IMPROVER METHOD – PT. 3

MICHAEL GEORGE

WCA ID: 2015GEOR02

ORIENTATION OF THE LAST LAYER (**OLL**)

"Beginner" method used 2 simple algorithms

- 6-move modified trigger **F (U R U' R') F'** for edge orientation (**EOLL**)
- 8-move combination of triggers **(R U2 R' U2')** and **(U R U' R')** for corner orientation (**OCLL**)

"Improver" method will use 4 simple algorithms

- 6-move algorithms for edge orientation (**EOLL**) – **F R U** moves
- 7-move algorithms for corner orientation (**Sune** and **Anti-Sune**) – **R U** moves

The new algorithms will be easy to learn and incorporate into your solves

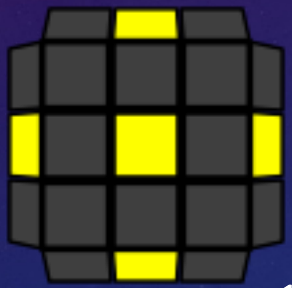
- The beginner **Anti-Sune** (combination of triggers) will be shortened from 8 moves to 7 moves
- The additional **EOLL** + **OCLL** algorithms will each be a simple "**inverse**"

EDGE ORIENTATION OF THE LAST LAYER (**EOLL**)

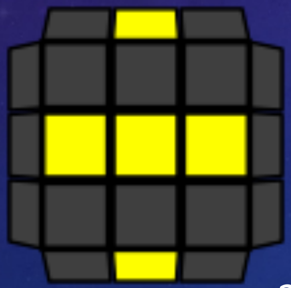
There are 4 possible cases during **EOLL** including the “**solved**” case

The “**adjacent edge flip**” is the most common **EOLL** as it occurs in **4/8** solves (i.e. **50%** of the time)

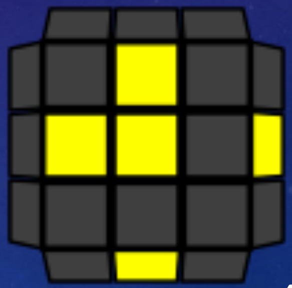
The other two cases take longer to solve if you only know the **F (U R U’ R’) F’** algorithm



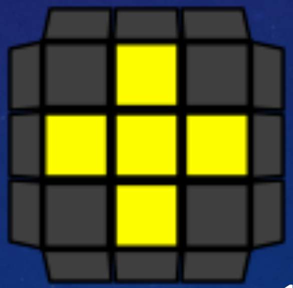
1/8



2/8



4/8



1/8

EOLL ALGORITHMS

To change the orientation of edges you will use a mixture of **F R U** moves

- The first **EOLL** algorithm that will be used is **F (U R U’ R’) F’**
  - Notice how it includes the **U R U’ R’** trigger that is used during **F2L**
  - The **F** is referred to as a “**setup**” move and the **F’** will undo the setup
  - The algorithm “**flips**” two adjacent LL edges (UF and UR) and has a side effect of “**permuting**” LL pieces
- The second **EOLL** algorithm that will be used is **F (R U R’ U’) F’**
  - Notice how it includes the **R U R’ U’** trigger that is used during **F2L**
  - The **F** is referred to as a “**setup**” move and the **F’** will undo the setup
  - The algorithm “**flips**” two opposite LL edges (UF and UB) and has a side effect of “**permuting**” LL pieces

Note how the two algorithms are both an “**inverse**” of each other

DOT CASE

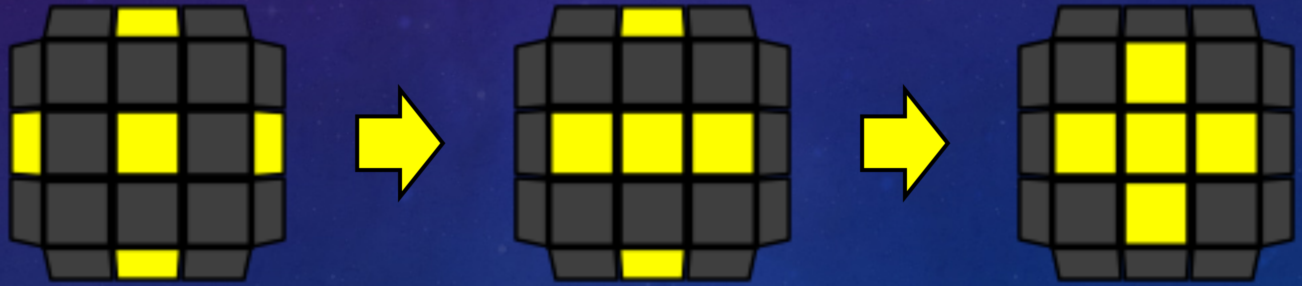
The “**Dot**” case will still take the most effort to solve as it will require both algorithms

Approach: “**Dot**” -> “**opposite edge flip**” -> “**solved**”

Setup: Ensure the “**opposite edge flip**” is a horizontal line before executing the **F (R U R’ U’) F’** algorithm

Algorithm: **F (U R U’ R’) F’ U’ F (R U R’ U’) F’**

e.g.



Dot Opposite Solved

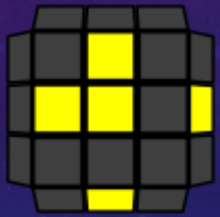
EOLL EFFICIENCY

“Beginner” – 1 algorithm – **F (U R U’ R’) F’**

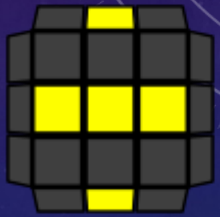
• **Adjacent Flip** / **L** / **Clock**	**6** moves, **1** look
• **Opposite Flip** / **Line**	**12** moves, **2** looks
• **4-Flip** / **Dot**	**19** moves, **3** looks
• **“Weighted Average”**	**8.4** moves, **1.5** looks

“Improver” – 2 algorithms – **F (U R U’ R’) F’** and **F (R U R’ U’) F’**

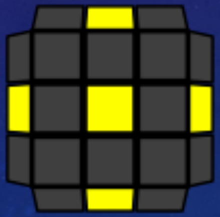
• **Adjacent Flip** / **L** / **Clock**	**6** moves, **1** look
• **Opposite Flip** / **Line**	**6** moves, **1** look
• **4-Flip** / **Dot**	**13** moves, **2** looks
• **“Weighted Average”**	**6.1** moves, **1.1** looks



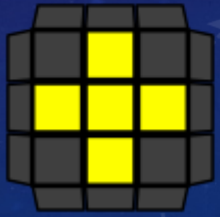
Adjacent



Opposite



Dot

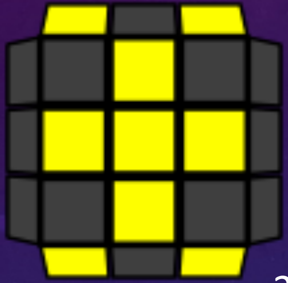


Solved

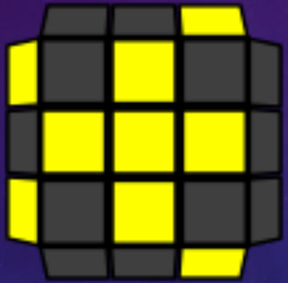
ORIENTING CORNERS OF THE LAST LAYER (OCLL)

There are 8 possible cases during **OCLL** including the “**solved**” case

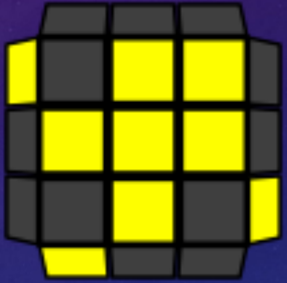
2 twisted corners require the most effort if you only know the triggers **(R U2 R’ U2’)** and **(U R U’ R’)**



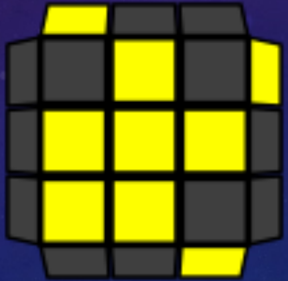
2/27



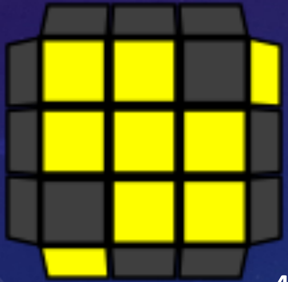
4/27



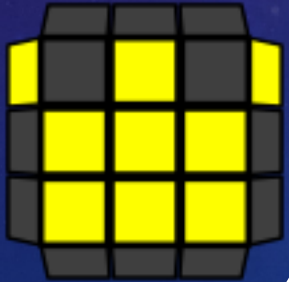
4/27



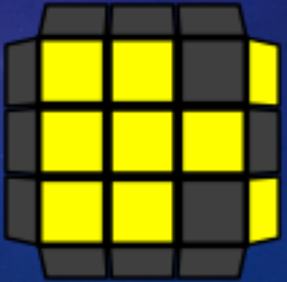
4/27



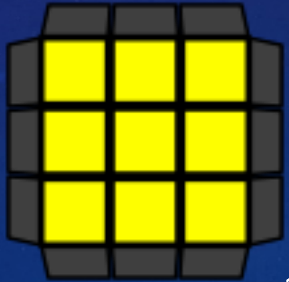
4/27



4/27



4/27



1/27

OCLL ALGORITHMS

To affect the orientation of corners on the last layer you only need to use **R U** moves

- The first **OCLL** algorithm is a combination of two triggers with a "**cancellation**" – **R U2 R' U' R U' R'**
  - The **R U2 R' U2'** trigger extracts the front-right **F2L** pair
  - The **U R U' R'** trigger re-inserts the front-right **F2L** pair
  - The **(R U2 R' U2') (U R U' R')** cancels to **R U2 R' U' R U' R'** and is known as the "**Anti-Sune**" algorithm
  - The "**Anti-Sune**" algorithm will "**twist**" three LL corners counter-clockwise and will "**permute**" the LL edges

- The second **OCLL** algorithm is a combination of two triggers with a "**cancellation**" – **R U R' U R U2' R'**
  - The **R U R' U'** trigger extracts the front-right **F2L** pair
  - The **U2 R U2' R'** trigger re-inserts the front-right **F2L** pair
  - The **(R U R' U') (U2 R U2' R')** cancels to **R U R' U R U2' R'** and is known as the "**Sune**" algorithm
  - The "**Sune**" algorithm will "**twist**" three LL corners clockwise and will "**permute**" the LL edges

Note how the two algorithms are both an "**inverse**" of each other

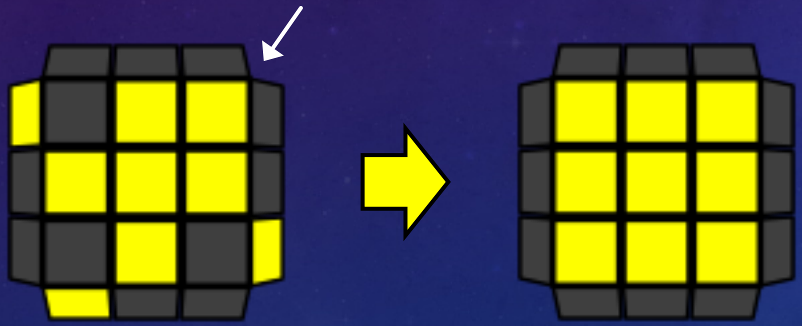
3 TWISTED CORNERS

There are 2 **OCLL** cases with 3 twisted corners

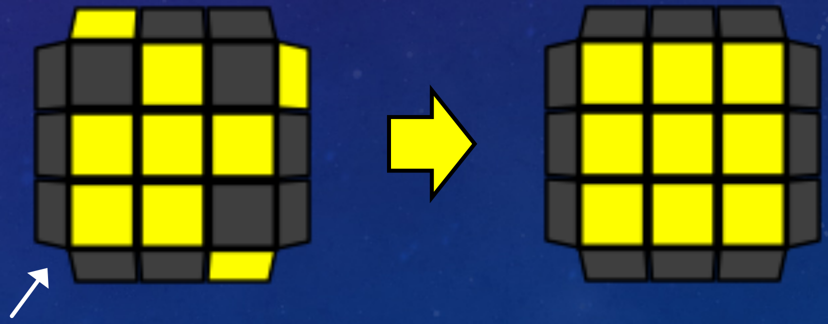
Approach: “**Sune**” or “**Anti-Sune**” -> “**Solved**”

Setup: Ensure the “**oriented**” corner is positioned correctly before executing the **OCLL** algorithm

Algorithm: **R U2 R’ U’ R U’ R’** or **R U R’ U R U2’ R’** – depending on the case



Anti-Sune **Solved**



Sune **Solved**

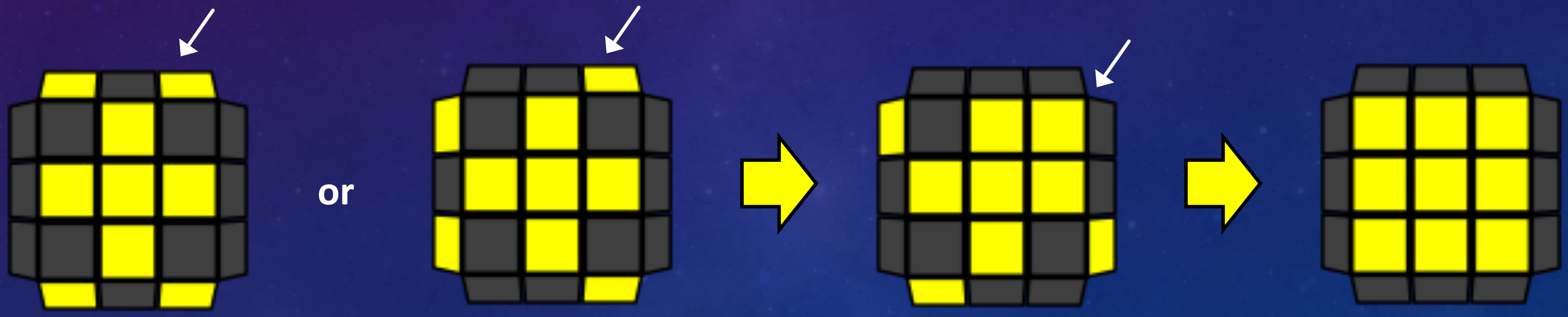
4 TWISTED CORNERS

There are 2 **OCLL** cases with 4 twisted corners

Approach: "**H / Double Sune**" or "**Pi / Bruno**" -> "**Anti-Sune**" -> "**Solved**"

Setup: Ensure the back-right corner has its U-sticker on the back before executing the **OCLL** algorithm

Algorithm: **R U2 R' U' R U' R'** – twice, remembering to **AUF** prior to execution



H / Double Sune

Pi / Bruno

Anti-Sune

Solved

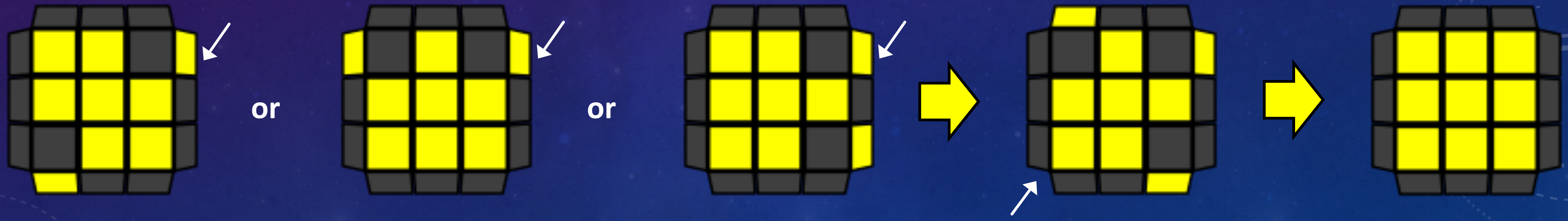
2 TWISTED CORNERS

There are 3 **OCLL** cases with 2 twisted corners

Approach: "**L / Bowtie**" or "**T / Chameleon**" or "**U / Headlights**" -> "**Sune**" -> "**Solved**"

Setup: Ensure the back-right corner has its U-sticker on the right before executing the **OCLL** algorithm

Algorithm: **R U2 R' U' R U' R'** then **R U R' U R U2' R'** – remembering to **AUF** prior to execution



L / Bowtie

T / Chameleon

U / Headlights

Sune

Solved

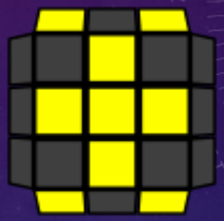
OCLL EFFICIENCY

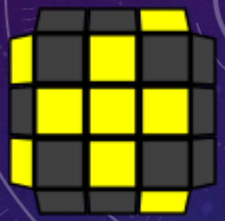
"Beginner" – 1 algorithm – **(R U2 R' U2') (U R U' R')**

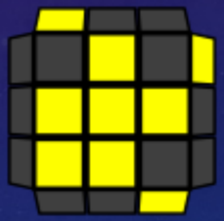
4 twisted corners	**16** or **17** moves, **2** looks
3 twisted corners	**8** or **17** moves, **1** or **2** looks
2 twisted corners	**26** moves, **3** looks
"Weighted Average"	**19.0** moves, **2.3** looks

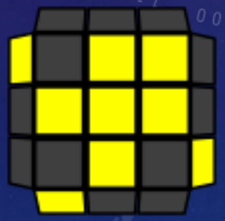
"Improver" – 2 algorithms – **R U2 R' U' R U' R'** and **R U R' U R U2' R'**

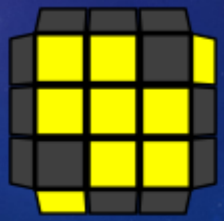
4 twisted corners	**14** or **15** moves, **2** looks
3 twisted corners	**7** moves, **1** look
2 twisted corners	**15** moves, **2** looks
"Weighted Average"	**12.0** moves, **1.7** looks

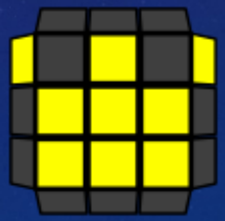

H / Double Sune

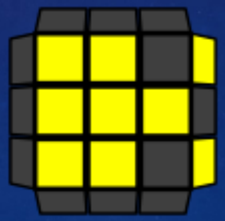

Pi / Bruno

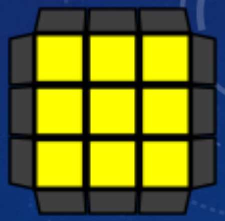

Sune


Anti-Sune


L / Bowtie


T / Chameleon


U / Headlights


Solved

OLL EFFICIENCY

"Beginner" – 2 algorithms – One **EOLL** algorithm + beginner **Anti-Sune**

- **"Worst Case"** **45** moves, **6** looks
- **"Weighted Average"** **27.3** moves, **3.8** looks

"Improver" – 4 algorithms – Two **EOLL** algorithms + **Sune** + **Anti-Sune**

- **"Worst Case"** **28** moves, **4** looks
- **"Weighted Average"** **18.1** moves, **2.8** looks

Comparison

- **"Worst Case"** "Improver" method saves **17** moves, **2** looks –> **37.8%** saving
- **"Weighted Average"** "Improver" method saves **9.2** moves, **1** look –> **33.7%** saving

NEARLY DONE!