QTER

Arhan, Henry, Asher

What is qter?

```
Puzzles
A: 3x3

1  | input "First number"
           R' F' L U' L U L F U' R
           max-input 90
2  | input "Second number"
           U F R' D' R2 F R' U' D
           max-input 90
3  | B2 R L2 D L' F' D2 F' L2
     B' U' R D' L' B2 R F
4  | solved-goto DFR FR 6
5  | goto 3
6  | R' F' L U' L U L F U' R
7  | R' U F' L' U' L' U L' F R
8  | solved-goto ULF UL 13
9  | R' U F' L' U' L' U L' F R
10 | solved-goto ULF UL 13
11 | U F R' D' R2 F R' U' D
12 | goto 7
13 | halt "The average is"
          D' U R F' R2 D R F' U'
          counting-until DFR FR
```

How does qter work?

How does qter work?

"Zero"

How does qter work?

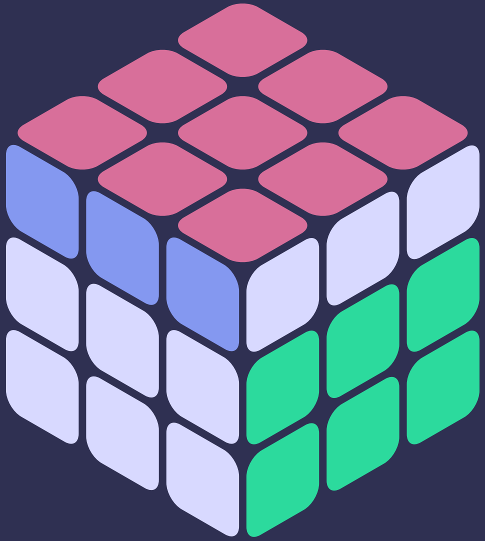
"One"

How does qter work?

"Two"

How does qter work?

"Three"



How does qter work?

"Four?"

Can we do math?

$1 + 2 \rightarrow (\mathrm{Up})(\mathrm{Up\ Up})$

"One"

Can we do math?

$$1 + 2 \rightarrow (\mathrm{Up})(\mathrm{Up\ Up}) = \mathrm{Up\ Up\ Up} \rightarrow 3$$

"Three"

Bigger numbers?

(Right)(Up)

Conditional jump?

$((\text{Right})(\text{Up}))\ \times ?$

Conditional jump?

$((\text{Right})(\text{Up})) \times 0$

Multiple registers?

$(\mathrm{Up})(\mathrm{Down})$

Examples of architectures

1260

- R U2 D’ B D’

90×90

- R’ F’ L U’ L U L F U’ R
- U F R’ D’ R2 F R’ U’ D

30×30×30

- U L2 B’ L U’ B’ U2 R B’ R’ B L
- R2 L U’ R’ L2 F’ D R’ D L B2 D2
- L2 F2 U L’ F D’ F’ U’ L’ F U D L’ U’

Examples of architectures

30×18×10×9

- U L B’ L B’ U R’ D U2 L2 F2
- D L’ F L2 B L’ F’ L B’ D’ L’
- R’ U’ L’ F2 L F U F R L U’
- B2 U2 L F’ R B L2 D2 B R’ F L

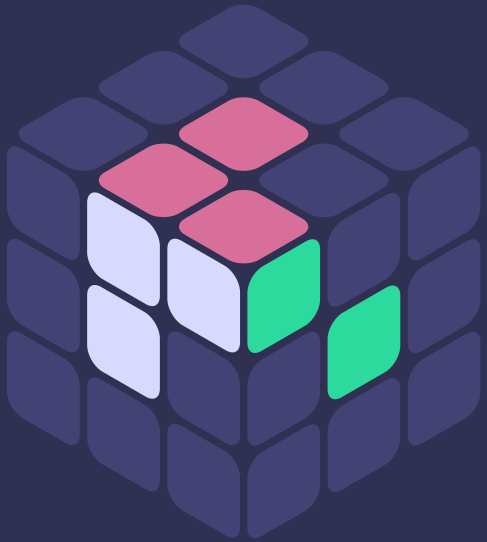
What about solved-goto?

Register "Up" is zero

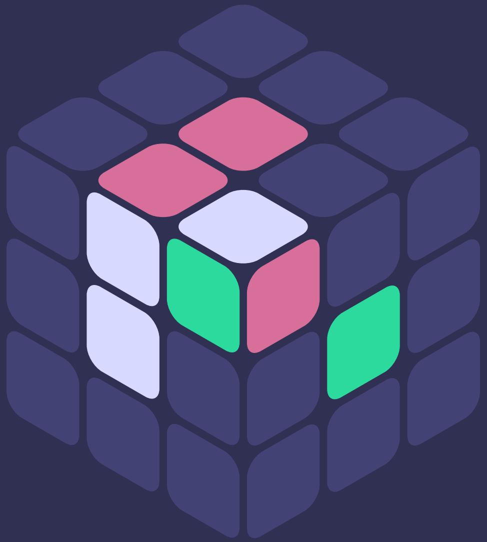
What about solved-goto?

```
solved-goto UF UFR 8
```

Branch taken



Branch not taken



How does this look in Q?

```
Puzzles
A: 3x3
B: 3x3

1 | U D
2 | goto 1
3 | solved-goto UF UFR 2
4 | switch B
```

QAT

Register declaration

Register declaration

```
.registers {
  A, B ← 3x3 builtin (90, 90)
}
```

Register declaration

```
.registers {
  A, B ← 3x3 builtin (90, 90)
}
```

—

```
.registers {
  A ← 3x3 builtin (1260)
  B ← 3x3 builtin (1260)
}
```

Primitive instructions

Primitive instructions

```
add A 1
```

Primitive instructions

```
add A 1
spot:
```

Primitive instructions

```
add A 1
spot:
goto spot
```

Primitive instructions

```
add A 1
spot:
goto spot
solved-goto A spot
```

Primitive instructions

```
add A 1
spot:
goto spot
solved-goto A spot
input "Your favorite number:" A
```

Primitive instructions

```
add A 1
spot:
goto spot
solved-goto A spot
input "Your favorite number:" A
halt "The result is" A
```

Macros

Macros

```
.macro if {
  (solved $R:reg $code:block) => {
        solved-goto $R do_if
        goto after_if
    do_if:
        $code
    after_if:
  }
}
```

Macros

```
.macro if {
  (solved $R:reg $code:block) => {
        solved-goto $R do_if
        goto after_if
    do_if:
        $code
    after_if:
  }
}

if solved A {
  add A 5
}
```

Macros

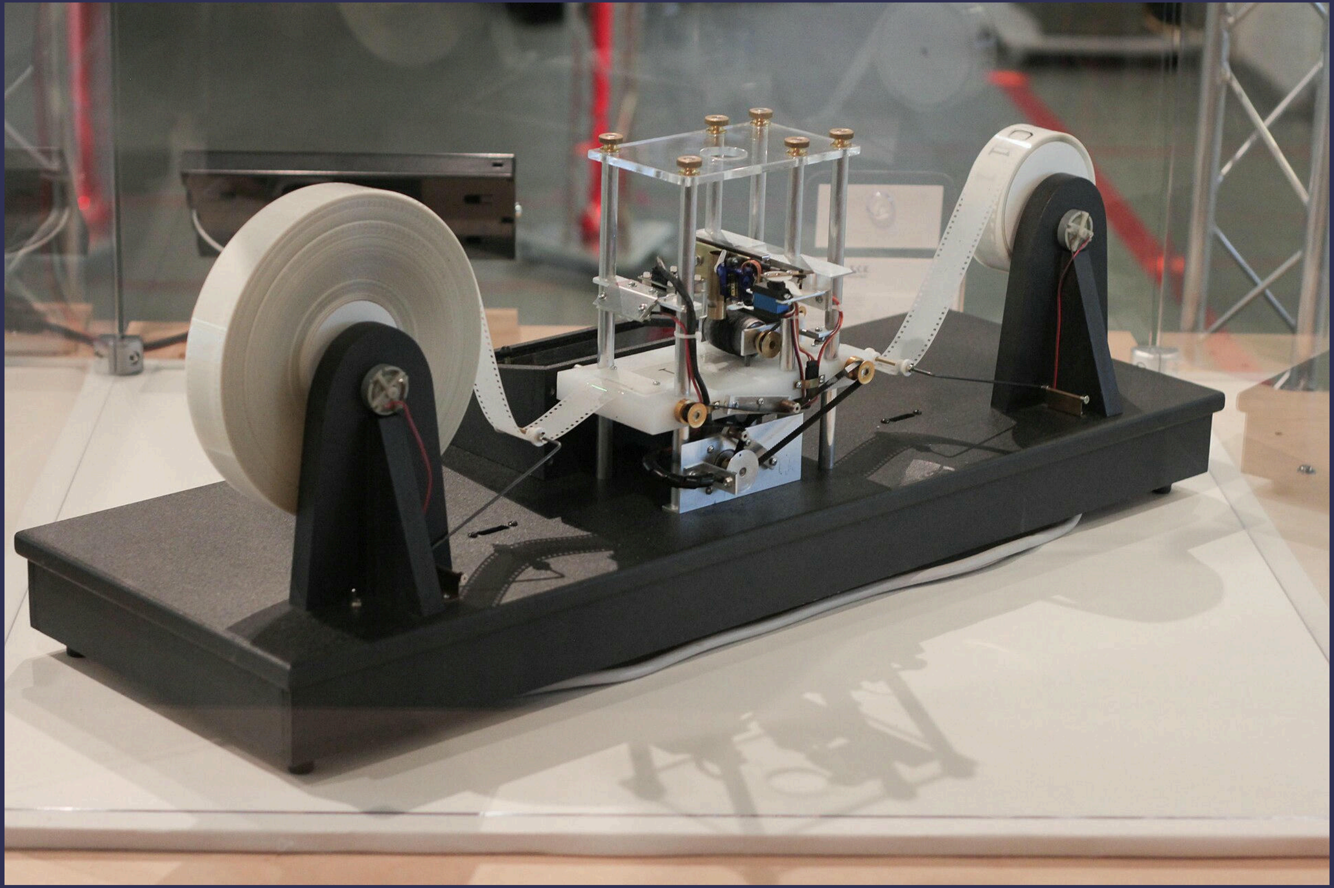
```
.macro if {
  (solved $R:reg $code:block) => {
        solved-goto $R do_if
        goto after_if
    do_if:
        $code
    after_if:
  }
}

if solved A {
  add A 5
}
```

```
    solved-goto A do_if
    goto after_if
do_if:
    add A 5
after_if:
```

Turing completeness

How can qter be turing complete?

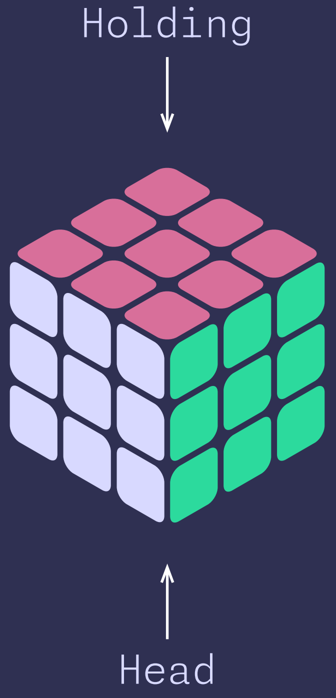


How can qter be turing complete?

```
Puzzles
tape A: 3x3

1 | move-right A
2 | switch-tape A
3 | R U
4 | move-left A
```

Next instruction: 0

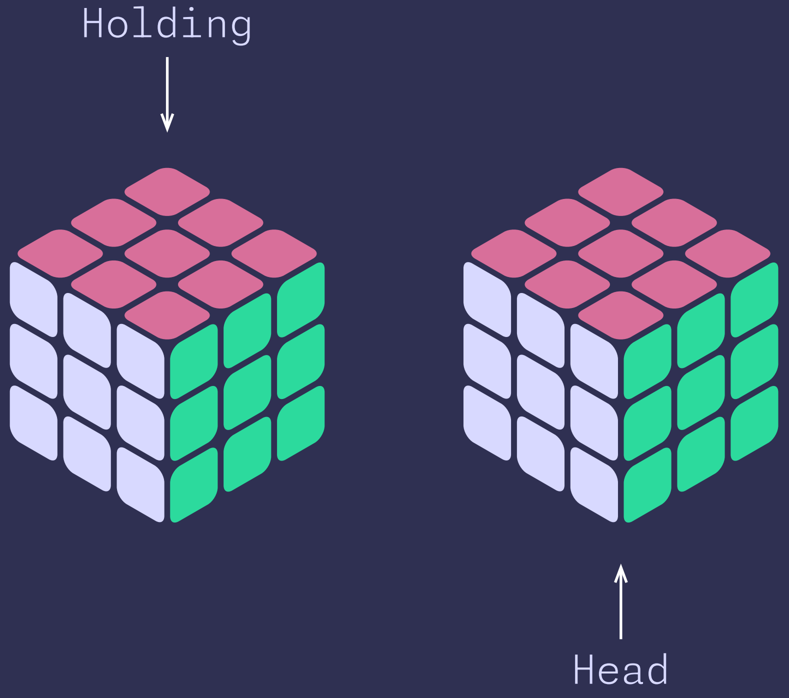


How can qter be turing complete?

```
Puzzles
tape A: 3x3

1 | move-right A
2 | switch-tape A
3 | R U
4 | move-left A
```

Next instruction: 1

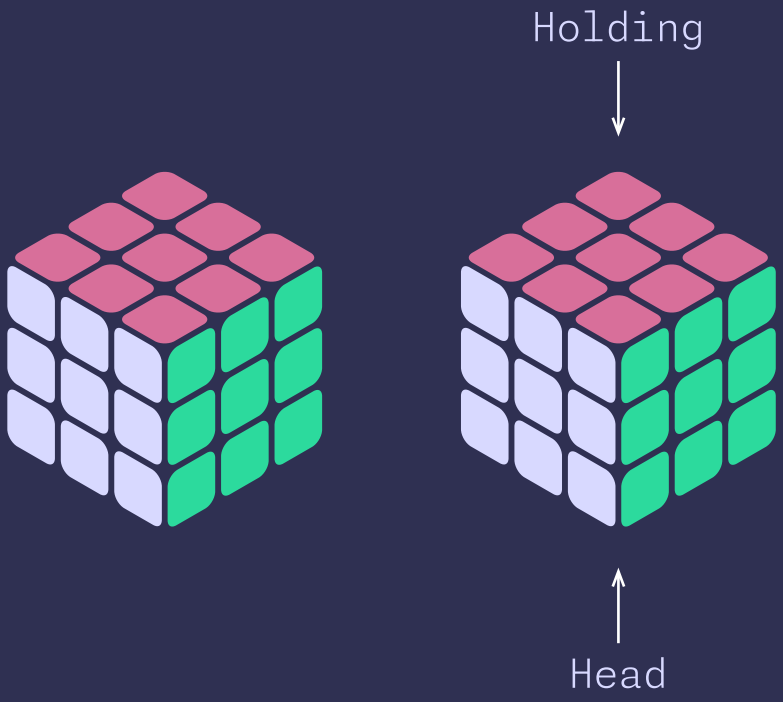


How can qter be turing complete?

Puzzles
tape A: 3x3

```
1 | move-right A
2 | switch-tape A
3 | R U
4 | move-left A
```

Next instruction: 2

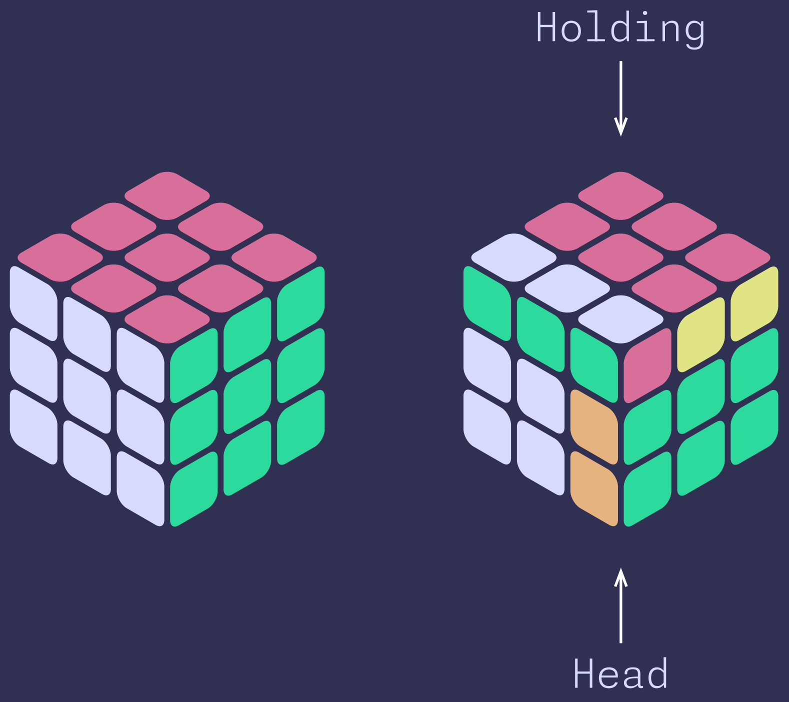


How can qter be turing complete?

```
Puzzles
tape A: 3x3

1 | move-right A
2 | switch-tape A
3 | R U
4 | move-left A
```

Next instruction: 3

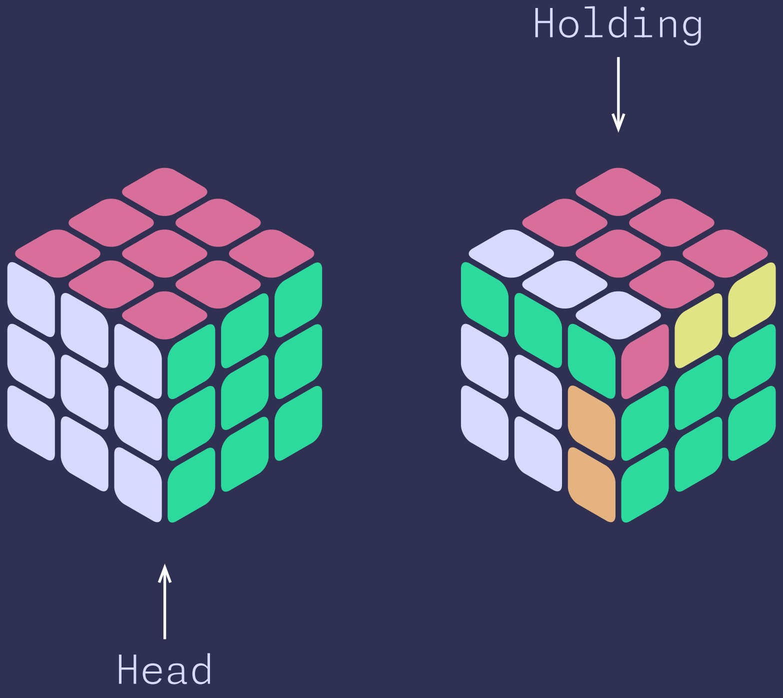


How can qter be turing complete?

Puzzles
tape A: 3x3

```
1 | move-right A
2 | switch-tape A
3 | R U
4 | move-left A
```

Next instruction: 4



But wouldn’t qter just be brainfuck?

But wouldn’t qter just be brainfuck?

- Multiple tapes are allowed
-
-

But wouldn’t qter just be brainfuck?

- Multiple tapes are allowed
- This makes call stacks easy
-

But wouldn’t qter just be brainfuck?

- Multiple tapes are allowed
- This makes call stacks easy
- We can use a global register to keep track of the head position

But wouldn’t qter just be brainfuck?

```
.macro index {
  ($tape:tape $current:reg $to:reg) => {
    while not-solved $current {
      dec $current
      move-left $tape
    }

    while not-solved $to {
      dec $to
      inc $current
      move-right $tape
    }
  }
}
```

How do we find qter registers?

An extremely simplified overview

Qter Architecture Solver

- Computes optimal qter registers in two phases
  - Cycle Combination Prover: Find best cycles that provably exist
  - Cycle Combination Solver: Find shortest algorithms that produce the cycles

Cycle Combination Prover

The maximum number of repetitions for an algorithm on the Rubik’s cube is 1260

Cycle Combination Prover

The maximum number of repetitions for an algorithm on the Rubik’s cube is 1260

This is formed from:

- LCM 56 on edges: 4 cycle, another 4 cycle, and 7 cycle
- LCM 45 on corners: 9 cycle and 15 cycle
- LCM(45, 56) = 1260

Cycle Combination Prover

- We can generalize this idea!

Cycle Combination Prover

- We can generalize this idea!
- N registers, not just one
-

Cycle Combination Prover

- We can generalize this idea!
- N registers, not just one
- Any twisty puzzle, like the 4x4x4 or megaminx

Cycle Combination Solver

We have a structure of the cycles we want. Now, find an actual algorithm for the cycle.

Cycle Combination Solver

We have a structure of the cycles we want. Now, find an actual algorithm for the cycle.

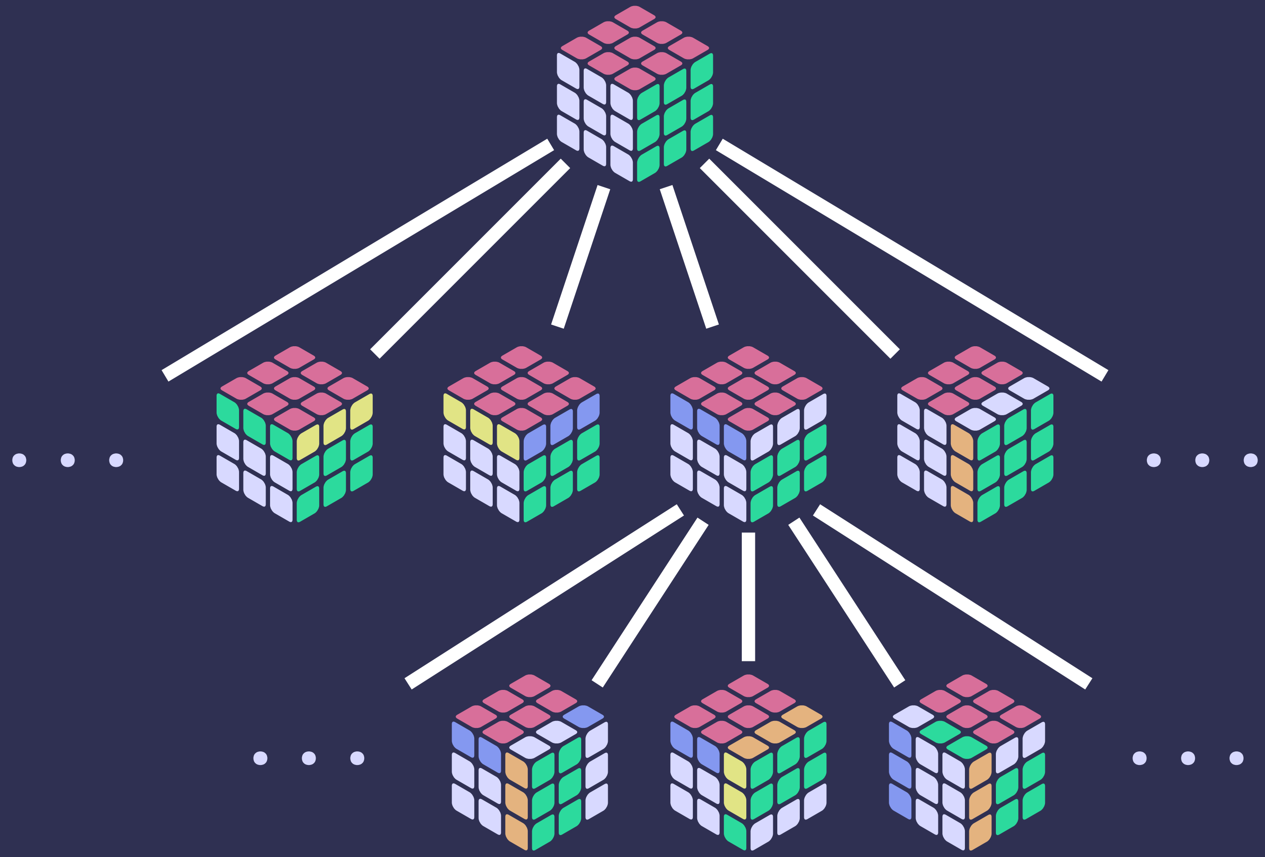
- The algorithm must be as short as possible
-

Cycle Combination Solver

We have a structure of the cycles we want. Now, find an actual algorithm for the cycle.

- The algorithm must be as short as possible
- The only known optimal solving technique is brute force :-(

Cycle Combination Solver



Cycle Combination Solver

Modified Korf’s algorithm

Cycle Combination Solver

Modified Korf’s algorithm

- Iterative DFS + heuristic
-
-
-

Cycle Combination Solver

Modified Korf’s algorithm

- Iterative DFS + heuristic
- Movecount coefficient calculator
-
-

Cycle Combination Solver

Modified Korf’s algorithm

- Iterative DFS + heuristic
- Movecount coefficient calculator
- Fixed pieces
-

Cycle Combination Solver

Modified Korf’s algorithm

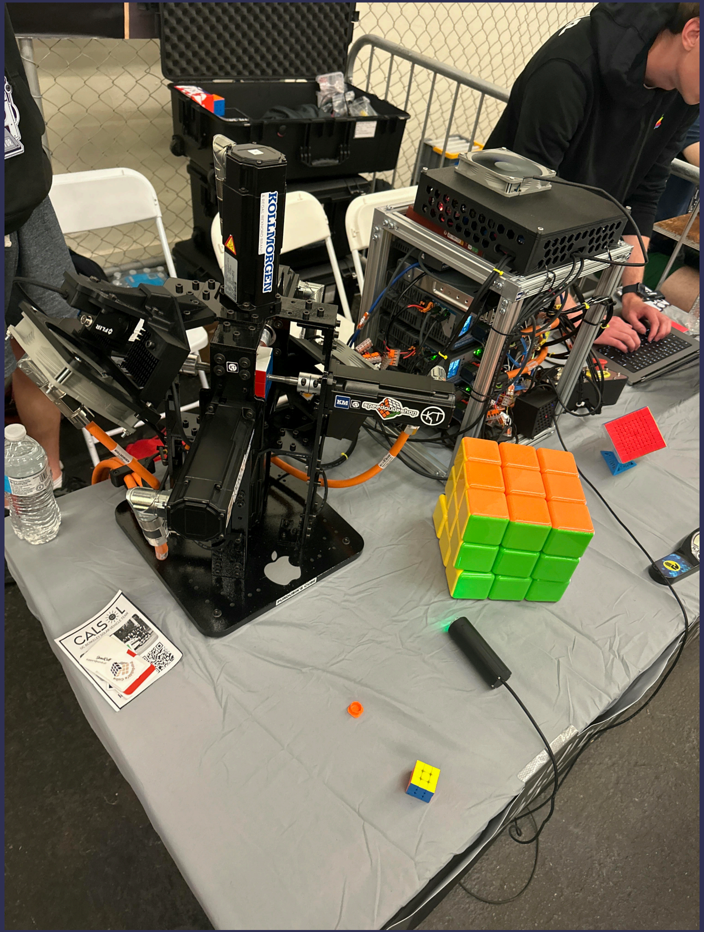
- Iterative DFS + heuristic
- Movecount coefficient calculator
- Fixed pieces
- ... The optimizations gets complicated

We integrated Qter into a robot!

Raw video: https://drive.google.com/file/d/121oxXZX2t8l1pAY0NNbxVoiUOWuV8dqR/view?usp=drive_link

Slo-mo video: https://drive.google.com/file/d/1dQrUkTKFgRiQjZEsESq42mu1uAC41Vrr/view?usp=drive_link

We demoed Qter at OpenSauce 2025!



RESTROOMS
ELECTRICAL AND COMPUTER ENGINEERING
Qter
The Rubik's Cube Computer
CERTIFICATE

The future of qter

Thank you!

Any questions?