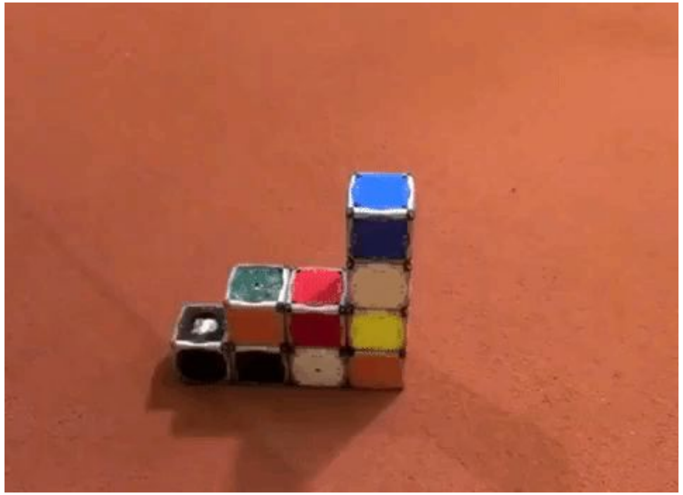
Reconfiguration of Self-Assembling Modular Robots

Tufts Computational Geometry Lab

Colton Wolk | 07.24.2020

Square Robots Model Applications

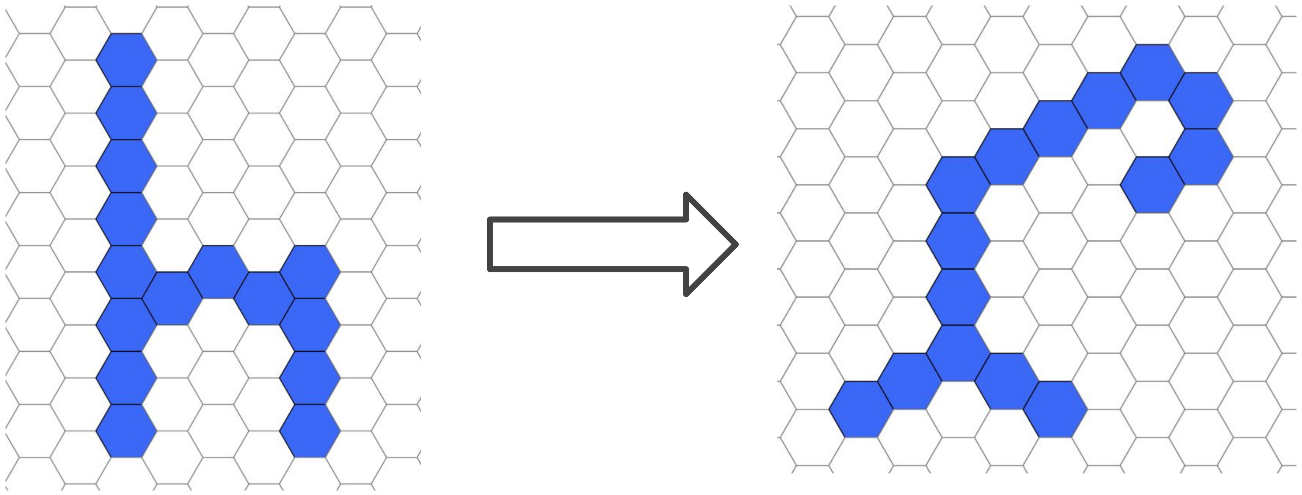
Roombots by the Swiss Federal Institute of Technology Lausanne (2020)



M-Blocks by MIT (2013)

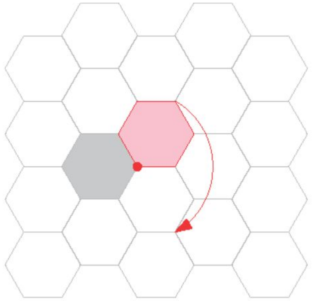
Hexagonal Model in 2D

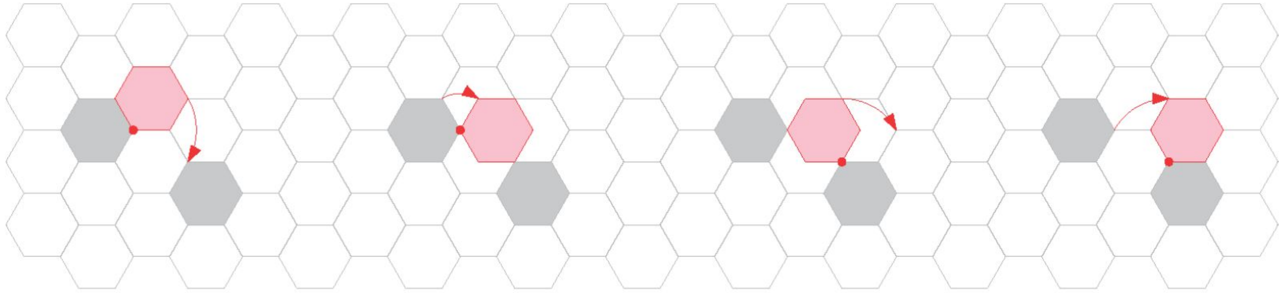
- The reconfiguration problem: Can we transform one configuration of the hexagonal robot modules into any other configuration?
- Example: transforming the modules from a chair to an arm/grabber



Hexagonal Model in 2D

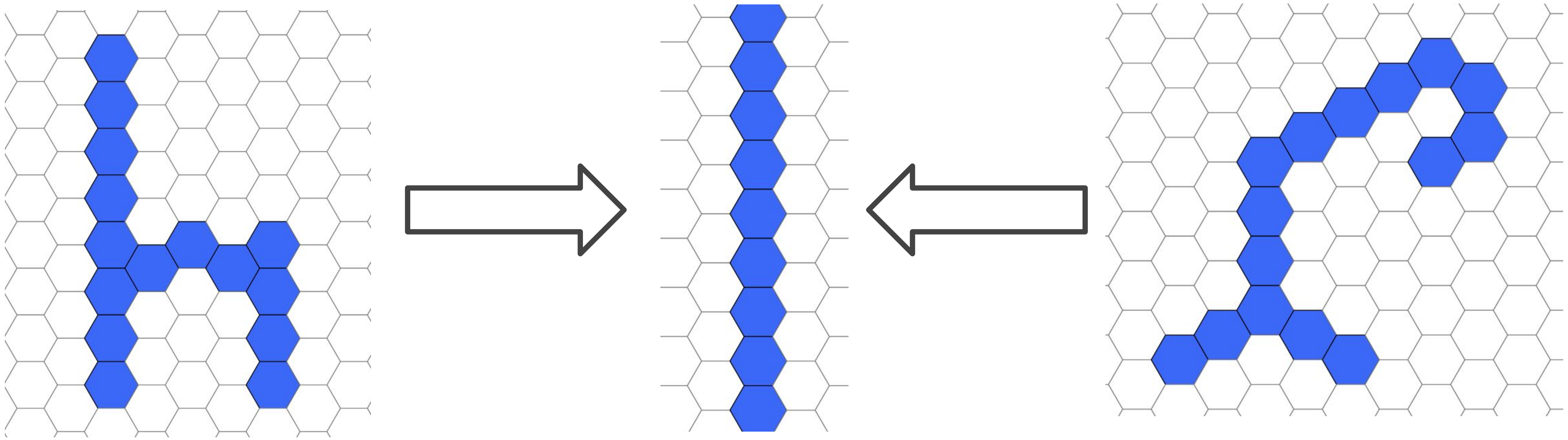
- The reconfiguration problem: Can we transform one configuration of the hexagonal robot modules into any other configuration? How?
- Only two types of moves are allowed:
  - Restricted: pivot around a connected module until sides are connected
  - Monkey: pivot around a connected module then transfer to a nearby one
- Must always stay connected; cannot “jump” across modules


Restricted move


Monkey move

The Algorithm

- Idea: transform both configurations into a standard vertical strip
- If we can transform them both into this intermediate shape, then we can transform one into the other

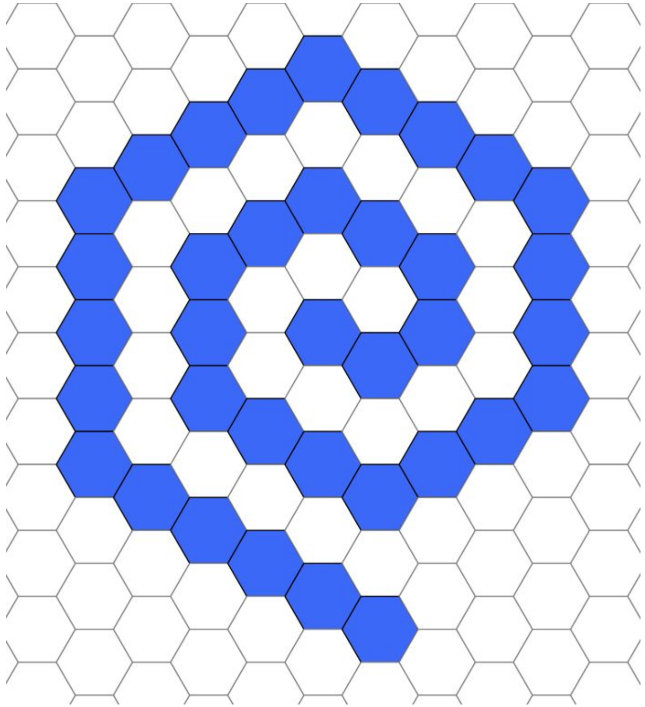


My Goals

- Implement and visualize the algorithm in python
- Combine functionality with an existing hexagon drawing tool
- Expand testing and ensure correctness

The Algorithm

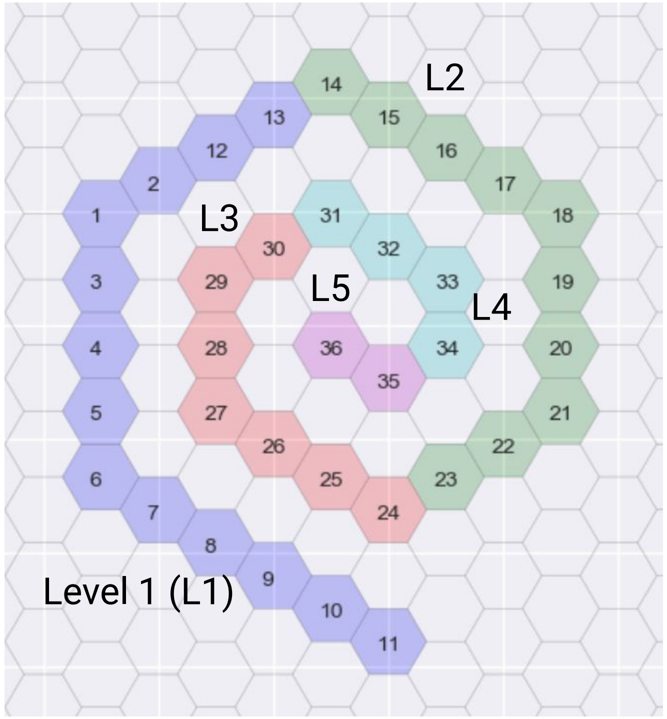
- *Levels*: measures how far away any module is from the outermost modules in the configuration
- *Moving trajectory*: given a particular module, where are we allowed to move it?



The Algorithm (Levels)

- *Levels*: measures how far away any module is from the outermost modules in the configuration
- *Moving trajectory*: given a particular module, where are we allowed to move it?

- On the right, we define the outermost modules in blue, and each color represents a different level
- Each additional level is representative of how much more we have to move a module to reach Level 1 (blue)

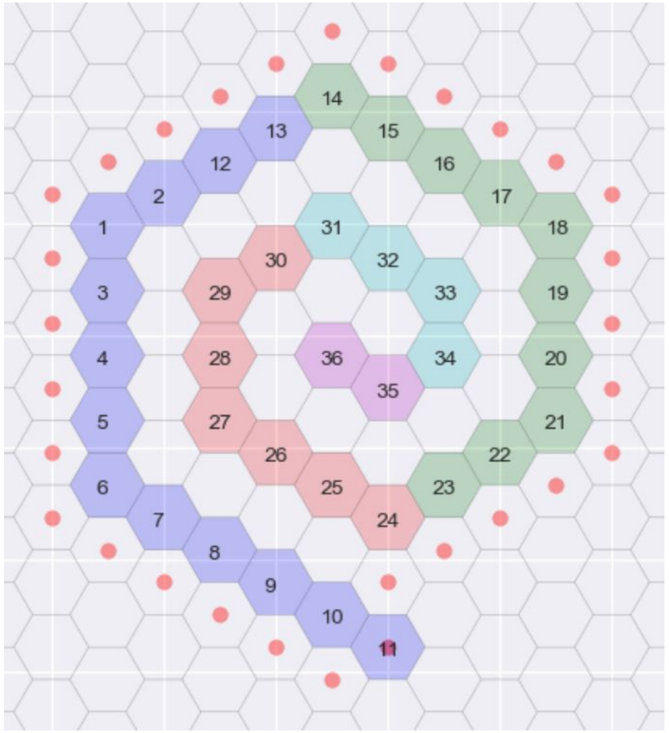


Program Output

The Algorithm (Moving)

- *Levels*: measures how far away any module is from the outermost modules in the configuration
- *Moving trajectory*: given a particular module, where are we allowed to move it?

- On the right, red dots represent positions where module 11 is allowed to move
- Module 11 cannot access the interior because it is wedged between modules 10 and 24 (there isn't enough room)
- The robot must always be connected



Program Output

Summary and Future Work

- Created a program for previously purely theoretical work
  - Implemented levels, moving trajectory, and other parts
- Goal: implement and visualize the complete algorithm
  - E.g., show dynamic movement of modules
- Eventually to be used for testing the algorithm, verifying correctness
- Can be adapted for multiple uses relating to modular robots

Thank you

Mentors:

- Prof. Matias Korman
- Prof. Diane Souvaine
- Andrew Gonczi
- and other lab members