Yuxuan’s PLL Algorithm List

This document contains a list of every PLL algorithm that Yuxuan uses for normal 3x3 solves with a short description of how she learned/remembers/recognizes the algorithm.

Note 1: All images and some algorithm names/categories are taken from the speedsolving wiki at https://www.speedsolving.com/wiki/index.php/PLL

Note 2: Most names are pretty arbitrary.

Note 3: The pictures will be presented in the angle you are supposed to hold the cube at to perform the algorithm.

Note 4: If there is more than one algorithm listed, the one that I use the most will be first. All other algorithms will be for the specific cases that are listed in the comments.

Note 5: All algorithms will also be broken down into triggers (a sequence of moves that is fast and easy to execute). Each case will have two cells in the algorithm section. The first cell will have the normal algorithm(s). The second cell will have the same algorithm(s) broken down into triggers.

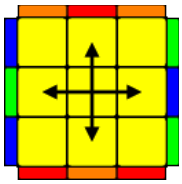
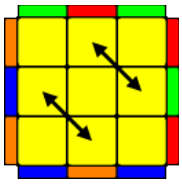
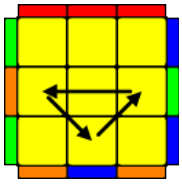
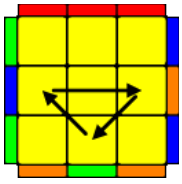
Note 6: For the most part, all algorithms will be written the way I execute them (some double turns will have a ‘ even though a 180 degree turn is the same whether the layer was turned clockwise or counter clockwise).

Note 7: There will be a separate document for 2 look PLL or 4 look last layer (4LLL), which is going to be a subset of these algorithms.

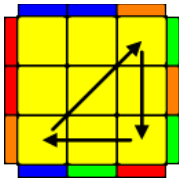
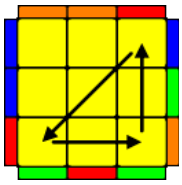
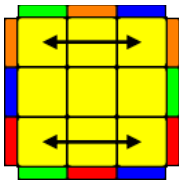
Note 8: My philosophy for learning algorithms was/is picking algorithms that are easy to learn (even if it might mean it is more moves/slower) so many of my algorithms build on each other.

Note 9: If there are any problems with the algorithms, contact Yuxuan.

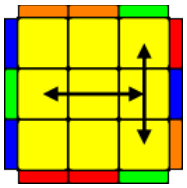
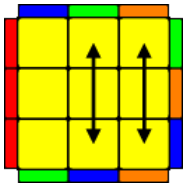
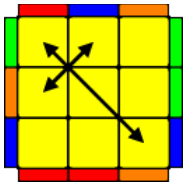
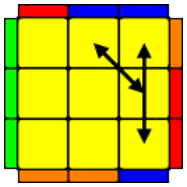
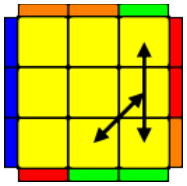
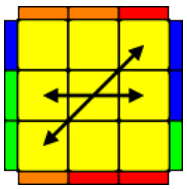
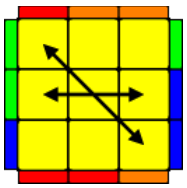
Edges only

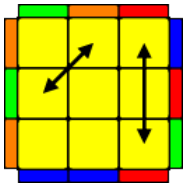
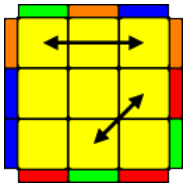
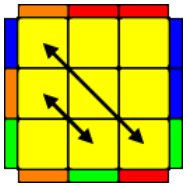
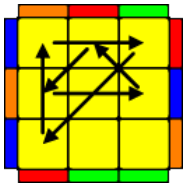
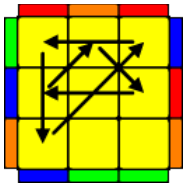
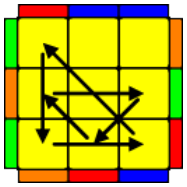
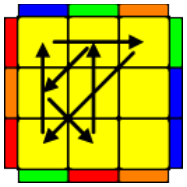
PLL	Algorithm(s)	Comments
	M2’ U’ M2’ U2’ M2’ U’ M2’ With triggers: (M2’ U’ M2’) U2’ (M2’ U’ M2’)	H perm
	M2 U’ M2’ U’ M’ U2’ M2’ U2’ M’ U2’ With triggers: (M2 U’) (M2’ U’) (M’ U2’) (M2’ U2’) (M’ U2’)	Z perm
	R U' R U R U R U' R' U' R2 With triggers: (R U' R U) (R U R U') (R' U' R2)	Ua perm Not opposite color on left Opposite color on right
	R2 U R U R' U' R' U' R' U R' With triggers: (R2 U) (R U R' U') (R' U' R' U) R'	Ub perm Opposite color on left Not opposite color on right Inverse of Ua perm

Corners Only

PLL	Algorithm(s)	Comments
	x’ R2 D2’ R’ U’ R D2 R’ U R’ With triggers: x’ R2 D2’ (R’ U’ R) D2 (R’ U R’)	Aa perm D2’s tend to switch between right hand and left hand a lot
	x' R U' R D2 R' U R D2 R2 With triggers: x' (R U' R) D2 (R' U R) D2 R2	Ab perm Inverse of Aa perm D2’s tend to switch between right hand and left hand a lot
	z U2’ R2 F (R U R' U')3 F' R2 U2’ With triggers: z (U2’ R2 F) (R U R' U')3 (F' R2 U2’)	E perm Lots of moves but super easy to learn Expanded: z U2 R2 F R U R' U' R U R' U' R U R' U' F' R2 U2

Other (arbitrary order)

PLL	Algorithm(s)	Comments
	R U R' U' R' F R2 U' R' U' R U R' F' With triggers: (R U R' U') (R' F) (R2 U' R' U') (R U R' F')	T perm Really just 2 OLL's together with a cancellation
	R' U' F' R U R' U' R' F R2 U' R' U' R U R' U R With triggers: (R' U' F') (R U R' U') (R' F) (R2 U' R' U') (R U R') (U R)	F perm R' U' F' + T perm + undo
	F R U' R' U' R U R' F' R U R' U' R' F R F' With triggers: F (R U' R' U') (R U R' F') (R U R' U') (R' F R F')	Y perm 2 OLL's (no cancellation)
	R' U2' R U R' z R2 U R' D R U' With triggers: (R' U2') (R U R') z (R2 U R') D (R U')	Ja perm
	1. R U R' F' R U R' U' R' F R2 U' R' U' 2. L R U2' R' U' R U2' L' U R' U' With triggers: 1. (R U R' F') (R U R' U') (R' F) (R2 U' R' U') 2. L (R U2' R' U' R U2') L' U R' U'	Jb perm 1. T perm with last trigger first 2. For OH (sometimes) L + inverse antisune + solve F2L pair
	R U R' U R U R' F' R U R' U' R' F R2 U' R' U2' R U' R' With triggers: (R U R' U) (R U R' F') (R U R' U') (R' F) (R2 U' R' U2') (R U' R')	Na perm R U R' U + Jb perm + undo
	L' U' L U R' U2 R U R' z R2 U R' D R U' R' U' R U With triggers: (L' U' L U) (R' U2) (R U R') z (R2 U R') D (R U') (R' U' R U)	Nb perm L' U' L U + Ja perm + undo

PLL	Algorithm(s)	Comments
	R U R’ F’ R U2’ R’ U2’ R’ F R U R U2’ R’ U’ With triggers: (R U R’ F’) (R U2’ R’ U2’) (R’ F R U) (R U2’ R’) U’	Ra perm
	R’ U2’ R U2’ R’ F R U R’ U’ R’ F’ R2 U’ With triggers: (R’ U2’) (R U2’) (R’ F) (R U R’ U’) (R’ F’) (R2 U’)	Rb perm
	R U’ L’ U R’ U’ R U’ L U R’ U2’ L’ U2’ L With “triggers”: (R U’ L’ U) (R’ U’ R U’) (L U R’ U2’) (L’ U2’ L)	V perm Almost niklas near the beginning Look at how the F2L blocks and the top layer colors move
	R2' u R' U R' U' R u' R2 y' R' U R With triggers: (R2' u R' U) (R' U' R u') R2 y' (R' U R)	Ga perm Look at how the top layer blocks change
	F' U' F R2 u R' U R U' R u' R2' With triggers: (F' U' F) (R2 u R' U) (R U’ R u') R2'	Gb perm Look at how the top layer blocks change Inverse of Ga perm
	R2 u' R U' R U R' u R2 B U’ B’ With triggers: (R2 u') (R U' R U) (R' u R2) (B U’ B’)	Gc perm Regrip for first 2 R’s B U’ B’ executed as U F’ U’
	R U R' y' R2 u' R U' R' U R' u R2 With triggers: (R U R') y' (R2 u' R U') (R' U R' u) R2	Gd perm Look at how the top layer blocks change Inverse of Gc perm