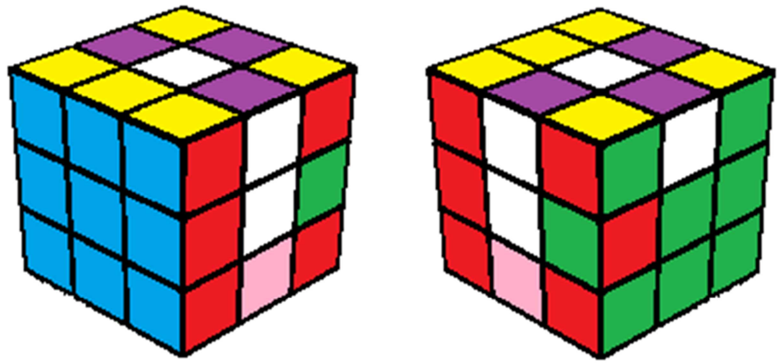
Waterman Set 2: Redge in R layer flipped in the position of the other redge (48 cases):



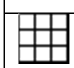
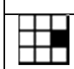
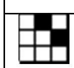
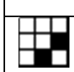
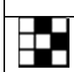
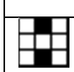
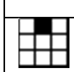
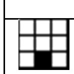
Waterman Set 2: One redge is flipped and in the wrong position. The other redge (green-red) is in the M-slice at either DF or BD, with the green facelet on the D-face. Examine the three purple facelets to determine the orientation case with respect to the U-center. There are 24 cases for green-red at DF, and 24 cases for green-red at BD, 48 cases total. The colors are just examples.

	Algorithm for Green-Red at DF position	Redge @	Algorithm for Green-Red at BD position
	R' U M2 U R U' M U R' U' M U' [12] R U M2 U R' U' M U R U' M U' [12] R2 U M2 U R2 U' M U R2 U' M U' [12]	BR FR DR	Reflections coming soon!
	R' U2 M R U2 M' U M' U R' U2 [11] R U2 r' U2 M' U M' U R U2 [10] R2 U2 r' R' U2 M' U M' U R2 U2 [11]	BR FR DR	
	M' R' U2 r U M U' R' U2 [9] r U2 M' R' U M U' R U2 [9] M' U2 r R U M U' R2 U2 [9]	BR FR DR	
	R' U R' r2 U' M U r' U' [9] R U M2 R' U' M U R M U' [10] R2 U r2 U' M U r' R' U' [9]	BR FR DR	
	M2 R' U' R U2 M U M2 U r' U [11] M2 R U' R' U2 M U M2 U M R U [12] r2 U' R2 U2 M U M2 U r' R' U [11]	BR FR DR	
	R M2 U' r' U M U' r2 R' U [10] M2 U' M R U M U' M2 R' U [10] M2 U' r' R' U M U' r2 U [9]	BR FR DR	
	R' U2 R U' M' U R' U2 [8] R U2 R' U' M' U R U2 [8] R2 U2 R2 U' M' U R2 U2 [8]	BR FR DR	
	R' U M2 U M' U' M U' r U' M2 U [12] R U M2 U M' U' M U' M' R' U' M2 U [13] or M U r U M U M' U' M' U' r' U' [12] M' U M U M D U M U M' D' [12]	BR FR DR	

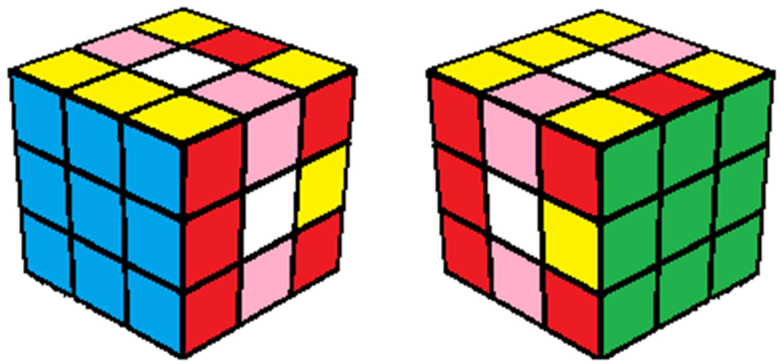
Waterman Set 3: Redge in R layer flipped in its position (48 cases):



Waterman Set 3: One redge is flipped in its position and the UR slot is empty. The green-yellow edge is in the M-slice at either DF or BD position, with the green facelet on the D-face. Examine the three purple facelets to determine the orientation case with respect to the U-center. There are 24 cases for green-yellow at DF, and 24 cases for green-yellow at BD, 48 cases total.

	Algorithm for Green-Yellow at DF position		Redge@	Algorithm for Green-Yellow at BD position
	U M' U' R' U' M' U2 M2 U' U M' U' R U' M' U2 M2 U' U M' U' R2 U' M' U2 M2 U'	[9] [9] [9]	BR FR DR	Reflections coming soon!
	R' U M U2 r U M' U' M' R' U R U M U2 M' R' U M' U' r U R2 U M U2 r R U M' U' M' R2 U	[11] [11] [12]	BR FR DR	
	M2 R' U' R U' M U' M U M U r' U M U' r U2 M' U M2 U2 M' U' R' U r2 U' R2 U' M U' M U M U r' R' U or M R2 U M U' R2 U2 M' S' U' S U'	[12] [12] [13] [11]	BR FR DR	
	M2 R' U' M R U M U' r' U2 M' U' r M' U' r' U M U' M R U2 M' U' r2 U' r' R' U M U' r' R' U2 M' U'	[12] [12] [12]	BR FR DR	
	M2 R' U' R U2 M U' M U' r' U r M' U' R' U2 M U' M U' M R U r2 U' R2 U2 M U' M U' r' R' U	[11] [12] [11]	BR FR DR	
	R' U2 R U' M U2 M2 U' R' U2 R U2 R' U' M U2 M2 U' R U2 R2 U2 R2 U' M U2 M2 U' R2 U2	[10] [10] [10]	BR FR DR	
	M2 R' U M U2 M2 U R U2 M U M2 U r M' U M U2 M2 U R' U2 M U M2 U or R U B' U2 M' U M' U M' B U' M2 R2 U M U2 M2 U R2 U2 M U M2 U or M S' U M2 U2 M U M S	[13] [13] [11] [13] [9]	BR FR DR	
	M2 R' U M2 U2 M U r U M2 U' [12] or M U M2 U' r' U' M' U2 M2 U' [10] r M' U M2 U2 M U M' R' U M2 U' [12] or M U M2 U' M R U' M' U2 M2 U' [11] r2 U M2 U2 M U r R U M2 U' [11] or M U M2 U' M R2 U' M' U2 M2 U' [11]		BR FR DR	

Waterman Set 4: Last two redges are oriented correctly but interchanged (12 cases):



Waterman Set 4: The last two redges are oriented correctly but interchanged. There are 12 cases. First you choose the location of the two unsolved redges (adjacent on the R layer, or opposite on the R layer). Then you observe the orientation of the four midges. Use the table below.

No disoriented midges:

U2 R' U M2 U2 M2 U R U2 [redges at UR and BR] [9]
U2 R U M2 U2 M2 U R' U2 [redges at UR and FR] [9]
U2 R2 U M2 U2 M2 U R2 U2 [redges at UR and DR] [9]

2 disoriented midges at UB and DB:

U r' U M2 U2 M' U r U' [redges at UR and BR] [9]
R U r' U M2 U2 M' U r U' [redges at UR and FR] [9]
U r' R' U M2 U2 M' U r R U' [redges at UR and DR] [11]

2 disoriented midges at DF and UB:

U' M' U M2 U R' U M U' r U' [redges at UR and BR] [11]
R U' M' U M2 U R' U M U' r U' [redges at UR and FR] [12]
U' M' U M2 U R2 U M U' r R U' [redges at UR and DR] [12]

4 disoriented midges:

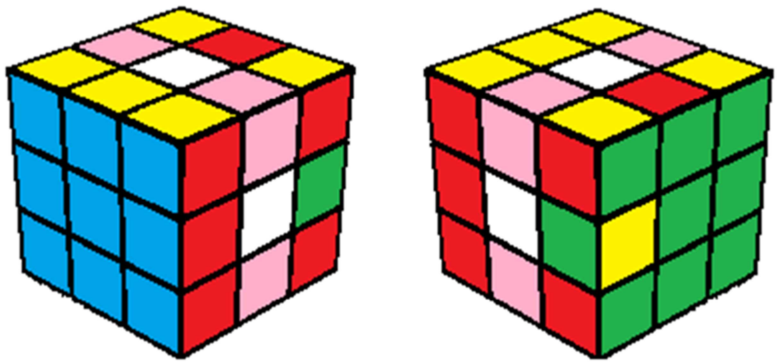
R' U M U2 R U' F2 M' F2 U r' U [redges at UR and BR][12]
(or U M U M' U2 M U R' U M U' r U')

U M U2 R U' F2 M' F2 U r' U [redges at UR and FR][11]
(or R U M U M' U2 M U R' U M U' r U')

U' [M' R2] U F2 M' F2 U' R2 U2 M' U' [redges at UR and DR][11]

12 cases, 10.33 move average.

<u>**Waterman Set 5: Last two redges are interchanged and one of the last two redges is flipped (6 cases):**</u>



Waterman Set 5: The last two redges are exchanged, one is oriented the other is disoriented. Place the oriented edge at UR. Select the algorithm based on the location of the disoriented redge, and the orientation of the 4 midges. Use the table below. There are 6 cases.

<u>One disoriented midge at DF:</u>

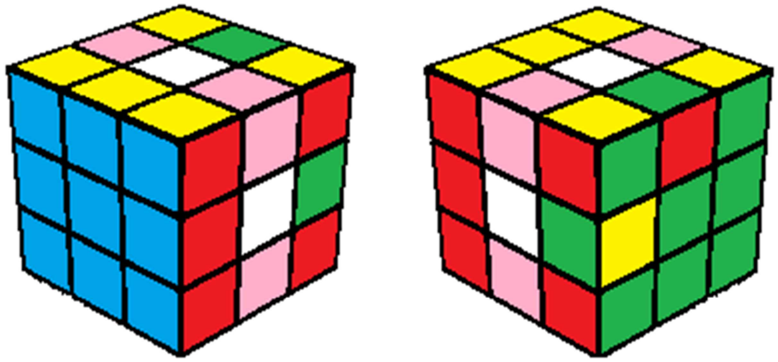
U M' U' R U' M' U R' U M2 U'	[11] [oriented redge at UR, disoriented redge at FR]
U M' U' R' U' M' U R U M2 U'	[11] [oriented redge at UR, disoriented redge at BR]
U M' U' R2 U' M' U R2 U M2 U'	[11] [oriented redge at UR, disoriented redge at DR]

<u>3 disoriented midges (oriented midge at UB):</u>

U M U' R U' M2 U r' U' M U	[11] [disoriented redge at UR, oriented redge at FR]
U M U' R' U' M2 U M R U' M U	[12] [disoriented redge at UR, oriented redge at BR]
U M U' R2 U' M2 U r' R' U' M U	[12] [disoriented redge at UR, oriented redge at DR]

6 cases, 11.33 move average.

Waterman Set 6: Last two redges are interchanged and both redges are flipped (12 cases):



Waterman Set 6: The last two redges are interchanged and both disoriented. First choose your algorithm based on whether the two redges are adjacent or opposite on the R layer. Then observe the orientations of the 4 midges and select the algorithm from the table below.

No midges disoriented:

U M U' R U' M2 U R' U M' U'	[redges at UR and FR] [11]
U M U' R' U' M2 U R U M' U'	[redges at UR and BR] [11]
U M U' R2 U' M2 U R2 U M' U'	[redges at UR and DR] [11]

2 midges disoriented at UF and DF:

U r' U' M2 U r U'	[redges at UR and BR] [7]
R U r' U' M2 U r U'	[redges at UR and FR] [8]
U r' R' U' M2 U r R U'	[redges at UR and DR] [9]

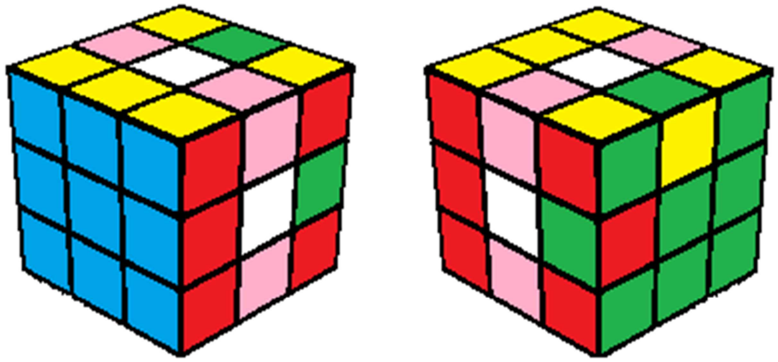
2 midges disoriented at DF and UB:

U M U2 R U' M2 U R' U2 M' U'	[redges at UR and FR] [11]
U M U2 R' U' M2 U R U2 M' U'	[redges at UR and BR] [11]
U M U2 R2 U' M2 U R2 U2 M' U'	[redges at UR and DR] [11]

4 disoriented midges:

U2 R' U M' U' r U' R' U M' U' R U'	[redges at UR and BR] [13]
R U2 R' U M' U' r U' R' U M' U' R U'	[redges at UR and FR] [14]
U2 R2 U M' U' M' R2 U' R2 U M' U' R2 U'	[redges at UR and DR] [14]

Waterman Set 7: Last two redges are flipped in their correct positions (12 cases):



Waterman Set 7: The last two redges are both in their correct position but disoriented. First select your algorithm based on whether the redges are adjacent or opposite on the R layer; then observe the orientation of the 4 midges and select your algorithm from the table below. 12 cases.

No disoriented midges:

U M U2 M2 U R' U' M2 U2 M' U'	[redges at UR and BR] [11]
U M U2 M2 U R U' M2 U2 M' U'	[redges at UR and FR] [11]
U M U2 M2 U R2 U' M2 U2 M' U'	[redges at UR and BR] [11]

2 disoriented midges at UF and UB:

U2 R' U M U2 M' U2 M U' R U2	[redges at UR and BR] [11]
U2 R U M U2 M' U2 M U' R' U2	[redges at UR and FR] [11]
U2 R2 U M U2 M' U2 M U' R2 U2	[redges at UR and DR] [11]

2 disoriented midges at DF and UB:

U2 r U M U2 M' U2 M U' R' U2	[redges at UR and FR] [11]
R' U2 r U M U2 M' U2 M U' R' U2	[redges at UR and BR] [12]
U2 r R U M U2 M' U2 M U' R2 U2	[redges at UR and DR] [12]

4 disoriented midges

U r' U M2 U2 M U r U'	[redges at UR and BR] [9]
R U r' U M2 U2 M U r U'	[redges at UR and FR] [10]
U r' R' U M2 U2 M U r R U'	[redges at UR and DR] [11]

12 cases, 10.91 move average.

FINISH THE SOLVE

Once you finish Step 4 (solve last ledge/redges and orient midges), you permute the midges with the well known sequences:

U2 M2 U2
E2 M E2
U2 M U2

At this point you should NOT need to adjust the L and R face, as you should have done that smoothly at the end of the midge orientation algorithm by looking ahead.

LMCF Beginner's Method

One of the amazing things about this method is that you can get REALLY fast (even sub-10) with a tiny algorithm set of 26-33 algorithms. This allows you to 'test' or 'try out' the method and see if it works for you, before having to commit to a huge algorithm set. In fact learning as few as 10 algorithms can get you started. The LMCF basic algorithm set is:

1. Ortega for 2x2 (12 algorithms) (or any more advanced method you know for 2x2)

2. E2L algorithms (3)
Algorithm 7 (page 8) U' M U2 M2 U' (shown graphically on page 9 top left)
Algorithm 14 (page 8) M U M U2 M' U (shown graphically on page 11 top left)
Algorithm 18 (page 8) U M' U' (shown graphically on page 11 top left)
These are BY FAR the fastest situations to recognize. In the early phase of learning E2L, recognition of the cases is difficult, but with these three cases, recognition is super fast even for a beginner. The rule for recognition is simple: Solve the DF edge into the UR slot. Does the UR slot contain an edge of the opposite color (i.e. going on to the opposite side)? If so solve it with either U M' U' (if the edge is disoriented) or M U M U2 M' U (if the edge is oriented). If the UR slot does not contain such an edge, does the UL slot contain an edge which is solved but disoriented? If so solve it with U' M U2 M2 U'. These are three very easy sequences; learn them intimately by watching what is happening with the cube and learn to execute these sequences from ANY angle (there are four reflections for each case). You can take a solved cube and execute the algorithms backwards and see exactly what they are solving (inverse is U M2 U2 M' U and U' M U2 M' U' M', and U M U').

3. L5E algorithms (8)
Learn the DFL set (page 14), 8 algorithms (solving DF->UL while orienting the midges). Using just this one set you can always solve the last ledge/redge while orienting the midges. If you end up with a reflected case, you can do D2 or U2 or y2 to 'transpose' the cube into a reflected situation for which you know the algorithm. For example, execute D2 [then do the DFL algorithm] then D2 again. Or U2 [then do the alg] then U2 again.

4. Midge permutations (3)
U2 M2 U2
E2 M E2
U2 M U2
To 'complete' your knowledge of the Basic LMCF set, the following SEVEN algorithms would be the next most important to learn in terms of how often they occur by accident. These are all midge orientation algorithms of special cases:

UR and UL both solved by accident (page 18)
2 disoriented midges DF UB: U' M U M U2 M' U M' U
2 disoriented midges UF DF: U M U M U2 M' U M' U'
4 disoriented midges: precede with U' M U M U' M' U then finish with DFL algorithm you already know (U2 M' U M U M' U M U') [this is longer than the 'fixed' version of this case but it is very easy to learn because it incorporates an algorithm you already know)

Page 16: learn all four of the situations at the top of page 16 (one edge inverted)
U L' U M' U2 M2 U L U'
U' R U' M' U2 M2 U' R' U
M' U M' U M' U M' U [extremely easy, just [M' U]x4)
M' U' M' U' M' U' M' U' [extremely easy, just [M' U']x4)

So start with the 26 algorithm set then move up to the 33 algorithm set. I would comment that since most people know Ortega and the midge permutations are intuitive as is U' M' U, there are really only TEN algorithms to learn to get going with the basic set, or 17 if you learn the extended basic set. That means that most people can learn the basic set of 10 in a single day!

ADVANCED SPEED SOLVING TIPS & COMMENTS

Generally speaking, your turns per second (TPS) will be fastest on the corner solving (EG phase) and the L6E phase, because those steps are the most algorithmic and the most ergonomic. The E2L phase is a little more intuitive, requires recognition and good lookahead, and sometimes can have a few awkward move sequences. So we expect TPS during E2L to be a little less than TPS during corners and L6E, but this doesn't matter since E2L is only a third of the solve anyway.

If you go with 8 TPS on the corners and L6E, and 6 TPS on E2L, then the full method would have predicted splits of:

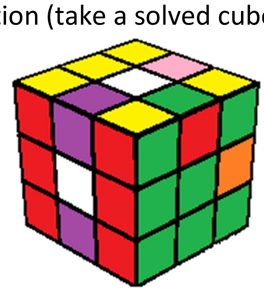
Corners: 13 moves @ 8 TPS = 1.62
E2L: 16 moves @ 6 TPS = 2.67 seconds
L6E: 13 moves @ 8 TPS = 1.62
Total average = 5.92 seconds

Looking at a real world solve, I had a full step 9.10s solve that had 2.02/4.74/2.34 splits on csTimer; that scales down to 0.96/2.25/1.11 = 4.32 second full step at expert TPS, which seems about right since on that solve the corners would definitely have been a sub-1 by a 2x2 expert and L6E was very fast as well.

It is possible to get very fast at E2L even with only the 3 basic algorithms I mention in the basic section (above). As you add more and more E2L algorithms, you need to be able to finger trick them VERY FAST in order for them to be worthwhile. If you learn a new E2L algorithm, you should be able to finger-trick the algorithm in less than 1 second for pair algorithms. If you can finger trick the E2L algorithm in less than 1 second then it should be okay to add it to your real time solve and gain an improvement when solving pairs. For E2L triplet algorithms you have significantly more time, around 1.5 seconds to finger trick the algorithm and still benefit.

Advanced L6E

Even with the basic algorithm set of just 26-33 algorithms, it is possible to get very fast times with LMCF. The problem you will face is inconsistency. Some solves may be very fast, others much slower. The reason for this will come down to the transition from E2L into L6E. On a 'smooth' solve, the last E2L pair or triplet will immediately leave you with one slot (UL or UR) empty and a smooth transition into an L5E algorithm to solve the last UL/UR piece and orient the midges. That's what you hope for and that is what will happen on 'smooth' fast solves. The problem you will face is that in perhaps 1 in 4 solves this smooth transition is not smooth at all, and you end up with a strange situation that is not favourable. As an example I will use this situation (take a solved cube and do this scramble: M U2 M' U' r' U' M2 U r U')



Waterman Set 6, Case 5: 2 midges disoriented at UF, DF

U r' U' M2 U r U'

This is an example of a situation that is considered 'unfavorable' using the basic LMCF algorithm set. You have solved the entire cube except for the last 6 edges. As you blaze through the E2L phase with very fast TPS, you suddenly find yourself with the above situation; UR and BR are disoriented and exchanged. The entire left face (blue) is entirely solved. With the basic LMCF algorithm set, you will have to do intuitive solving to convert this situation into something you can solve. One example of how to convert this would be:

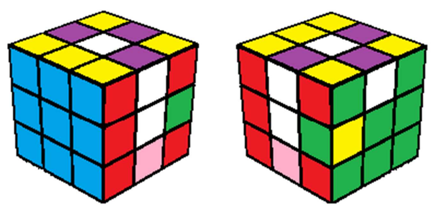
R' F' U M U' M' F M' U M' U' [11]
Then you convert this to the basic DFL L5E set with U2 [1], and perform the L5E algorithm:
M U M' U2 M' U [5]
Now we do U2 + R [2] to transpose back
Permute midges: E2 M' E2 [3]
So that 'unfavorable' L6E situation took [11]+[1]+[5]+[2]+[3] = 22 moves to solve! This doesn't happen all the time but 22 moves is a lot! This shows how the extended L6E sets are extremely valuable. If you knew Waterman Set 6, the solve is reduced to:

U r' U' M2 U r U' [7]
U2 M U2 M' [4]

So with the extended L6E set you solve the situation in 11 moves instead of 22, saving 11 moves in the process. Not all of the extended set will result in such short sequences, the average movecount for Waterman L6E is about 10.7 plus midge permutations (4), so around 14.7 moves; but if you consider that with the basic LMCF algorithm set these 'unfavorable' situations often require 18-23 moves to solve, you are saving a lot.

In fact, once you know the extended L6E set, these 'unlucky' situations actually become 'lucky' and you WANT them to happen. You also have the option of forcing these situations to happen by intuitive solving. The more sets you know the more options you have. For example, consider the following 'unlucky' situation:

Scramble: M2 U2 M2 U' M' R' U' M' U R M2 U' R'



If you know Waterman Set 2, the solve becomes:
R U M2 R' U' M U R M U' [10] // Waterman Set 2
U2 M2 U2 M2 [5] // Permute midges
Total: 15 moves

If you only basic LMCF, the solve could be done like this:
R U M' U' R' U M U' M2 R' [10] // intuitive solve & setup
Now orient midges with special case:
U' R U' M' U2 M2 U' R' U [9] // orient midges
R2 M' U2 M U2 M [6] // permute midges
Total: 25 moves

But if you know the iDFR L5E set (intermediate LMCF), then you can transpose the solve quickly into the iDFR set as follows:
R U M' U' R' y2 [5] // intuitive set up to iDFR set
r F R U' M' U R' F' [8] // iDFR L5E algorithm
L2 M' U2 M2 U2 M' [6] // permute midges
Total: 19 moves

This shows that even without memorizing the Waterman sets, you gain great power with the iDFR set, and the other special case L6E sets. To summarize this section, the special case L6E sets are the difference between random fast times and consistent fast times. With the extended sets, every solve is a fast solve because there is no 'unlucky' situation at the end. With the 'basic' LMCF algorithm set, you can get really fast solves, but you will occasionally get 'unlucky' situations that require a lot of moves to resolve.

FAST SINGLES OPTIMIZATION

Since fast singles are a highlight of this method, if you are aiming for a PB single or a record single, there are certain rules you should follow:

1. Always create a CLL or EG2 face in the beginning. This allows for a CLL skip or EG2 'skip' (R2 F2 R2), 1 in 162 solves; it also allows for some of the really fast CLL algorithms.

2. If you are presented with an easy way to fully solve the L and R faces at the end of E2L, do it. This is a risky move since if the midges end up disoriented it can be a bad case (especially if 4 midges are all bad), but there is a 1 in 8 chance that the midges will all be oriented and you will skip the entire midge orientation step, resulting in a fast single. Another alternative is to try to always use the rDFR set and reflections since these include the 3 move midge orientation (1 in 8 chance).

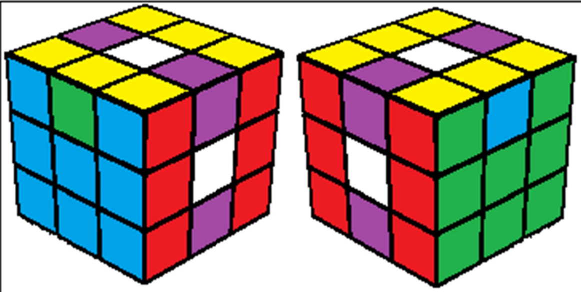
3. For maximum speed singles, only solve E2L pairs if they are already set up. On a lucky solve each pair will essentially set themselves up. This is the fastest way to solve. If you have to manually set up each pair you lose time. E2L triplets are the key to fast singles. These are worth setting up.

CHEAT SHEETS

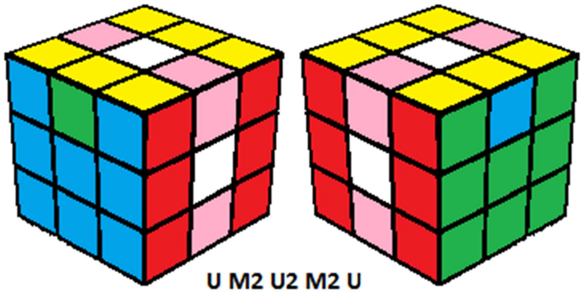
The next few pages are 'cheat sheets' that you can print out to quickly and easily memorize some of the algorithm sets.

LMCF L6E Special Cases - Sheet 1

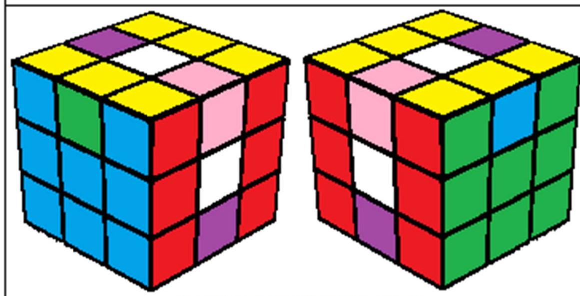
UL, UR swapped and oriented



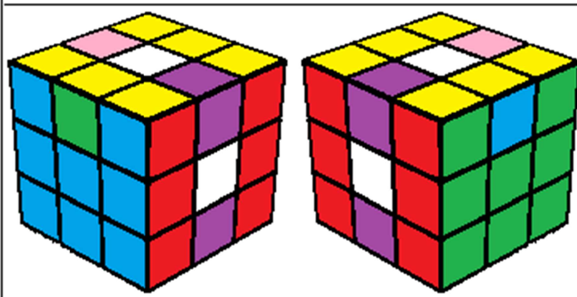
U M' U' M' U' M2 - U M U M' U M U M' U



U M2 U2 M2 U

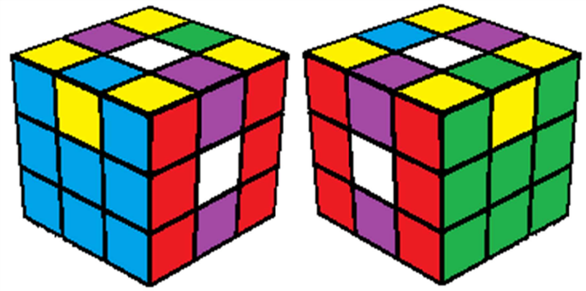


x' F U R U M2 U' R' U' F'

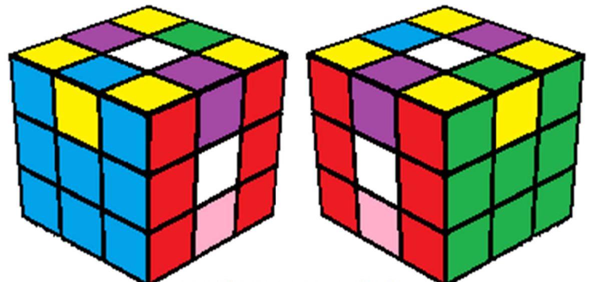


U' M U' M' U2 M' U M' U2 M2 U

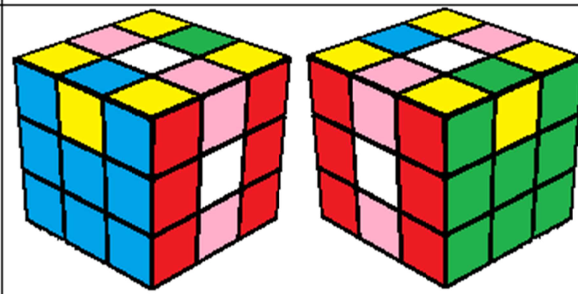
UL, UR solved and disoriented



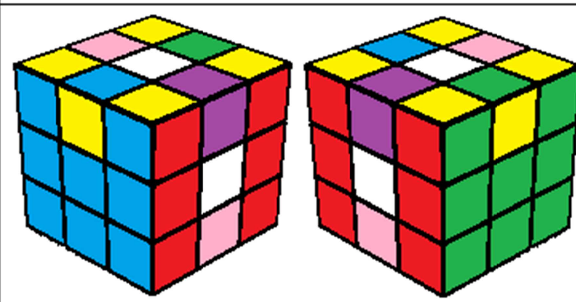
R' F R U M' U2 M2 U R' F' [R]



U M' U2 M U2 M' U'

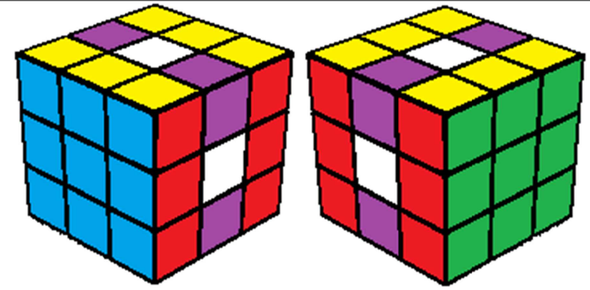


U M' U M' U M2 U M' U M' U'

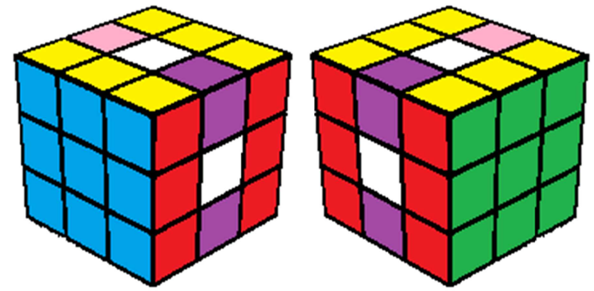


U M2 U [M' U M' U M' U] M U'

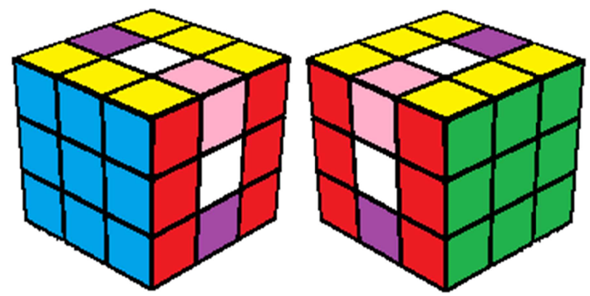
UL and UR both solved



U r' U M U' M2 R U' R' U M U' [12]
U' M U2 M' U' M' r' F R U' M' U R' F' [14]
U' M U M U' M' U' M' U M U M' U M U' [15]

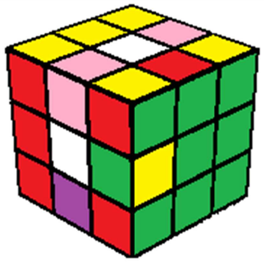
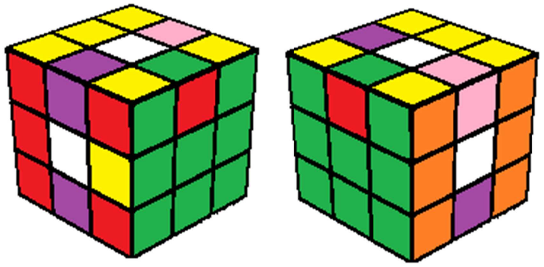
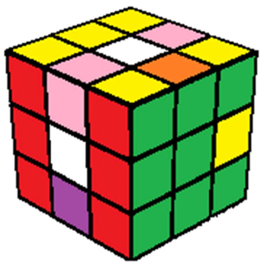
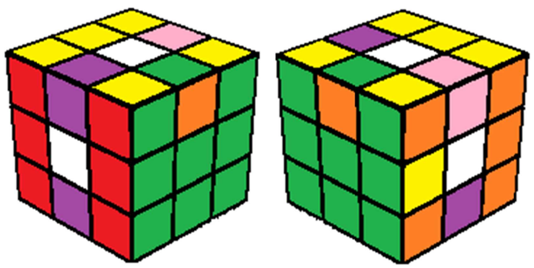
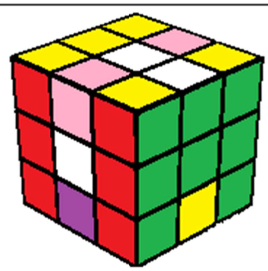
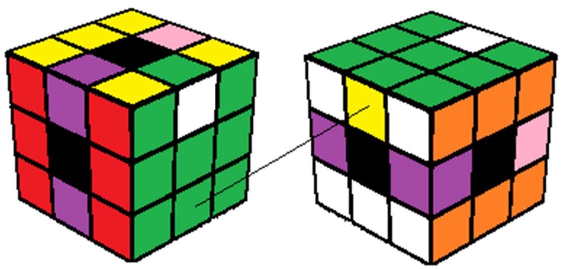


U M U M U2 M' U M' U'

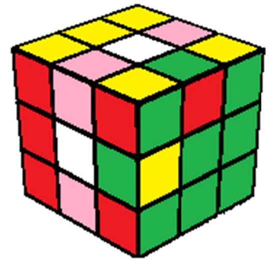
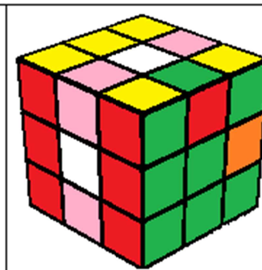
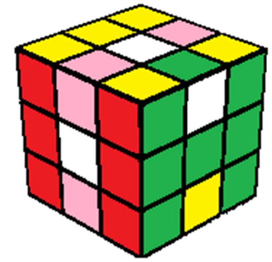
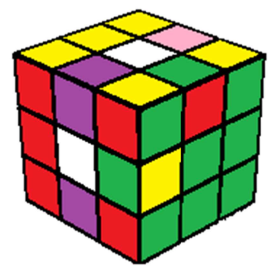
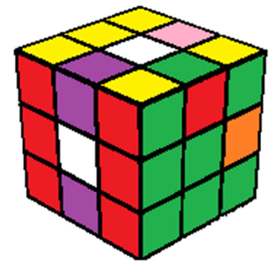
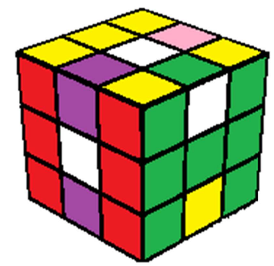
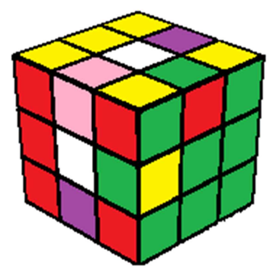
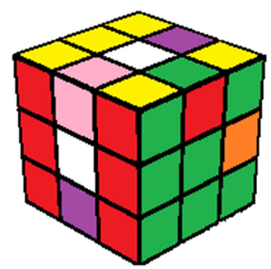
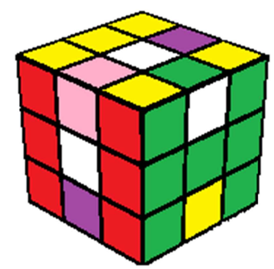
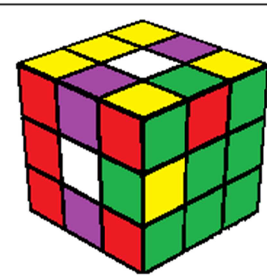
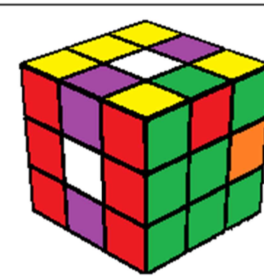
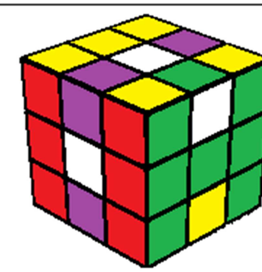


U' M U M U2 M' U M' U

Waterman Set 5: One oriented redge, one disoriented redge

One disoriented midge	Three disoriented midges
 U M' U' R U' M' U R' U M2 U'	 U M U' R U' M2 U r' U' M U
 U M' U' R' U' M' U R U M2 U'	 U M U' R' U' M2 U M R U' M U
 U M' U' R2 U' M' U R2 U M2 U'	 U M U' R2 U' M2 U r' R' U' M U

Waterman Set 6: Two disoriented redges interchanged

Redges at UR, FR	Redges at UR, BR	Redges at UR, DR
 U M U' R U' M2 U R' U M' U'	 U M U' R' U' M2 U R U M' U'	 U M U' R2 U' M2 U R2 U M' U'
 R U r' U' M2 U r U'	 U r' U' M2 U r U'	 U r' R' U' M2 U r R U'
 U M U2 R U' M2 U R' U2 M' U'	 U M U2 R' U' M2 U R U2 M' U'	 U M U2 R2 U' M2 U R2 U2 M' U'
 R U2 R' U M' U' r U' R'U M'U'R U'	 U2 R' U M' U' r U'R'UM'U'RU'	 U2R2UM'U'M'R2U'R2UM'U'R2 U'

CLL and EG1 algorithms Optimized for 3x3 LMCF method

by Eric Fattah (Feb 2017)

(First algorithm is CLL, second algorithm is EG1)

SUNE

CLL: R U R' U R U2 R'
EG1: z' R' U' R U2 B2 l U'

U' R' F R2 F' R U2 R' U' R2
R U R' F2 U F R U R'

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

F R' F' R U2 R U2 R'
U' R U' R' F2 U2 F U' F2

R' F2 R U2 R U' R' F
R' F R2 U' R' U R U' R' F

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

ANTI-SUNE

R' U' R U' R' U2 R
U' B U' r2 F2 U' F

R U2 R' F R' F' R U' R U' R'
R U' F2 R U2 R U' F

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

x' U' R U R' F2 R' F2 R
R U2 R' y' R2 U R' U2 R'

l U2 R' F2 R' F R U'
x' R' F D R D F2 l'F U' F2

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

L CASES

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

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

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

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

U R' F2 R U y' R2 U R U' R2
R U' R' y' R' U2 R' U R2

R U2 R' U' y' R2 U' R' U R2
U L' U L y' R U2 R U' R2

T CASES

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

U F U' R U2 R' U' F2 R U R'
L' U L2 F' L' U' L' U2 L

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

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

U' R' U R' U2 R U2 R' U R2 U' R'
R2 B2 U' R' U' R U' R' U R'

U' F R U R' U' R U' R' U' R U R' F'
L' U' L2 x U L' U' L U L'

U CASES

U' R2 F2 R U R' F U' R U R2
x B2 U' R' U R U' F R U R U'

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

U' R2 U R' U' R2 D' R' F2 R
U' R' F R2 U' R' U y' R U R'

U' R U2 R' U l U2 R U' R' U2 R
x B U B' R' U2 R U' B2 U'

F R U' R' U' F2 U' R U R' D R
U F' U2 R U2 R' U2 F

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

PI CASES

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

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

F U' R' F2 R U' F2 R U2 R'
F R U' R' F R U2 R' U F'

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

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

U' R U' R U' R' U R' F R2 F'
U' F2 R U R' U2 F2 R U' R'

H CASES

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

U' R U R' U R U R' F R' F' R
U' R' U' R' F2 x B U' R U'

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

F (R U R' U')x3 F'
R2 U R U' R F2 U' R U R

Permutation Algs:

R2 F2 R' U' R F2 R' U R'
R2 F2 R D R' F2 R D' R

R2 U' B2 U2 R2 U' R2

R2 F2 R2

R U R F2 R' U R'
R' D R' F2 R D' R

R U' R' U' F2 U' R U R' U F2

EXAMPLE SOLVES WITH INTERMEDIATE (201 ALGORITHM) SET

This section will go over some example solves.

Scramble: B2 L2 F2 L F B U R' U2 F' L2 B D2 L2 U2 F' R2 U2 F2 D'
y' x2 U R' U2 // yellow face and AUF
R U' R' F' U' R U R' U' F // EG1
D l R // AUF and set up
F U M U' F' // E2L pair
L d' M U M' // E2L triplet
y R // set up
D' M U M' U' D // E2L triplet
x M // setup L6E
M' U' M U' M' U' M U' // solve UL and orient midges DFL set
M' U2 M2 U2 M' // permute midges
Total: 45 STM

Scramble: D' F2 U' L2 U' R2 U F2 U R2 D2 B' U2 F U' R' D L' D B' U R
x' U r F r U' R' y2 // face [6]
R U R' F2 U F R U R' // EG1 Sune [9]
z R2 U M' U' // E2L pair [4]
R2 L2 U M' U2 M2 U // E2L pair [7]
x M2 U' M U2 M' U' // E2L pair [6]
R2 U M U2 M U R' // solve last edge while orienting midges [7]
U2 M' U2 M' // permute midges [4]
Total: 43 STM

Scramble: B2 U2 B2 D' R2 B2 D R2 U2 B2 D' L U B F' L' D2 R D U
x' y' U2 R' D U // create green face and AUF in prep for EG1
F' U R U' R' U F R U R' // EG1 Sune case
U2 M U2 M' // solve blue-white edge piece on U layer
x y r U' M' U // rotate, solve green-yellow edge piece and displace blue-yellow edge piece
x L' U M' U' // solve blue-yellow edge and displace blue-orange
r U M U' // E2L pair, blue-orange and green-white
L' R2 U' M' U // solve last E2L pair and gamble on a midge orientation skip
U2 M' U2 // midge orientation skip, permute midges
Total: 38 STM

Scramble: R2 U' L2 F2 L2 U2 F2 D' L2 B' U2 L U L D B2 R' D
z x' R U' R' M2 // Build blue face, solve blue center
y U' B U' r2 F2 U' F // EG1 Anti-Sune, diagonal antimatch case
z B U M' U' B' // solve 1st E2L pair (one blue edge, one green edge)
R2 x L' U' M U // solve 2nd E2L pair (one blue edge, one green edge)
x2 M U M' U' M2 // solve final E2L edge piece
M' U M' U M' U M' U // orient last 5 edges
U2 M2 U2 M' // permute last 4 edges
Total: 38 STM

Scramble: L2 D F2 L2 D2 F2 U' F2 U' L2 B2 F D2 R' B R' D2 L D2 F U'
x2 B' R U R U // solve blue face
R U2 R' y' R2 U R' U2 R' // EG1 anti sune
u D2 M // solve blue center and red blue edge

U' M U' M' // solve yellow green edge
z' M B U M' U' M B' // rotate solve E2L triplet
r2 L U M2 U' l // solve last single E2L edge and set up L6E iDFL set
U M' U' M' U' M' U2 M U' R // iDFL set, solve last ledge/redge + orient midges
U2 M2 U2 // permute midges
Total: 46 STM

Scramble: D B2 U2 B2 U L2 F2 R2 B2 D' F2 R U F' U2 F2 D' B' R' B2 U'
z x U2 R' U R' D' U' // blue face
x' R' F D R D F2 l' F U' F2 // EG1 anti-sune
z' M U' M' U2 M U' // E2L pair
x L2 U M U2 M2 U // E2L pair
R L U M' U2 M2 U // E2L pair
l M // set up L6E
U M' U' M U M U' // BDL set, solve last ledge + orient midges
U2 M' U2 // permute midges
Total: 47 STM

Scramble: D2 L2 B2 L2 U' F2 L2 F2 L2 D' L2 R' U2 F D F2 R' B D' B2
x2 y F' R U2 R' u M // solve face
R U R' U R U2 R' // CLL sune
z x' R2 U M U2 M2 U // E2L pair
x M2 U M U2 M2 U // E2L pair
x2 U' M U // E2L pair
r' L U M U' M // set up L6E BDR set
U' M' U M U' M U // BDR solve redge + orient midges
M' U2 M U2 // permute midges
Total: 45 STM

Scramble: R' D2 L2 F2 D' U2 L2 B2 F2 D' L2 F R' U L' B2 R F D' R
y2 x' R' U2 F2 U D2 // face
R2 U' B2 U2 R2 U' R2 // corner permutation
z R M U' M U // 1st edge
l2 y M' U M d' // triplet
x l' L' U' M' U2 M2 U' // pair
x2 U' M' U // last edge
U M U M U M' U2 M' U' // L5E BDL set
E2 M E2 M // permute midges
Total: 45 STM

Scramble: U F' R2 B U2 B2 F D2 B R2 U2 L' D R2 B2 F2 L U' R'
y2 x F R U2 R D2 U // face
L' U' L2 x U L' U' L U L' // EG1 T case
x' U2 M U' M' // solve edge on top face
M' D' M // solve edge on bottom face
z l L U' M' U // solve edge
L' U M U' // last pair
L' M // setup
U2 M U' M' U' M U' M' U // L5E BDL set
E2 M' E2 M' // permute midges
Total 46 STM

Scramble: L D2 L2 F2 R' D2 B2 R2 F2 L D L D L' B' F2 L' F' U' R'

x' U' R' U R' y' // face
R' U R2 U' R2 U' F R2 U' R' // EG1 L case
z' r' M2 U' M U // 1st pair
x' M' U' M U // 2nd pair
x' l B U' M' U M B' // 3rd pair
L' M2 U' M U M // set up L5E
M U M' U M U M' U // L5E BDL set
M2 // finish
Total 45 STM

Scramble: F2 L2 F' R2 B D2 R2 U2 D' L2 B2 R D2 R2 U' B'
x2
b u' b U2 // face
R U' R' F' U' R U R' U' F // EG1 anti-sune
y M U2 M' // red blue edge
M' D2 M // yellow green edge on bottom
z' B U' M' U B' // pair
L2 U M2 U' // green orange edge
r' U' M2 U M2 D2 // set up L5E
r' F R U' M' U R' F' R // L5E
U2 M' E2 M // permute midges
Total 48 STM

Scramble: U2 F2 U2 R B2 F2 L B2 F2 U2 L2 B U B2 U2 F U2 L' F' U R'
y U2 R F' U2 F y // yellow face
R' F2 R U y' R2 U R U' R2 // CLL
U D x // centers
R2 U M U' // 1st E2L pair
x R2 y' M' U M d' // E2L triplet
x r U M U' M' // solve last UL edge
M U M U' M' U' M U' M2 U2 R' // L6E
Total 41 STM

Scramble: L2 D' R2 U F2 D R2 B2 U F2 R2 L B' F' L' B' D B D2 U'
y x' L' U L S' // green face
U // AUF
R2 U R' U' R F2 U' R U R // EG1 H case
x y2 M' U' M d // E2L pair
R M' U M' U' R2 // E2L pair
M U M' U2 M U2 M' U' // LMCF L6E DD case
r' U2 M U2 M2 // permute midges
Total 38 STM

Scramble: U' R2 B2 R2 D' R2 D2 U R2 B2 F2 L' U' R F2 D2 L D F' U L2
y' U2 R2 U' R2 U2 // blue face
(R U R' F) (R U R' F) (U' R U' R') // EG1 T-case
E' M' z' x' // solve centers and rotate
L2 M U M U2 M' U // E2L pair
x M2 U M' U' L' U' M U // last edges
E2 M E2 // permute midges
Total 37 STM

Scramble: U' R2 U L2 R2 B2 D2 U' R2 B2 L2 R' D2 R B F' U B R F D

z' x2 R' U' r' U2 // blue face
U' R2 U R' U' R2 D' R' F2 R //CLL
D M' D M // blue-yellow edge
z' R' M' U' M U2 M' U' // E2L pair
x' U M' U' // blue orange edge
M2 M' U M' U' M U' M' U // L5E orient
E2 M E2 R // permute midges
Total 41 STM

EXAMPLE SOLVES WITH WATERMAN L6E

Scramble: U F2 D' L B R2 F2 D L2 B2 U2 B' R2 F2 U2 B' L2 U2 B2 L'
y z' x' L' u l U // yellow face
U F U' R U2 R' U' F2 R U R' // CLL T case
U' E2 l' M' U' M U // 1st pair
x' M D' M U M' U' D // triplet
x' M2 U' M2 U // solve green orange edge
M R' U' R U' M U' M U M U r' U // waterman L6E
r2 U2 M U2 M' // permute midges
Total: 51 moves STM

Scramble: F2 U2 F' D2 R2 F R2 F U2 F U L D' F U2 F2 L U F L'
x z2 x' R2 F R // yellow face
R' F R2 U' R' F R U R' F' //EG1 h case
u' M2 L U' M' U // pair
x R U M U' // single edge
x B U' M' U B' // pair
r' // setup for Waterman L6E
L' U' M' U' M' U2 l U' M U // Waterman L6E Set 1 reflection
l' U2 M U2 M2 // permute midges
Total 44 STM

Scramble: L2 U F' D R' B' L D' R D2 B U2 B' D2 F2 D2 L2 B' U2 F
y' z' x' F' R' U2 F y' // red face
U F U' R' F R U' F2 R U R' // EG1 Pi case
D E2 y2 // center
x2 R M // set up
U' r U' M U R' U // triplet
x2 M U' M U R U' M' U // pair
M2 L' U L U2 M U' M2 U' M L' U' // waterman L6E set 2 reflection
r' U2 M U2 M' // permute midges
Total 51 STM

Scramble: F' L2 U2 B' U2 F U2 F R2 F2 L D F2 R D2 U' B F'
z' y' u U l' // blue face on D
U R U' R' U R U' R' U F R U' R' // EG1 Sune
z x U M' U' // 1st edge
L2 M' U M' U2 M U // 1st pair
x M U' M' U2 M2 U' // 2nd pair
x U M U' R U' M2 U R' U M' U M2 U2 M2 // Waterman L6E (Set 6)
Total 46 STM

Scramble: B2 L2 U' B2 F2 U L2 D' L2 B2 U2 L R B L' R F R' D' R
z2 U R2 U l U' R' // green face and CLL skip
M U2 M' // solve blue-red edge
D' M' D2 M // solve green-orange edge on D face
z x' L' U' M2 U // E2L pair
x2 R U' M' U M R2 // solve green-white edge and set up Waterman L6E
R2 U2 R2 U' M' U R2 U2 // Waterman
E2 M E2 L2 // permute midges
Total 34 STM

Scramble: B2 L2 U' B2 F2 U L2 D' L2 B2 U2 L R B L' R F R' D' R
z2 U R2 U l U' R' // green face and CLL skip
M U2 M' // solve blue-red edge
D' M' D2 M // solve green-orange edge on D face
z x' L' U' M2 U // E2L pair
x2 r' U' M2 U r R // set up
M2 U' r' R' U M U' r2 U R2 // Waterman L6E Set 2 case 6C
Total 33 STM

Scramble: B' R' L' U R' D R2 F D B D2 B R2 B D2 F L2 D2 L2
y x2 u' F2 M' U2 // blue face
L' U' L2 x U L' U' L U L' // EG1 T case
x' U' D2 M2 U2 M2 // E2L pair
z x M' U M' U' // single edge
R' M' U M U' // blue edge
M2 // set up
M U' M' U' M' D' U' M' U' M D // Waterman Set 2
r U2 M U2 // Permute Midges
Total 43 STM

Appendix

Miscellaneous Algorithms & Corrections

[last revision March 18, 2017]

Alternative algorithms for L5E, with one of UL/UR solved, and the other one disoriented:

One edge disoriented DF, UR
U M U2 M U M' U M U2 M U [11]
One edge disoriented DF, UL
U' M U2 M U' M' U' M U2 M U' [11]

Other options:
U M' U2 M' U M U M' U2 M' U
U' M' U2 M' U' M U' M' U2 M' U'
M U M U2 M U M' U M U2 M U (12s)
U2 M' U M' U2 M' U M U M' U2 M' U' (13s)
R U' R' U' M' U2 M2 U' R U [10]
R U' R' U M2 U2 M U R U [10]
R U' R' U' M2 U2 M' U' R U [10]
R U' R' U' M' U2 M2 U' R U [10]

Transition Phase algorithms (solving one edge on the top layer):
M U' M2 U' M
M2 U' M U' M
F' R E2 R' F
U M U' M'
M' U M U'
M2 U2 M U M
M U2 M'

Improved orientation of 4 disoriented midges in M-slice: U' M U2 M' U' R' F R U' M' U R' F' [13]

3 disoriented edges on R, while all of L face is solved:
One disoriented midge:
f' U M2 U M' U M' U M' U M U' f [13] (bad edge at BD)
R U R' F R2 F' U' l' D' l D' l2 U F [14] (bad edge at UB)
Three disoriented midges:
R' U r' F U' r' U' F' r U' r U' F' U' [13]; three bad edges, good edge at UB

Solve 3 disoriented edges (UL, UR, FR) while orienting M-slice:
3 disoriented midges:
U M2 U2 M U r U' M2 U2 M' U' [11] (good midge at UF)
R U M U2 M2 U r' U' M' U2 M2 U' [12]
R U' M2 U2 M' U' r' U' M' U2 M2 U' [12]

U' R L2 M U' M' U' M' U' M' U' R D [13] (one bad edge at DF)