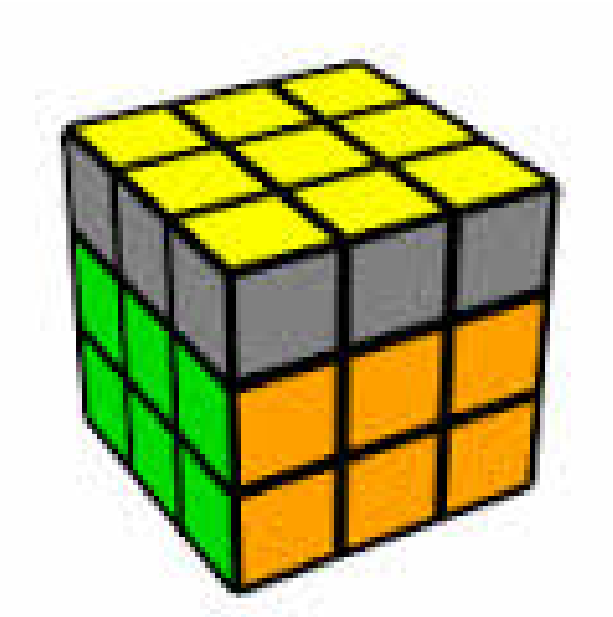
OLL ALGORITHMS

FOR PERSONAL USE ONLY

CONTENTS:

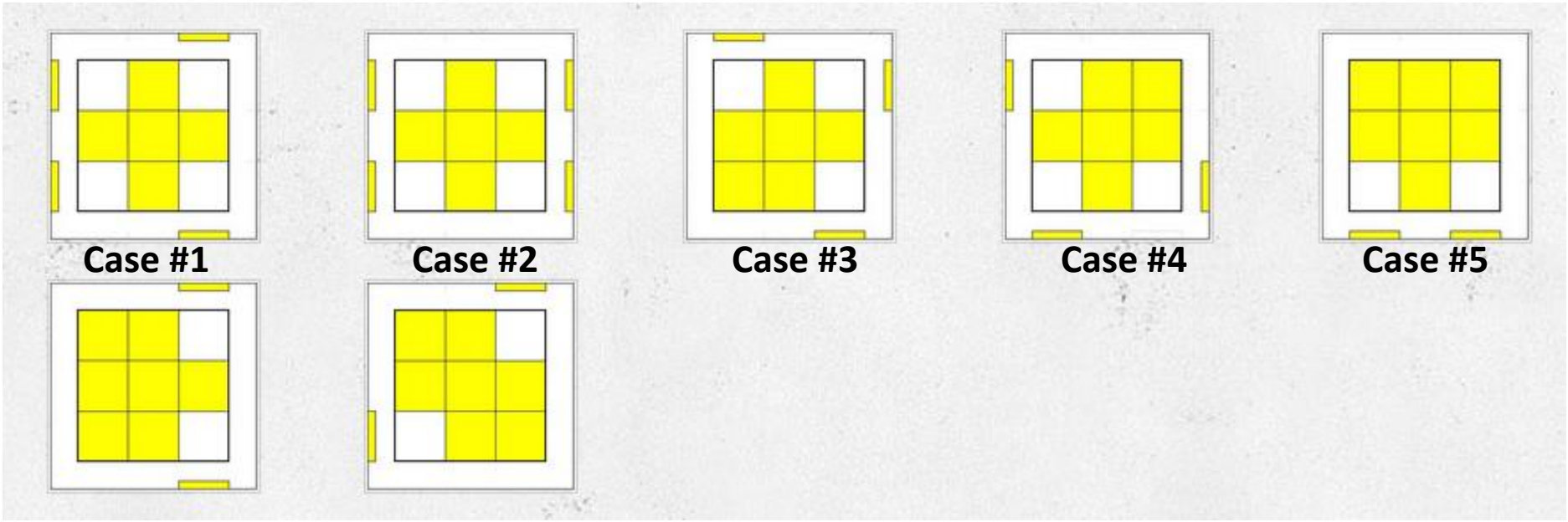


FS CUBING

Information

OLL is the third step in the CFOP method. OLL stands for Orienting Last Layer. In the beginner's method, you make the cross, position the cross' edges, position the corners and then finally permute the corners.This step orientates all of the pieces, so that the top of the cube is all one colour. It doesn't matter where the pieces are at this point, as we will be fixing that in PLL, Permuting Last Layer. OLL has 51- YES, 51_- different algorithms. I don't recommend learning them all at once, because you will probably forget them more quickly than learning them in stages. I recommend learning them at a rate of 1- 2 algorithms a day, or learning them in sections, like the ones displayed on this page. This will ensure that you learn them and they will stick in your brain. Learning full OLL takes a long time, so first of all, I would use 2 -Look OLL first, because that method gives you 7 of the 51 algorithms already, kind of giving you a head start in OLL. I would also recommend, when you come to it, learning 2- Look PLL, as there are also quite a few algorithms to learn in PLL. Before you start, I would learn full F2L first, so that you have already learnt 42 algorithms. You can then use your own method of remembering all of the 51 algorithms for this step. Every single one of the OLL algorithms will be displayed on this page, because you can't really learn it intuitively, like F2L

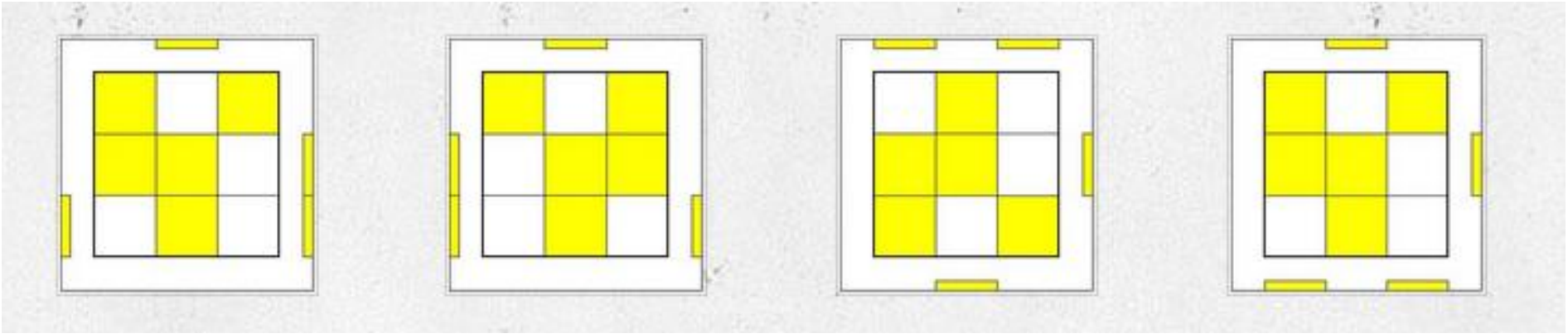
Algorithms- All Edges Flipped Correctly



Case #1- R, U2', R2', U', R2, U', R2', U2, R
Case #2- R, U, R', U, R, U', R', U, R, U2', R'
Case #3- R, U, R', U, R, U2', R'
Case #4- R, U2, R', U', R, U', R'
Case #5- R2', D, R', U2, R, D', R', U2, R'
Case #6- l', U', L, U, R, U', r', F
Case #7- l', U', L', U, R, U', L, U, (x')

FS CUBING

Algorithms- Awkward Shapes

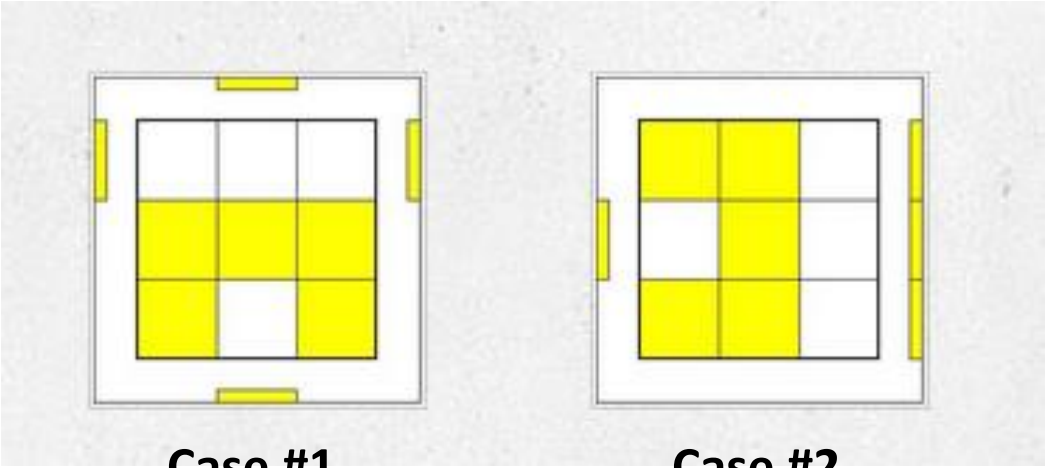


Case #1 **Case #2** **Case #3** **Case #4**

Case #1- B', R, B', R2', U, R, U, R', U', R, B2
Case #2- R2', U, R', B', R, U', R2', U, l, U, l'
Case #3- R, U, R', U, R, U2', R', F, R, U, R', U', F'
Case #4- R', U', R, U', R', U2, R, F, R, U, R', U', F'

FS CUBING

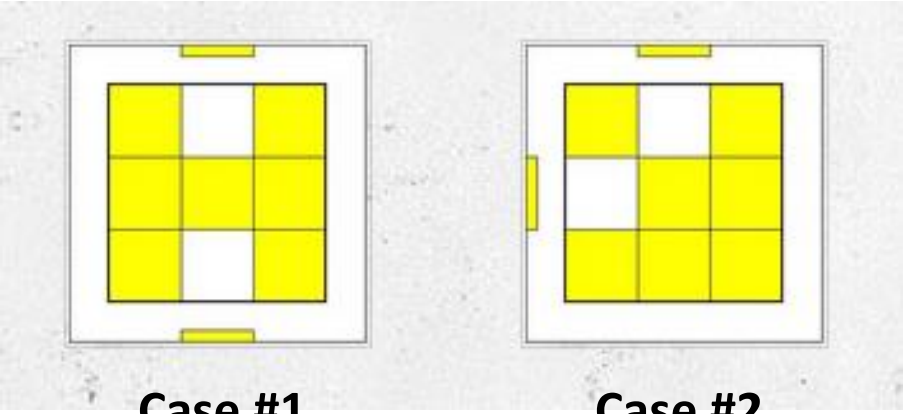
Algorithms- C Shapes



Case #1 **Case #2**

Case #1- R, U, R', U', (x), D', R', U, R, U', D, (x')
Case #2- R, U', l', U, l, F', U, R

Algorithms- Corners Correct, Edges Flipped

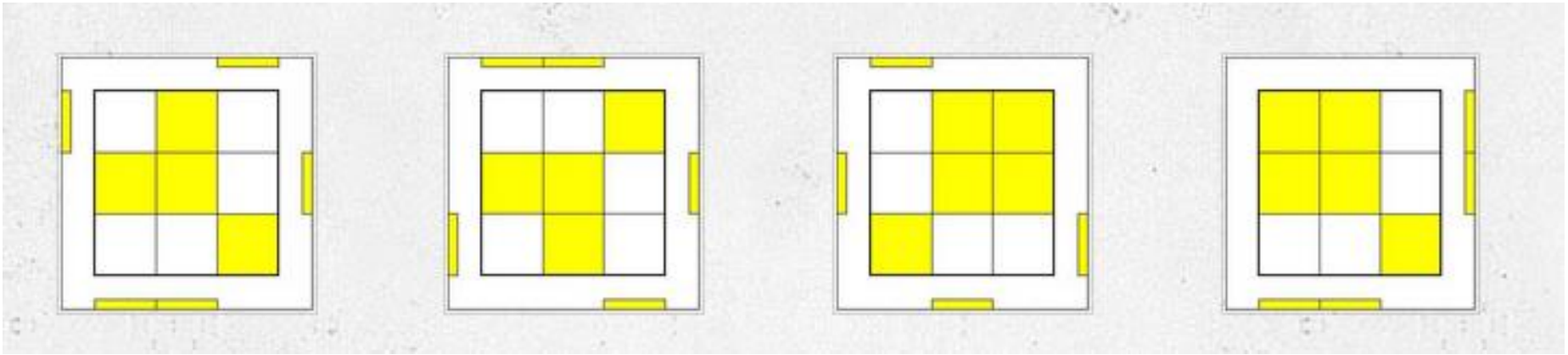


Case #1 **Case #2**

Case #1- R, U, R', U', r, R', U, R, U', r'
Case #2- r, R', U, R, r', U2, r, R', U, R, r'

FS CUBING

Algorithms- Fish Shapes

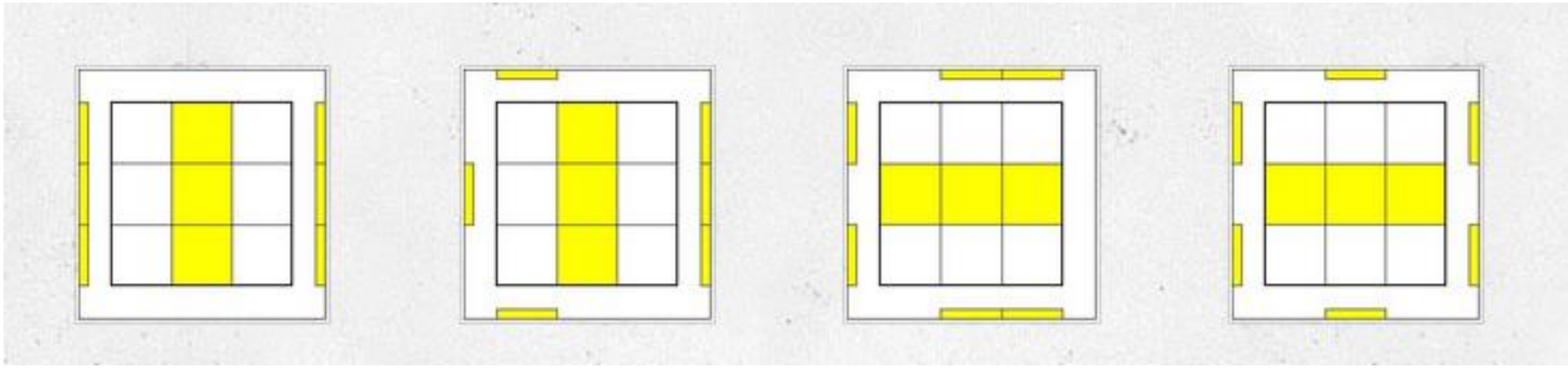


Case #1 **Case #2** **Case #3** **Case #4**

Case #1- R', U', R, (y', x') R, U', R', F, R, U, l'
Case #2- R, U, R', (x, z') R', U, R, B', R', U', l
Case #3- R', U2, l, R, U', R', U, l', U2',R
Case #4- F, R, U', R', U', R, U, R', F'

FS CUBING

Algorithms- I Shapes

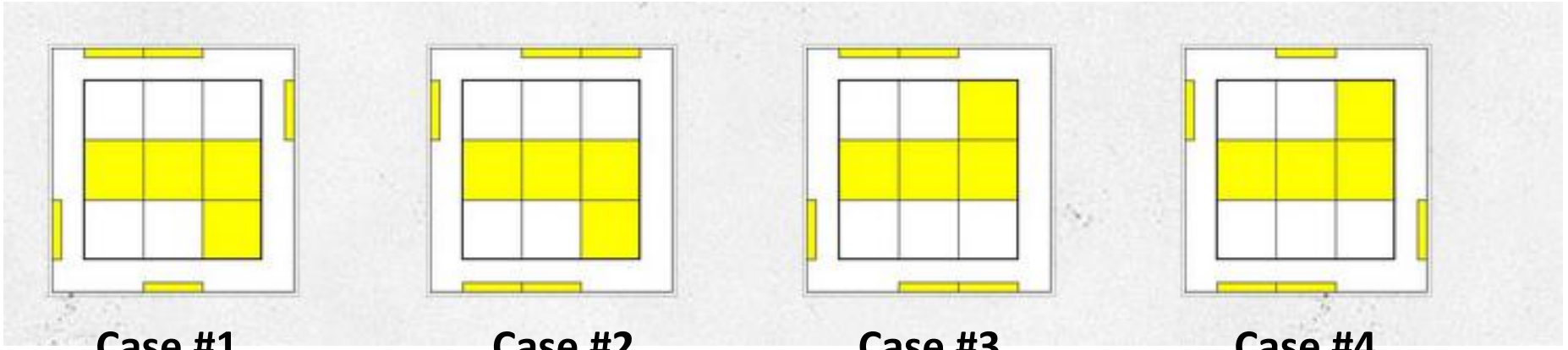


Case #1 **Case #2** **Case #3** **Case #4**

Case #1- R', U2, R2, U, R', U, R, U2, (x'), U', R', U, (x)
Case #2- R', U', R, U', R', d, R', U, R, B
Case #3- f, R, U, R', U', R, U, R', U', f'
Case #4- r', U', r, U', R', U, R, U', R', U, R, r', U, r

FS CUBING

Algorithms- Knight Moves

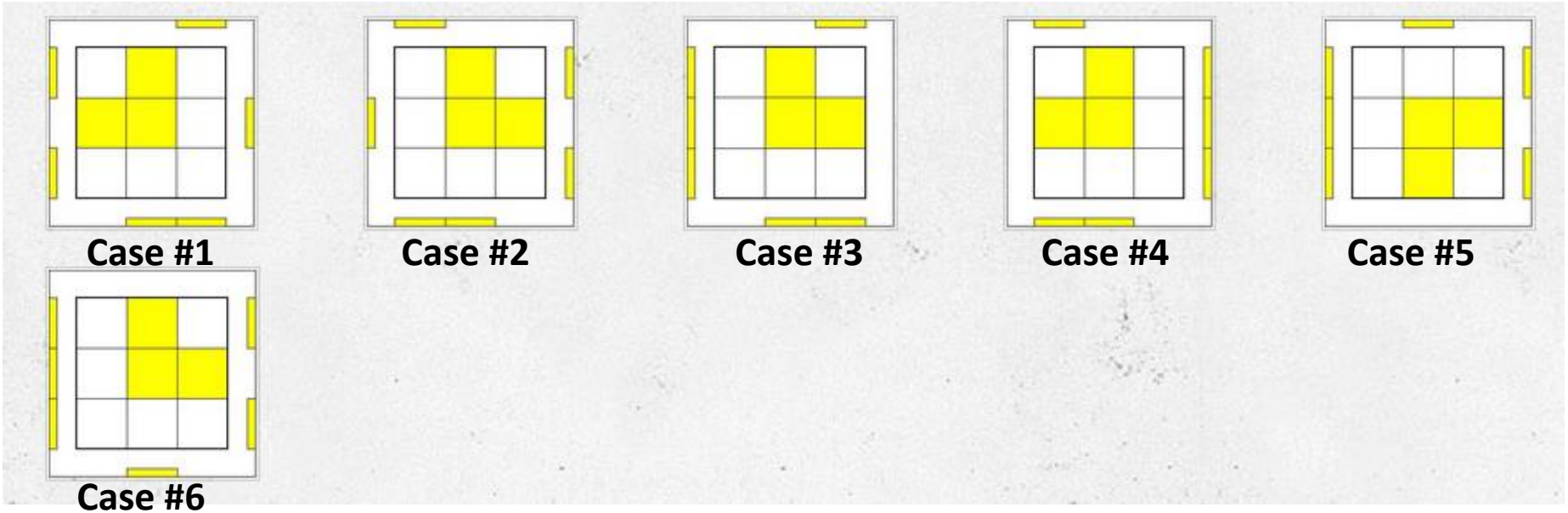


Case #1 Case #2 Case #3 Case #4

Case #1-R', F, R, U, l', U', L, (y'), R, U', R'
Case #2-L', B', L, R', U', R, U, L', B, L
Case #3-(x') R, U', R', F', R, U, R', (x, y), R', U, R
Case #4- L, F, L', R, U, R', U', L, F', L'

FS CUBING

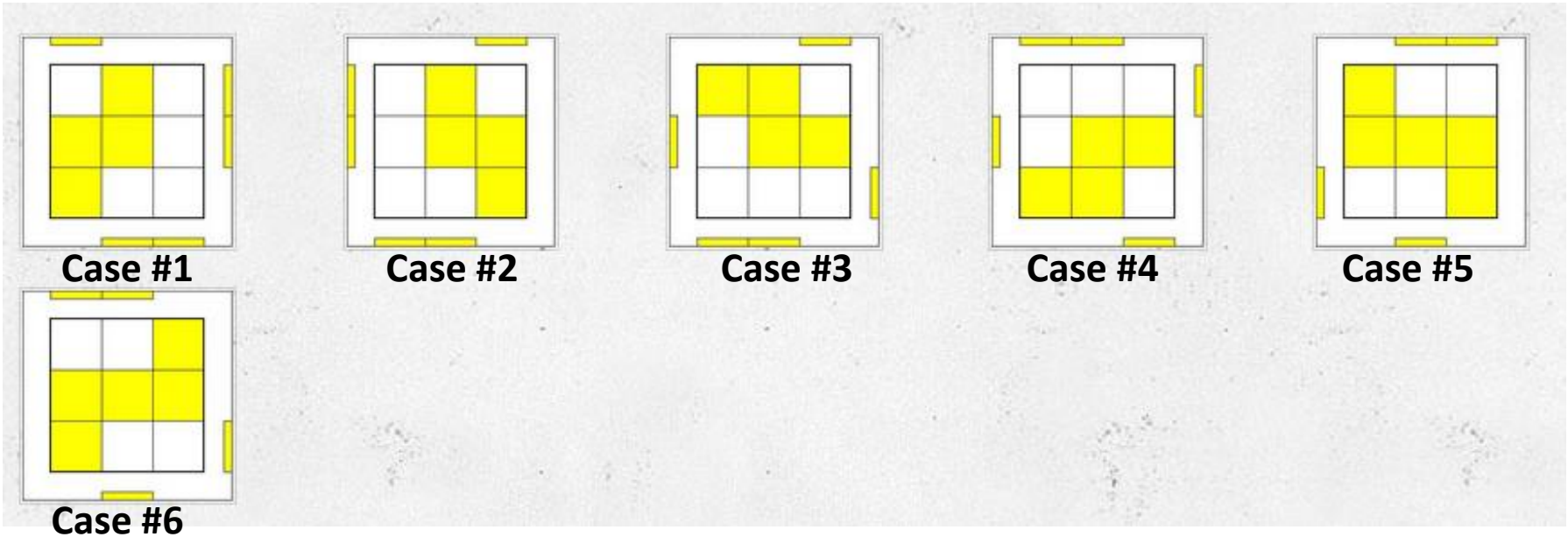
Algorithms- L Shapes



Case #1- F, R, U, R', U', R, U, R', U', F'
Case #2- F', L', U', L, U, L', U', L, U, F
Case #3- l', U, R', U', R, l, U2, (x'), U', R, U, l'
Case #4- R', F, R2, B', R2', F', R2, B, R'
Case #5- r', U', R, U', R', U, R, U', R', U2, r
Case #6- r, U, R', U, R, U', R', U, R, U2', r'

FS CUBING

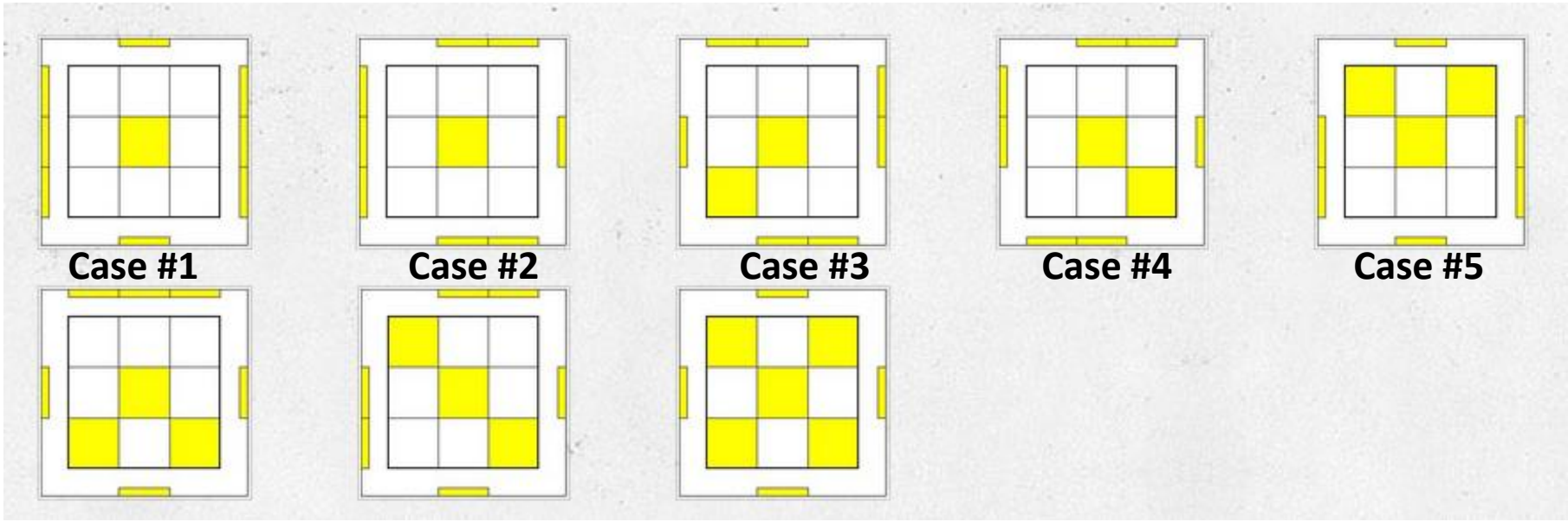
Algorithms- Lightening Bolts



Case #1- r, U, R', U, R, U2, r'
Case #2- l', U', L, U', L', U2', l
Case #3- r, R2', U', R, U', R', U2, R, U', R, r'
Case #4- r', R2, U, R', U, R, U2, R', U, R', r
Case #5- R', F, R, U, R', U', F', U, R
Case #6- L, F', L', U', L, U, F, U', L'

FS CUBING

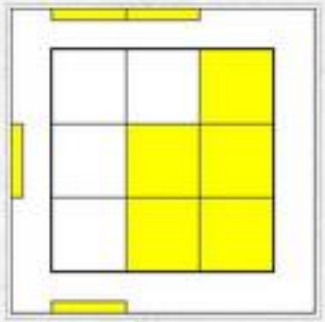
Algorithms- No Edges Flipped Correctly



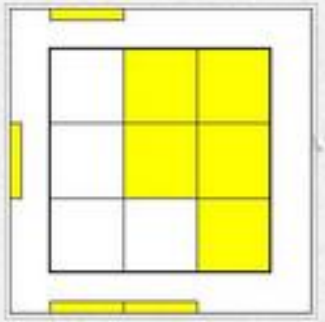
Case #1- R, U, (x'), U', R, U, l', R', U', l', U, l, F'
Case #2- F, R, U, R', U', F', f, R, U, R', U', f'
Case #3- r', R2, U, R', U, r, U2, r', U, R',r
Case #4- r', R, U', r, U2, r', U', R, U', R2', r
Case #5- r', R, U, R, U, R', U', r, R2', F, R, F'
Case #6- F, R, U, R', U, (y'), R', U2, R', F, R, F'
Case #7- R, U, R', U, R', F, R, F', U2, R', F, R, F'
Case #8- r', R, U, R, U, R', U', r2, R2', U, R, U', r'

FS CUBING

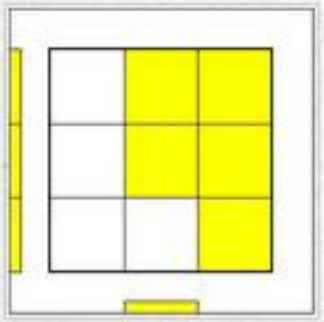
Algorithms- P Shapes



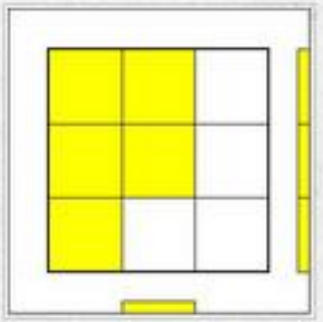
Case #1



Case #2



Case #3

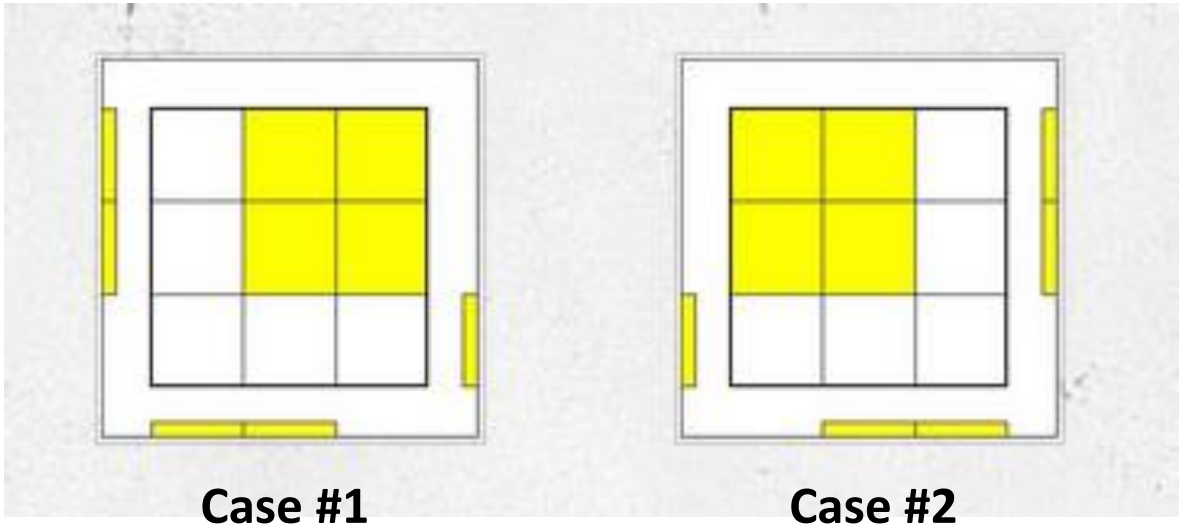


Case #4

Case #1- R, U, B', U', R', U, l, U, l'
Case #2- R', U', F, U, R, U', R', F', R
Case #3- F', U', L', U, L, F
Case #4- F, U, R, U', R', F'

FS CUBING

Algorithms- Square Shapes

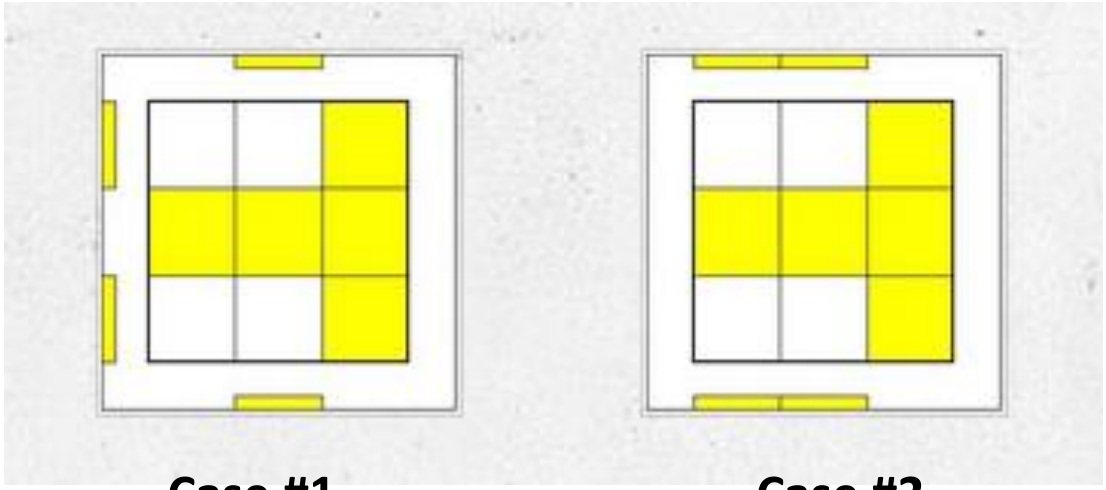


Case #1 **Case #2**

Case #1- r, U2, R', U', R, U', r'
Case #2- l, U2, L, U, L', U, l

FS CUBING

Algorithms- T Shapes

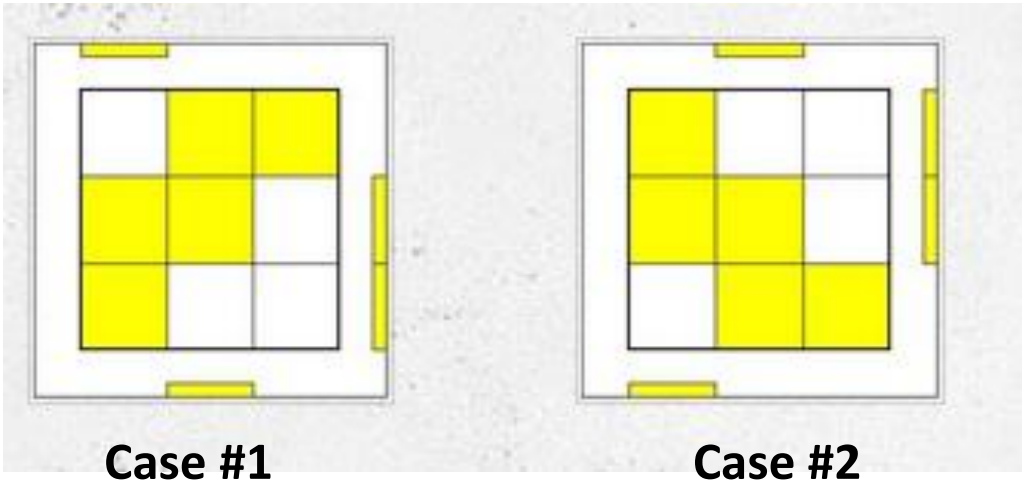


Case #1 **Case #2**

Case #1- F, R, U, R', U', F'
Case #2- R, U, R', U, R', F, R, F'

FS CUBING

Algorithms- W Shapes



Case #1 **Case #2**

Case #1- R, U, R', U, R, U', R', U', R', F, R, F'
Case #2- R', U', R, U', R', U, R, U, l, U', R', U, (x)

FS CUBING