5-Style V1.2: Getting CFOP length solutions in Blindsolving

5-Style is a method that tries to solve a 3x3 cube in around 50-70 moves using commutators which is the go-to way to do a 3x3 blindfolded. This method can be used in 3BLD to reduce the time of execution which stands at ~12 seconds now. The main applicability of this method is for the MBLD since there is no restriction on the upper limit on the number of cubes so newer methods can continue building complexity to improve the MBLD score.

1. Context

As a kid, I have always been fascinated by the game of chess, and I play it for quite a while during high school. The thing that I used to find that separated the very high-class GM from a normal amateur chess player, was the amazing preparation the GM used to put to get his technique and repertoire correct. As an amateur player, I had faint level positional chess sense, and I was riding high on attacking chess and tactics. But this thought of how important chess preparation is always stayed in my mind.

I am always been perplexed by the Rubik's Cube, and I can say my perception keeps changing as I mature. At first, it used to be a great feat just to solve one side, upon which I stayed satisfied for years, then one day I learned how to solve it completely by looking up an online tutorial.

Doing the cube blindfolded was the next challenge which I took up, which took me another four years to get the hang of, and I did it by leapfrogging from the Old Pochmann method to using M2/R2 in my solves.
As the years went by, I slowly and steadily replaced each inefficient M2/R2 algorithm with a newer and faster 3-style algorithm.

Abhijeet

Another major breakthrough in the cubing scene came with the US BLDers smashing the blindfold times, by making really fast 3-style algorithms. This was the turning point for me, as TPS was a thing I wasn't planning to invest on.

Motivation

I attended my first major competition at the Asian Championships in Beijing in 2016. It was an amazing experience, and the major takeaway I had from the tournament was the impact I got from three cubers, who seemed to be on just another level: Shivam Bansal, Kaijun Lin, and Gianfranco Huanqui.

Kaijun Lin had already inspired me to take up the Roux method as my main solving method, and he had shown how BLD times can be made such low and consistent with practice and focus.

Gianfranco Huanqui is a revolutionary BLDer, who has made new kinds of fingertricks and made many new algs which are novel and fluently executable.
On the final day of the Asians, Gianfranco Huanqui did over 300 3BLD solves in one day at the venue. I had lost the count of the sub-20s, sub-18s he got and it was spectacular to watch him practice. In every solve, he looked at a point where he thought he could have improved, and continued self-learning in this way.

I also remember Shivam Bansal saying a mind-blowing fact after the prize distribution that, our mind is so powerful that we can store petabytes worth of information in it which is even more than a supercomputer or a cluster of computers can ever harness. By having such brain power the limits of MBLD can never be reached, he said.

After the competition, I headed back to Chennai in India, feeling more driven to create something new.
The next month (Nov 2016), I finally thought of taking the plunge into making a new method that I had always thought of but never did. I had decided to list out and memorize all the 5 cycle algorithms for 3x3, for both corners and edges, also get some 4 cycle comms which can come handy in finishing off edges in most of the cases and new parity algs. I wanted to make a memory element for each letter quad which could be retrieved doubly fast than 2 letter pairs, and I wanted a 12ish movecount fingertrick-able 5-cycle algorithm that could solve the case in the fastest time and with very less finger movement.

Epiphany

I was attending Shaastra Open 2014, my second ever WCA competition. I was 18 years old at that time and had just finished a 4/8 MBLD attempt which felt quite satisfying. The competition went well, and I came second in 3BLD with a time of 2: 06, behind Kabyanil Talukdar who got a 1:20. After the prize distribution ceremony, Arunachaleswarar, an overzealous skewber, who was doing blindfold Skewb solves by insane tracking, saw me doing M U M' U' on a 3x3. I showed him that these 4 moves are so efficient that they cycle 5 pieces without affecting the rest of the pieces. He added up to me saying that, you should make a whole system out of this idea. I shrugged it off saying its just too hard as there are many cases, running into over million unique cases.

The same day, earlier I talked to the MBLD winner Vikram Mada who did 6/6 using only single letter memorization (not even letter pairs) and discussed with him conveying how I wish to go beyond letter pairs and go to letter quads. I quickly calculated the number and said a quarter million cases. He said that this just looks impossible , saying that he was already having a tough time transitioning to 480 letter pairs and there I was talking about an algset that runs into a hundred thousand cases.

Rouxinspiration

I have been using the Roux method since the year 2016. The step in a Roux that fascinates even a normal cube solver is the LSE part or the last 6 edges.

Most of the times we try and solve the LSE, we focus on getting the arrow edge orientation shape which will make all the edges "good", by performing M/M', U/U', M/M' moves.

One night when I was going only LSE solves, I realized that the speedsolving approach to the LSE is quite rudimentary, and even with EOLR, UL/UR prediction and pinkie pie algsets, we totally avoid the concept of commutators in the solving process.

2. Why try this method?

I feel similar to a prepared Chess player, or a prepared Go player before I do a 3BLD solve. Rather than a nervy person spamming the Y-perm, and locking up and getting frustrated about the lockups, you will feel composed and at ease during the solve.
3BLD solve would be similar to counting up to the number 5 in mental effort, and if you get comfortable in it there will be, on an average of just five letter quads in a particular scramble. You will come off the beginner tag that every CFOP user or M2/OP user gets when he/she stops learning algs, after they learn OLL, PLL, M2 and just focus on fingertricks, and not newer algs.

Till now I have gotten many easy scrambles officially. I once got a good 10/4 scramble in a competition in 2015. I reached a bottleneck in my improvement after that, which I could only improve on by drilling 3-style algorithms and getting all the algorithms in the algset sub-1 seconds. In hindsight, I do not want to reach another bottleneck, so I thought of developing this method.

Disclaimer: Please delve into this method only if you love speedcubing, and only if 3BLD/MBLD is your main event, otherwise this is not worth investing your time into.

Why I used the DF buffer for generating all 5 cycles?

I formed this idea in 2014 and started developing only in 2016 ie 2 years later. The UF buffer craze for 3BLD was quite less, and UF buffer was only perceived as a buffer for TurBo method which was an alternative to M2 back in those days. I will apologize for the UF fans if they find reluctance while form 5 cycle algorithms from DF. But if you just see, the chances that both UF and DF pieces are involved in a 5 cycle (1-20x18x16x14/22x20x18x16=1-7/11=4/11=0.3636), than UF and DF both involved in a 3-cycle (1-20x18/22x20=1-9/11=2/11=0.1818). (18% more involvement of both UF and DF)

It is better if the debate of the buffer is left aside, and only the focus is on efficiency and fingertrickability.

If you do not find this argument convincing enough, just do a z2 rotation to make every DF 5-Style algorithm into a UF one, the transformation that will happen is:
L-R
U-D
F-F

How to make 5 cycles less daunting?

Learn how to grind algorithms in an efficient manner can help you get a ton of algorithms into your head.

The best way to tackle this humongous algset method is to consider only one alg at a time, learn that one alg in a day, and move on to another alg. The daily rhythm is the best strategy to get everything solidly sorted out in your mind.
I also wish to make a video series focussing on subcategories of the algset and how to make a huge memory map of algs in the head.

How hard is 5-Style compared to the well known hard algsets like ZBLL?

5-Style has ~12600 edge algorithms and ~60000 corner algorithms. For the corners, the [R U D] 3-style algorithms are already really fast, and the edge algorithms move count has been reduced by ~40%. 5-Style algset size is similar to $(\text{ZBLL})^2$ which is just plain crazy.

You have got to think from a different perspective and have a 'why not' attitude to make peace with 5-style.

5-style vs 3-style

5-Style is assumed to be the extension to the efficient and finger-tricky 3x3 blindfolded method of 3-style.

3-style is a really fast method. The current WR is less than 20 seconds using 3-style. 3-style for corners is already quite optimized considering fingertricks and regrips. 5-Style does not immediately triumph over 3-style when it comes to only corners, as we can combine two similar corner comms (if they are coming in succession), and get a very efficient and fast execution.

Cube explorer is basically weak in finding good 5 cycles for corners, and one of the reason can be due to bad orientation of one of the corners in the 4 corners that need to be cycled, the overall algorithm can be long and inefficient. To make 5-Style work on corners, some addendum kind of work has to be done, where 3-style is extended out in some likely and unlikely cases, and categorized, and scaled to 5 cycles fully.

5-Cycle Algorithm Count

Edges: 126720 (excluding the flipped edges and cycle break cases)

Corners: ~80,000, Number varies a lot according to the interpretation. There are many considerations you can either put or not put, in say 2-cycles, floating buffers, which can vary this number.

Abhijeet

5. Examples

1. R B2 D2 F2 R2 D R2 B2 L2 D2 U' B' L B' F' L' D' F L' R2 Fw Uw2

Edges: **FQOD MLUV IK** + parity
Corners : **PHNJ UALC (My lettering scheme)**

Reconstruction (5-cycle)
Edges:
F' R' D M' D' F D' M D F' R F // 12
M' F' E2 R' E R2 E R' F M // 22
[L : [S, r U r']] // 32
Corners:
D' L2 U R U' L' U R' L' D L U' L' // 45
F U F D' F' U' F' R' F2 D F2 D' R D // 59
U' r2 R' U L' U2 R U' R' U2 L R U' r2 U // 74 Parity (15)

Reconstruction (3-style)
Edges:
[U L F L' : [S', L2]] // 12
[L F : [L' S' L, F]] // 24
[R' D R : [E', R2]] // 34
[R2 U' : M' U2 M' U2] // 42
[L : [S, r U r']] // 52
Corners:
F D' L U L' D L U' L' F' // 62
U2 L' U' R U L U' R' U' // 71
z x' : [R U2 R' , D2] // 79
D' R U' L2 U R' U' L2 U D // 89
U' r2 R' U L' U2 R U' R' U2 L R U' r2 U // 104 Parity

Difference in move count: 30 moves
Video link:

2. D R2 D2 R' U2 D' B U2 L' U' R2 U' R2 B2 R2 B2 R2 D' L2 F' Rw2 Uw'

Edges: **DNBP VGTK IBAE**
Corners: **JPDU SFNG BC**

Reconstruction (5-cycle)
Edges:
F E' F E' F M F M' F' E2 F // 11
S2 L D' S' L S L' D L' S2 // 21
F' U F M' F M' F M F M' U' F M2 // 34
Corners:
F' M U' M' F D' S R S' U R E' M F' M' y' // 49
U' S L F' R' F L F' R F L S' U S L S' // 65

Reconstruction (3-style)
Edges:
[F : [R2, E]] // 6
[M2, U R U'] // 14
[S', R' F' R] // 22
[S : [U' M' U, L]] // 32
[U' L U, M2] // 40
U’ M2 U M’ U M’ U2 M U M U M U2 M U // 55
Corners:
U R2 U' L U R' U' L' U R' U' // 66
D R F L' F' R2 F L F' R D' // 77
[R U’ R’ , D] // 85
L' : [U' R U ,L2] // 95
[L’ U’ L U, R2] // 105

Difference in move count: 40 moves
Video link:

3. D' L2 R2 D2 B' L2 B2 F' U2 R2 U' F2 R B U2 L2 U2 L2 F' Rw Uw'

Edges: **FDAP VNLC ITRE** + parity
Corners: **OJQL BNSF EC**

Reconstruction (5-cycle)
Edges:
E R' U D S' U' S D' R E' // 10
F L' S' R S L E R' E' F' // 20
E S D U2 S2 R D S D' R' U y // 31
Corners:
F B E R E' B' U F' U' L' F D' F' D L F' // 47
U' F D' L F' D2 L' S' R F' D B U R' U' L z // 63
U’ r2 R’ U L’ U2 R U’ R’ U2 L R U’ r2 U // 78 Parity

Reconstruction (3-style)
Edges:
[U : [R' F' R, S]] // 10
[D' L : [E', L2]] // 18
[S' : [U M' U', R']] // 28
[D R' : [E', R2]] // 36
[L' : [U M2 U', L']] // 46
[U' D : [R F R', S']] // 58
Corners:
U R' F L' F' R F L F' U' // 68
F' U2 R D R' U R D' R' U F // 79
[L’ U’ L U, R2] // 89
F R' F R' F L' F2 R2 B U2 F' B' L // 102
U’ r2 R’ U L’ U2 R U’ R’ U2 L R U’ r2 U // 117 Parity

Difference in move count: 39 moves
Video link:

4. F2 L2 D2 R2 F2 R2 B2 U L2 F2 L B' R' B D' B2 L' F2 R' D Fw'

Edges: **BAKH TORG VMJT**
Corners: **RJST HFSC**

Reconstruction (5-cycle)
Edges:
F' U' M D' L E' L' U S U L y' // 11
M' U L' E F' E' F L U' M // 21
L F R S L' E' L E S' R' F' L' // 33
Corners:
U F U M2 U' S2 D' F' D' S2 U' M2 // 45
F D F' M' U' M F D' F M' U' M F2 // 58

Reconstruction (3-style)
Edges:
U' D R' U' R U R U R U' R' D' // 12
[U L U', M'] // 20
[R' S' R, F'] // 28
[U D : [R F R', S']] // 40
[R2 : [U' M2 U, R]] // 50
[S : [L', U' M' U]] // 60
Corners:
F D F' U F D2 F' U' F D F' // 71
R U' D' L U' L' D L U L' U R' // 83
U' D' L D L' U L D' L' D // 93
[U’ L2 U , R2] // 101

Difference in move count: 43 moves
Video link:

5. U2 L B R' U2 R' F L F2 R D' F2 D F2 D' F2 D L2 D' B2 D' Rw Uw

Edges: **BVCU GNLM AHIA TPQB** + parity
Corners: **LAST GC**

Reconstruction (5-cycle)
Edges:
M2 D M S' U' S' M' U S2 D' // 10
// Now there is a barrage of 4-cycles coming…
L U M' U' R' U' M U M x // 19
R' F' R S R' E' F E R S' // 29
U' L' E' L D' L E L' U D // 39
Corners:
F' L' F' R F' L F M' D R' F2 R D' r' // 53
[U' L' U , R2] // 61
U' r2 R' U L' U2 R U' R' U2 L R U' r2 U // 76 Parity

Reconstruction (3-style)
Edges:
[M2, U R2 U'] // 8
[U : M U2 M U2] // 14
[U M' U', R'] // 22
[R' D R' : [E', R2]] // 32
[M' : [L U' L' U, M']] // 44
[L', U M2 U'] // 52
[M2 U' L' : [E', L2]] // 62
[U' : [L' F L, S']] // 72
Corners:
R U M' F M U' L D' L' D R' D' // 84
R U' D' L U' L' D L U L' U R' // 96
[U' R U, R2] // 104
U' r2 R' U L' U2 R U' R' U2 L R U' r2 U // 119 Parity

Difference in move count: 43 moves
Video link:

6. How to implementing this method in your solves?

The number of unique edge cases in 5-cycle is a whopping 126,720 cases.

If you look into a normal scramble, not everything case is a 5-cycle, there are also a few 4-cycles which forms due to going into cycle breaks or edge targets finishing up during a trace [Form: ABAC]or going into the parity setup [Form: ABCA].

For corners, using 5 cycles is still questionable, as there are only 8 corners in a 3x3, and 7 targets in a normal setup, so it is best to solve them using [R U D] 3-style algs from the most optimal buffer UFR.

The occurrence of any given letter quad in a scramble is extremely sparse. And there is no way to deal with this sparsity than to be prepared for every case beforehand.

The chance that any letter quad comes up again in a solve in edge memo is 3/126720 = 1/42240 = **0.002367%**.
The chance that any letter quad comes up again in a solve in corner memo is 2/68040 = **0.002939%**.

This is just insane sparsity that a normal human being just cannot handle. You need a ton of patience developing each of the letter quad, which has a contribution of only 5/194760 = **0.002567%** in the entire picture!!!

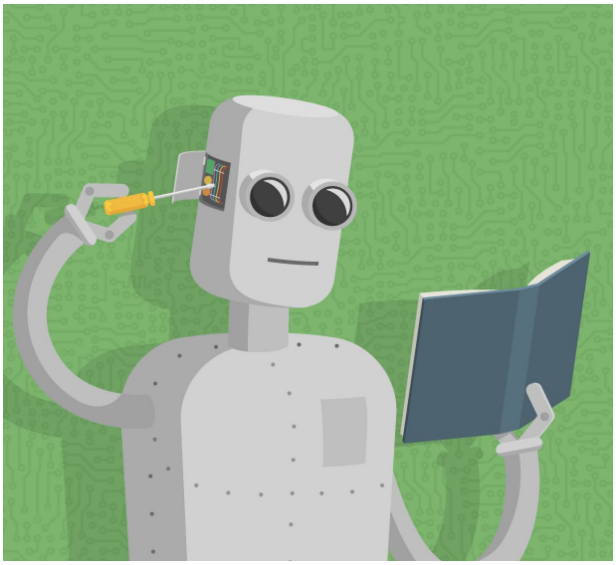
“How long before I master this method?”

There is no estimate on how long it will take, it all depends on the effort you can put in, and the focus you can garner up as you take a deep dive into this method.

The best way to implement this method right away is to always do some solves with it. And the way of learning you can do this is by deliberate learning (getting very analytical after each solve , on what all things you did and how you can improve on it).

7. My current motivation

Currently, I am doing Machine learning in huge sized image cube data. And generally, the number of classes that the model has to classify is ~10,000 classes.



Once I wondered, if I am training models using GPU to distinguish between hundreds of thousands of class, why would I not do the same for making memorization of a cube easier.

In a cube we can similarly create thousands of categories, each consisting of a unique 4 letters combination. The think-ahead becomes clearer in a solve because of this categorization.

Another source that spurred me to stick with 4 letter combination is the Indian art of percussion instrument of tabla. In tabla, there is a rich verbal language to represent the rhythmic sounds that the surface of both the drums makes. In that, there is a lot of divisions and basic counting mathematics, that makes it possible to have 16 beat or 10 beat cycle.

In tabla playing, there is a concept called the ‘rela’ which involves playing at very high speeds, with as much as 4-8 sounds in one count of the rhythmic cycle. This gave me the idea of having an impulse of 4 letters at once while memorization too.

Abhijeet

For those who did not understand the ‘tabla instrument’ musical example, I would compare the algorithm complexity of 5-style with the classical music instrument of the piano in the Western World.

A famous pianist tries to bring in a lot of abstract emotions in his/her playing.
There are no 21+53 or 480 or 500 set pieces that they have and the number combination of the notes they can produce goes into the hundreds of thousands.

8. How does the method work? (Types of cycles and swaps that emerge through this method)

We know that for a 3-cycle on a cube there are several types of commutators we can form out of it. And each one of them can be derived (I will not be deriving it here as it is a bit more mathematical)
They are classified as:

1. Pure Commutator
2. A9
3. Cycle Shifts
4. Columns
5. Per Specials
6. Orthogonals

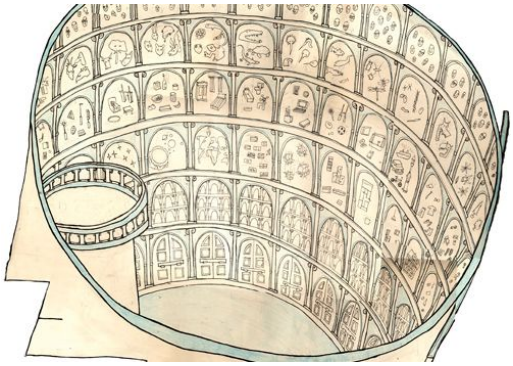
Ref: https://www.speedsolving.com/wiki/index.php/Beyer-Hardwick_Method
https://www.speedsolving.com/forum/threads/bh-tutorial.12268/

How effective is this method?

There is about 40% improvement in the move count, with little loss on the fingertrickability of the solve.
Since you cannot detect any insertion sequence in some 5-cycle alg quickly, it is best if you make triggers of batch 4 moves each and memorize each 4 move block using the Yo notation. Memorizing the algorithm via the Singmaster notation is quite cumbersome and difficult.

World record potential using this method: On a good 10/8 solve (9 algs), and assuming memo of 5 seconds, global times of 11-12 seconds is possible with execution times of 5-6 seconds.

9. 5-Style increasing the Multiple Blindfolded (MBLD) score



I previously made a video on how to memorize algorithms that do not have triggers and how to memorize these algorithms using Memory techniques.

The main motivation behind making 5-Style for me, is to make big MBLD attempts more seamless as this event is so information and memorization loaded, and also accuracy has to be spot on throughout the attempt.

This method is particularly very useful for the MBLD event since there is no restriction on the upper limit of the number of cubes, the complexity of memorization and execution can be increased to improve in MBLD event.

10. Future Scope

Get all the top BLDers to contribute to this mammoth alg database, and to make a lot of videos classifying these algorithms , and making new fingertricks some new types of triggers which will be seen.

There will have to be new techniques of remembering algorithms, without the involvement of cramming or muscle memory. There is a technique which I have developed which using Yo notation to memorize the algorithm in batches of 4 moves.

e.g., remembering algorithms via triggers will work in the case (oiag) : [U : [M,F]]
but not in the case (dula): F' U' F D' F' U R' D' R U R' D R U' F D, which have some 3 move insertion in its sequence but no set triggers or [A,B] inside it.

Analyzing the comparison 5-Style algorithm vs. two 3-style algorithms: To compare whether the tradeoff of 2 shorter 3-style algs is better or one 5-Style is better for all the cases. Eg, the hypothesis that for the 3x3 corners, the margin of movecount difference is less.

Making 5-Style algorithm to 4BLD wing algorithms (centers being preserved)
The length of each 5-cycle algorithm for wings on big cubes will shoot up in move count as many slice moves cannot be used in tandem, they will not be center safe.

If the 5-cycle is completely made and introduced, then it can be very useful , to use in FMC event. FMC solvers generally do efficient 2x2x3 block building, get a skeleton and do L5C without trying to look for some lucky insertion to reduce L5C.
If we already know the L5C algset, we can focus on block-building in the FMC attempt and do a 5 Corner insertion somewhere in between the solution using a 5-cycle algorithm (~move length 10-16).

Many new kinds of fingertricks coming out of 5-Style algs need to be analyzed. Because many of the move-sequences are different from the well known CFOP triggers and new ways of fingertricking them must be found.

11. How to scale this method and make it well verified and complete?

Abhijeet

The letter quads sometimes feel like feature engineering in old Machine learning terms, with a lot of toiling into making the data labeled and complete.

The best way to memorize a 3x3 scramble is to not use 2-letter or 4-letter, but do pattern abstract comprehension on the 3x3 (piecewise or sticker wise pattern making).

12. Creators

Hi , I am Abhijeet.

I am 23 and studying Machine Learning and Theoretical Physics. I have been speedcubing for over 6 years and I know how to solve a Rubik's Cube since 2008.

You can contact me at: abunickabhiyoyo@gmail.com

Yongqiang Peng
Yongqiang Peng is an active member on Chinese-cubing forum and has made several posts on higher order commutators on Chinese speedsolving forum.

His view of the 5-cycle is much different from me, and more driven towards the mathematics and the feasibility of 5-cycle rather than making an algset.

Contact: BBS Forum