**3BLD OP Method (Old Pochmann)**

OP/OP is a 2-cycle BLD method invented by Stefan Pochmann. It solves one piece at a time, using PLL algorithms (T), (Ja) and (Jb) for edges as well as (Y) for corners with appropriate setup moves.

**Notation**

A goes to B = the correct position of sticker A is sticker B, in the sense of solved cube;

Old piece = piece shown before; New piece = piece not shown before; A: B = A B A'.

**Memorize Corners**

Stickers **A**, **E**, **R** of corner buffer UBL should never be memorized. Start first cycle with corner buffer **E**, start subsequent cycles with any sticker of any new unsolved corner, and stop cycle when old piece appears. Scramble: U2 B D' U' L2 R2 U2 R F R' L U' B' L' B2 F2 L' F2 B2 D' R2 U2 D2 R2 L. Do x2 y to make yellow on top and red on front.

First cycle: **E** goes to **N**, which goes to **E** (old), then cycle is over and we get **N**;

Next cycle: **D** goes to **O**, which goes to **G**, which goes to **X**, which goes to **J**, which goes to **I** (old), then cycle is over and we get **ND OG XJ I**;

Next cycle: **V** goes to **P** (old) (actually this corner twists clockwise), then cycle is over and we get **ND OG XJ IV P**. Since all the corner pieces are covered, corner memorization is over. The number of corner letters is odd.

**Memorize Edges**

Stickers **B**, **M** of edge buffer UR should never be memorized, and stop cycle when old piece appears. Start first cycle with edge buffer **B**, start subsequent cycles with any sticker of any new unsolved edge, and stop cycle when old piece appears. Notes: The number of corners and the number of edges should always be both even or both odd.

First cycle: **B** goes to **M** (old), then this cycle is over, and we get nothing;

Next cycle: **C** goes to **E**, which goes to **W**, whichgoes to **U**, which goes to **G**, which goes to **T**, which goes to **P**, which goes to **Q**, which goes to **C** (old), then cycle is over and we get **CE WU GT PQ C**;

Next cycle: **L** goes to **H**, which goes to **V**, which goes to **F** (old), then cycle is over and we get **CE WU GT PQ CL HV F**. Since all the edge pieces are covered, edge memorization is over.

**Solve Edges**

Setup moves of edge swap should never touch UR side (UFR, UR, UBR pieces). There are three ways to swap one edge and the buffer **B**. Pick the shortest setup move:

|  |  |  |
| --- | --- | --- |
|  |  |  |
| **(T)**  **= (R U R' U') (R' F R2 U' R' U')**  **(R U R' F')** | **(Jb)**  **= (R U R' F') (R U R' U')**  **(R' F R2 U' R' U')** | **(Ja)**  **= U' (R' U L' U2) (R U' R' U2 R) L** |
| Setup edge to swapping spot **D**, do (T) and undo setup moves | Setup edge to swapping spot **C**, do (Jb) and undo setup moves | Setup edge to swapping spot **A**, do (Ja) and undo setup moves |

**Edge Swap Algorithms**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **A** | **(Ja)** | **B** | **buffer** | **C** | **(Jb)** | **D** | **(T)** |
| **E** | **L d' L: (T)** | **F** | **d' L: (T)** | **G** | **D l': (Jb)** | **H** | **d L': (T)** |
| **I** | **l': (Ja)** | **J** | **d2 L: (T)** | **K** | **l': (Jb)** | **L** | **L': (T)** |
| **M** | **buffer** | **N** | **d L: (T)** | **O** | **D' l': (Jb)** | **P** | **d' L': (T)** |
| **Q** | **l: (Jb)** | **R** | **L: (T)** | **S** | **l: (Ja)** | **T** | **d2 L': (T)** |
| **U** | **l2: (Ja)** | **V** | **D2 L2: (T)** | **W** | **l2: (Jb)** | **X** | **L2: (T)** |

**Do Parity (if Exists)**

Parity exists iff the number of corner letters is odd. In this case, need to do the parity alg

(R U' R' U') (R U R D) (R' U' R D') (R' U2 R' U').

**Solve Corners**

Setup moves of corner swap should never touch UL, UB pieces. Here is the way to swap one corner and the buffer **E**:

|  |
| --- |
|  |
| **(Y) = (R U' R' U') (R U R' F') (R U R' U') (R' F R)** |
| Setup the corner to swapping spot **V**, do (Y) and undo setup moves |

**Corner Swap Algorithms**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **A** | **buffer** | **B** | **R2': (Y)** | **C** | **F2 D: (Y)** | **D** | **F2: (Y)** |
| **E** | **buffer** | **F** | **F' D: (Y)** | **G** | **F': (Y)** | **H** | **D' R: (Y)** |
| **I** | **F R': (Y)** | **J** | **R': (Y)** | **K** | **R' D': (Y)** | **L** | **F2 R': (Y)** |
| **M** | **F: (Y)** | **N** | **R' F: (Y)** | **O** | **R2' F: (Y)** | **P** | **F D: (Y)** |
| **Q** | **R D': (Y)** | **R** | **buffer** | **S** | **D F': (Y)** | **T** | **R: (Y)** |
| **U** | **D: (Y)** | **V** | **(Y)** | **W** | **D': (Y)** | **X** | **D2: (Y)** |

The final solution is:

x2 y // memo

// edges

(Jb) // C

L d' L: (T) // E

l2: (Jb) // W

l2: (Ja) // U

D l': (Jb) // G

d2 L': (T) // T

d' L': (T) // P

l: (Jb) // Q

(Jb) // C

L': (T) // L

d L': (T) // H

D2 L2: (T) // V

d' L: (T) // F

// parity

(parity)

// corners

R' F: (Y) // N

F2: (Y) // D

R2' F: (Y) // O

F': (Y) // G

D2: (Y) // X

R': (Y) // J

F R': (Y) // I

(Y) // V

F D: (Y) // P