5 Style V1.1

Abhijeet Gokar and Yongqiang Peng

5 Style is an extension to the very popular , efficient and finger-tricky 3x3 blindfolded method , 3 style.

1. Motivation

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 four years to conquer , and I did it by leapfrogging from the Old Pochmann method to using M2/R2 in my solves.
As years and WCA competitions went by , I slowly and steadily started replacing each inefficient M2 or R2 algorithm with a 3-style algorithm.
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 do not plan to invest on.

Genesis of this Idea

I attended my first major competition at Asian Championships in Beijing in 2016. It was a good experience , and the major takeaway I had from the tournament were 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 practise and focus.

Gianfranco Huanqui is a revolutionary BLDist , who has made new kinds of fingertricks , which made many new algs which were unthought of and fluently executable.
I 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 . So , by such brain power theoretically the limits of MBLD can never be reached.

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 practise. In every solve he looked at a point where he thought he could have improved , and continued self-learning in this way.

After the tournament , I headed back to Chennai in India, feeling more driven to create something new.
The next month , in November 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 memorise 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.

An Epiphany in 2014

I was attending Shaastra Open 2014 , my second ever WCA competition. I was a 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, Arunachaleshwar , an overzealous skewber , who liked the Skewb very much and did it BLD on Skewb by one looking , saw me doing M U M’ U’ on a 3x3. I showed him that this 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 memorisation (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 , stating that he was having a tough time transitioning to 480 letter pairs and there I was talking about an algset that runs into hundred thousand cases.

My recent Roux-derived Motivation

I have been using the Roux method since the year 2016. The step in a Roux solve 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 a M/M’ , U/U’ , M/M’.

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

2. Why you should even try this method?

You will feel like a prepared Chess player , or a prepared Go player , before you are doing 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 will start to feel like counting up to the number 5 , 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.

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.

3. Why I want this method to see the light of day?

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.

Why use the DF buffer for forming 5 cycles?

Note that I formed this idea on 2014 and started developing it in 2016 , so 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 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 5cycle (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). (So , 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.

How to make 5 style look less daunting?

Learn how to grind algorithms in an efficient manner , by which you can get maximum number 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.

Abhijeet

How harder is 5 style compared to the well known algset , ZBLL?

5 style has total of 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 moves count has been reduced by ~40%. 5 style is like $(ZBLL)^2$, which is just plain crazy .

You have think from a different perspective and have a ‘why not’ attitude to get started with 5 style.

4. Walkthroughing some cases

(mibo) : **F D R D' F' M F M' D R' D' F'**
Memo : vdje wmvn djew

(oiag) : **[U : [M,F]]**
Memo : amvn wb

(dula) : **F' U' F D' F' U R' D' R U R' D R U' F D**
Memo : wbve wake jakd bvd

5 style vs 3 style

3 style is a really fast method. The current WR is less than 20 seconds using 3 style.

5. Example Solves

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)
[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

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

Differences: 30 moves

Video:

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)
[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

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

Differences: 40 moves

Video:

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)
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

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)
[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

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

Differences: 39 moves

Video:

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)
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

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)
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

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

Differences: 43 moves

Video:

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):
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

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)
[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

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

Move Differences: 43 moves

Video:

Abhijeet

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 non - odd cyclic things like 2-cycles , floating buffers which can vary the number.

6. How to attack this algset and start implementing it in your solves?

The number of edges formed using 5 cycles are 22x20x18x16 = 126,720 cases.

If you look into a normal scramble , not everything is entangled in a 5 cycle , there are a sizeable amount of 4 cycle which form due to going into cycle breaks [Form: ABAC]or going into the parity setup [Form: ABCA].

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

The occurrence of any letter quad in a solve is very sparse. By sparse I mean extremely sparse. And there is no other way to deal with this sparsity , than to be prepared for every case like a ninja. 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?

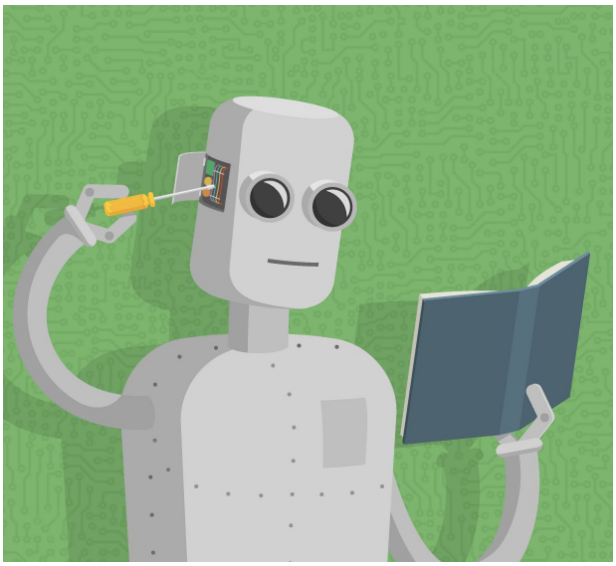
Abhijeet

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 , to digest all these algorithms.

The best way to inculcate the method is to always do some solves with it. And the way of learning you can do is deliberate learning , where you get 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 in is ~10,000 classes.



This makes me wonder, if I am training models using GPU to distinguish between hundreds of thousands of class, why would I not do the same for making memorisation of a cube easier.

In a cube by creating thousands of classes, each consisting of a unique 4 letters combination, the thinkahead becomes more clearer in a solve.

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 make. In that , there is 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 memorisation too.

Abhijeet

For those who did not understand this musical analogy , I would compare the algorithm complexity with the classical piano in the Western Music.

A famous pianist generally tries to bring in lot of abstract emotions while he/she tries to play a classical piece , and there are no 21+53 or 480 or 500 set pieces that they have. Generally the number combination of the notes they can produce goes into the hundreds of thousands.

8. How does the method work? (Types of Cycles that emerge in 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:
Pure Commutator
A9
Cycle Shifts
Columns
Per Specials
Orthogonals

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

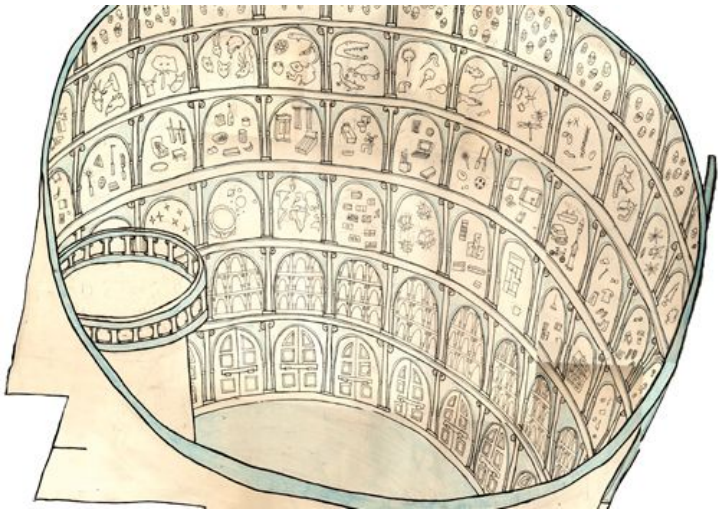
How effective is this method?

Abhijeet

There is about 40% improvement in the move count , with no or little loss on the fingertrickability of the solve.
Since , you cannot detect any insertion moves in some 5-cycle alg quickly , it is best if you make triggers of batch 4 moves each.

On a good 10/8 solve (9 algs) , and assuming memo of 5 seconds , a global average of ~11-12 seconds is feasible with execution times of 5-6 seconds.

9. Using Memory techniques to memorise these algorithms



I have made a video on how to memorise algorithms that do not have triggers.

10. Future Work to make this method easier and accessible

Try and get all the top BLDers to contribute to this mammoth alg database , and to make 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 methods of remembering algorithms , as cramming will not be an option due to poor retrieval.
Remembering algorithms via triggers will work in many cases like (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 inserts but no set triggers or [A,B] inside it.

There will have to be some people who can compare whether the tradeoff of 2 simpler 3 style algs is better or one 5 style is better for each of the cases. For corners mostly, there is less margin

, in movecount difference that makes analyzing the 5 style algorithm vs. two 3 style algorithm , worthwhile.

Adapting the 5 style algorithm to 4BLD wing algorithms will be another equally time consuming task , as most of the algs would be needing a modification to keep centers preserved on a 4x4 , to not mess the centers up. 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 , as they will be affect the centers.

If the 5-cycle is completely made , then it can be a very useful algset , to always use in FMC .
Top solvers generally do efficient 2x2x3 block building , and get to F2L-1 , and do ZBLS ZBLL , or reduce to L5C.
If we already know the L5C algset , we can focus on block building in that 1 hour period , and do the insertion of the 5 Corner cycle somewhere in between the solution.

Also , there will be many new kinds of fingertricks invented which will have to be analysed. Because many of the move sequence are without the well known CFOP triggers , we will need new way of finding ways to execute them optimally.

11. How to open source this method and make it grander?

The letter quads sometimes feel like feature engineering , with a lot of toiling going into making the data labelled and complete in all aspects.

The best way to memorise a 3x3 , IMO is to not use such labels (2-letter or 4-letter) , but try and do some sort of pattern recognition on the cube (piecewise or sticker wise).

12. Contact Us

I am Abhijeet. I am 23 years old and currently into Machine Learning and Theoretical Physics. I have been speedcubing for over 6 years now , and I know how to solve the Rubik's Cube since 2008. You can contact me at: abunickabhiyoyo@gmail.com

Yongqiang Peng
I met Yongqiang Peng on chinese cubing groups one day , and saw his post on Chinese speedsolving forum.

His view of the 5-style is much different , and he is more focussed on the mathematics and the feasibility of it.
His profile: BBS Forum

Keep Cubing!