Toward a 2D Modular and Self-Reconfigurable Robot for Conveying Microparts

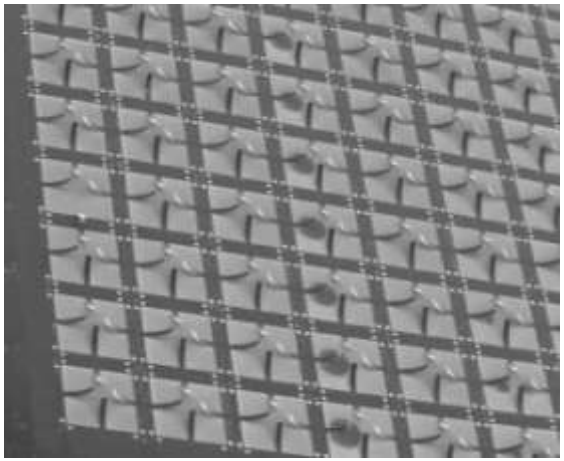
Sebastian Möbes, **Benoît Piranda**, Guillaume J. Laurent, Julien Bourgeois, Cédric Clévy, Nadine Le Fort-Piat

Work funded by:
ANR (ANR-2011-BS03-005)

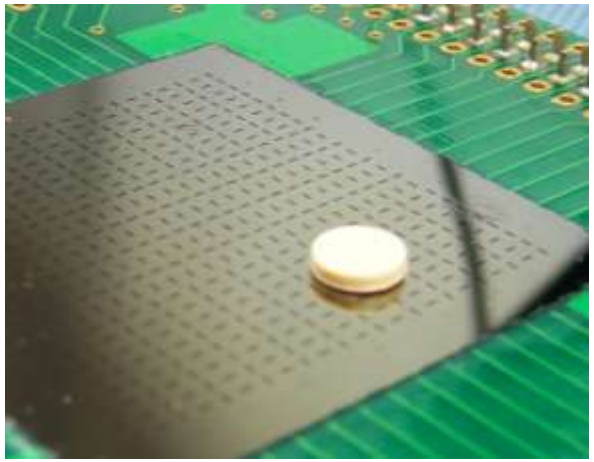
femto-st
sciences & technologies

Smart surfaces

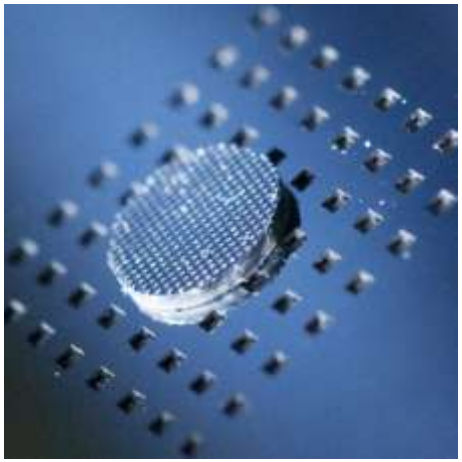
- **Distributed manipulation**
- **Up to 3 degrees of freedom**
- **Pneumatic => fast transport**
- **Ciliary actuators => high precision**



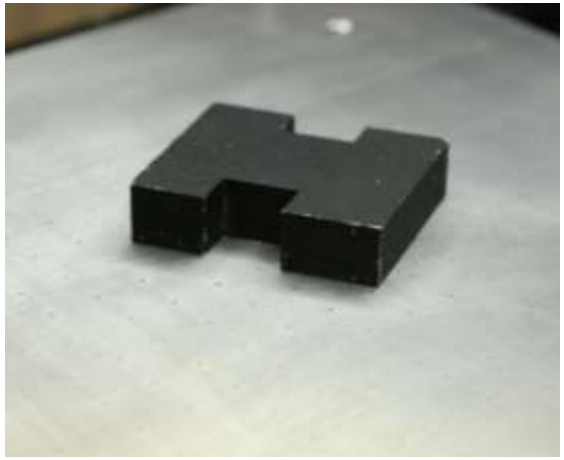
Ciliary actuator array
Ataka 2009, CIRMM/LIMMS, IIS



Active air-jet surface
Fukuta 2006, CIRMM/LIMMS, IIS

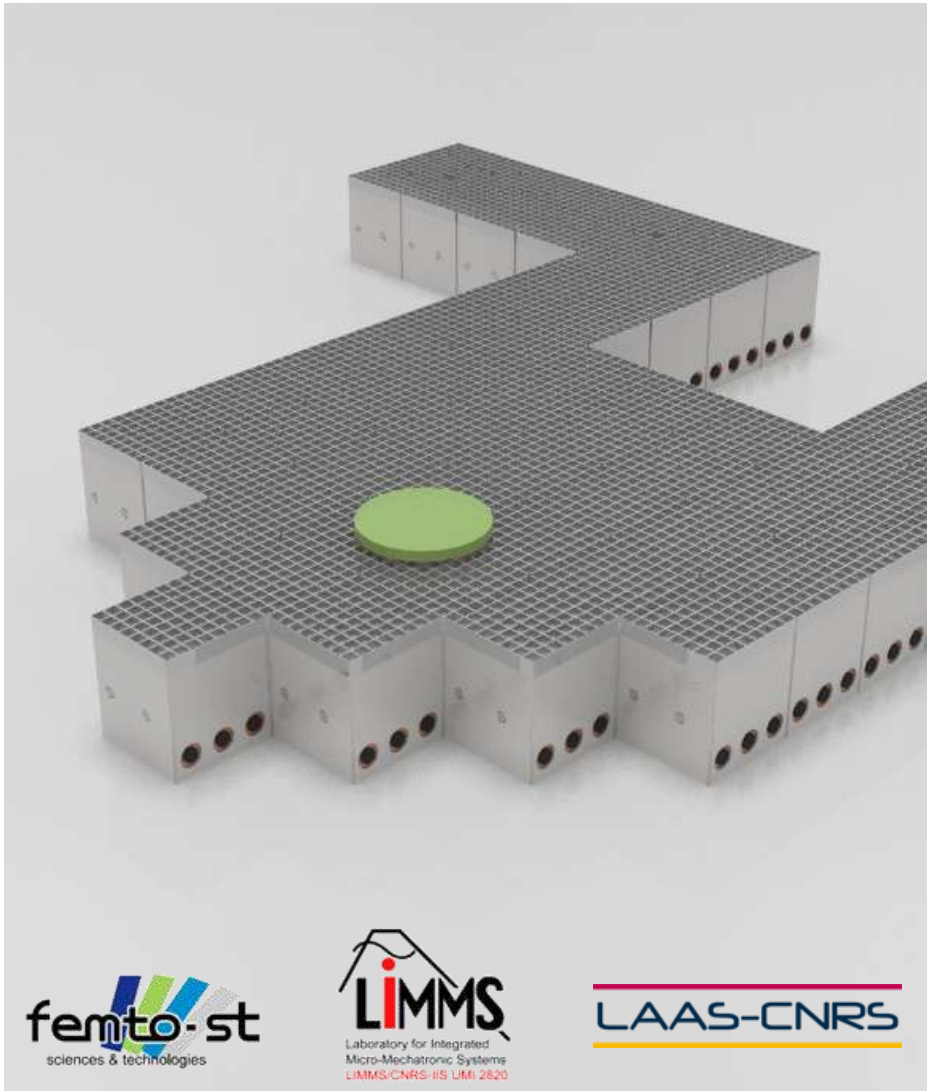


Passive air-jet surface
Zeggari 2010, FEMTO-ST

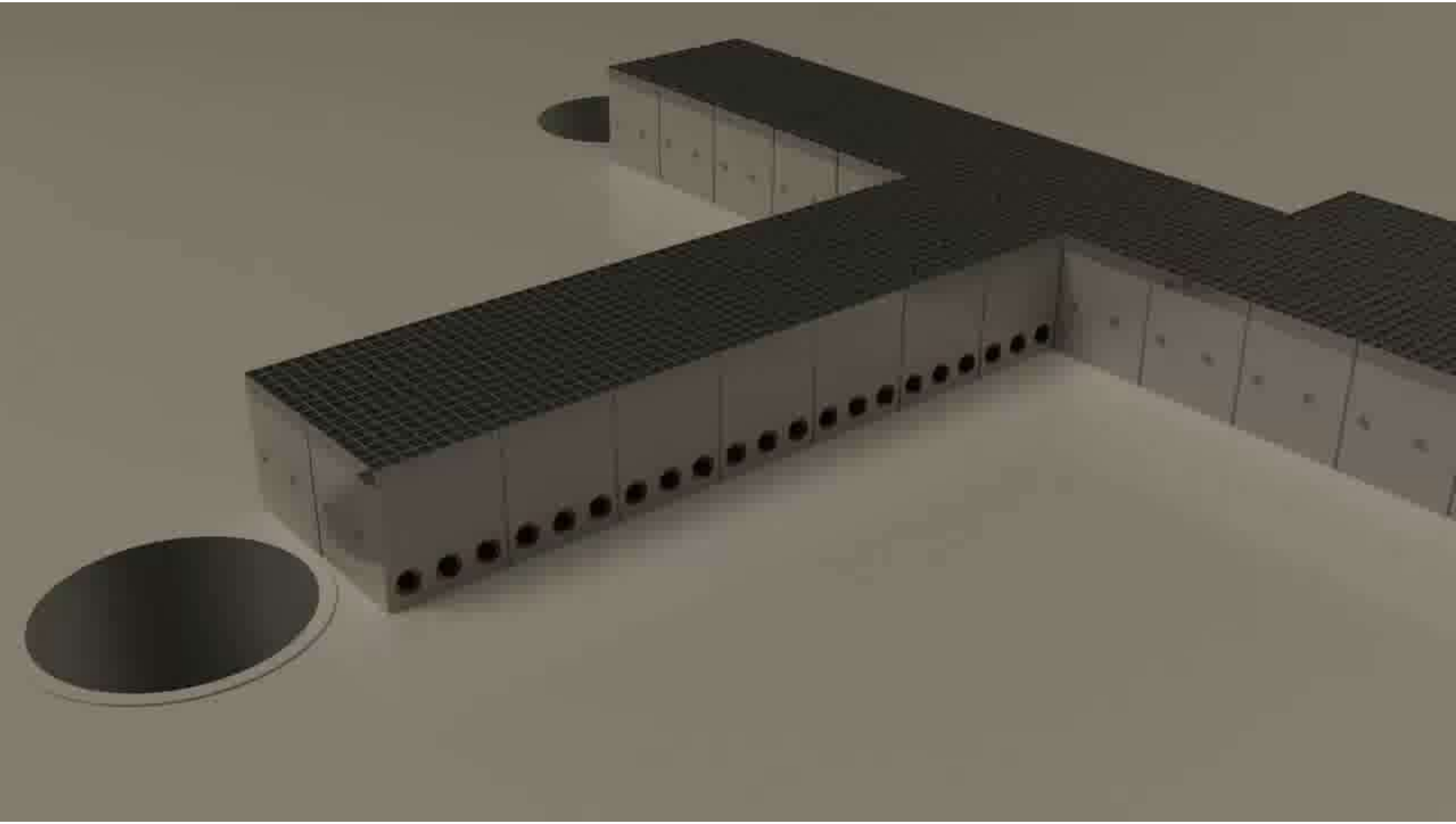


Potential air-flow surface
Delettre 2011, FEMTO-ST

- **Smart Blocks project**
  - Self-reconfigurable modular system
    - Moving
    - Processing
    - Communicating
  - Able to transport and sort small objects
    - Sensors/Actuators on the top.
  - Dynamically reactive to events
    - Create a new conveying path,
    - Replace a faulty block.



Example of application



Objectives of this paper

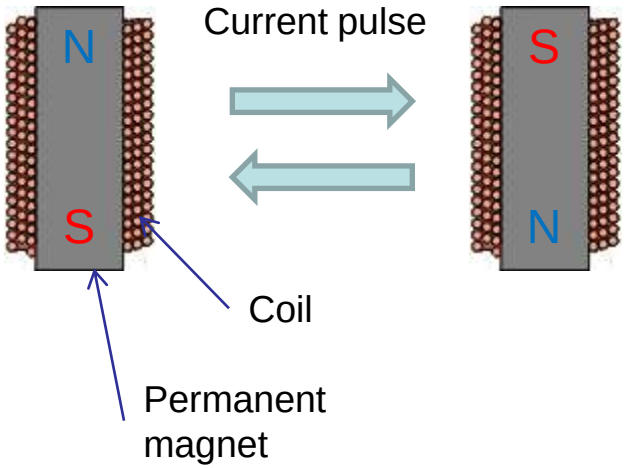
- **Presenting two aspects of the Smart Blocks project**
- **Multidisciplinary work**
  - FEMTO-ST/AS2M and FEMTO-ST/DISC
- **Actuation/locking system of the blocks**
  - 2 functions in one hardware
  - Use of electro-permanent magnets
- **Reconfiguration algorithm**
  - Distributed
  - Scalable
  - Based on the sliding actuation

Toward a 2D Modular and Self-Reconfigurable Robot for Conveying Microparts

Block actuation

Block actuation

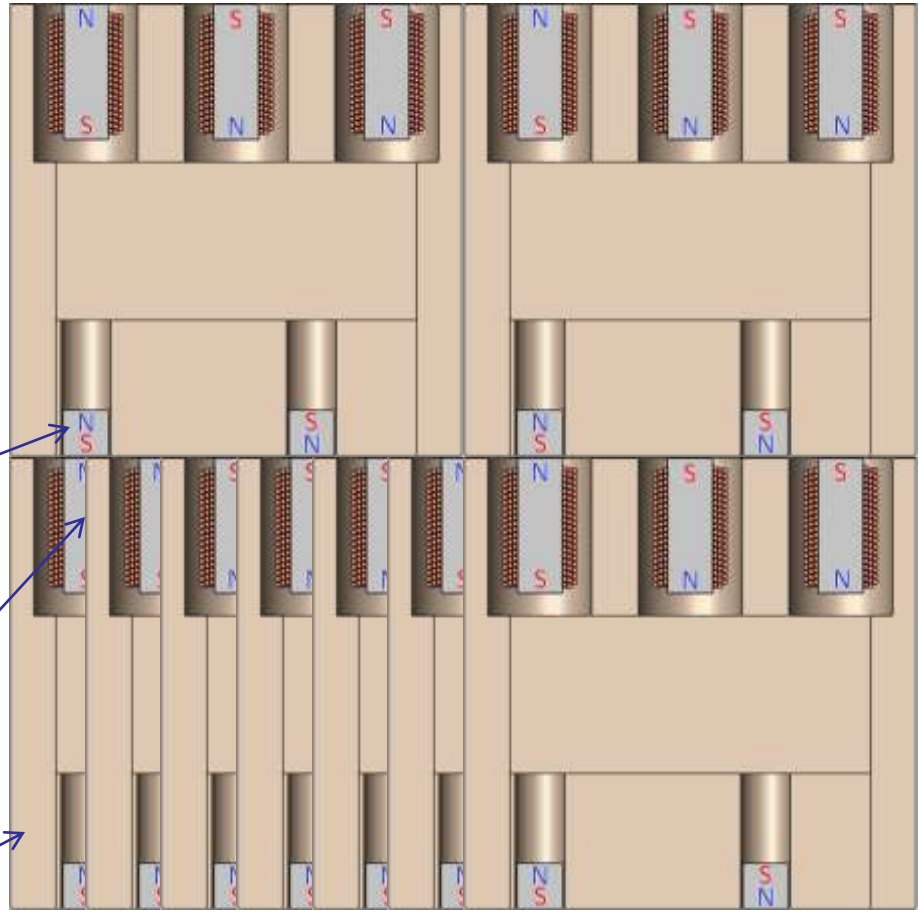
- **Electro-permanent magnets**
  - Permanent magnet + Coil
- **Permanent magnet = AlNiCo**
  - Aluminium, Nickel, Cobalt
  - High remanence
  - Weak coercive field
  - Changeable polarity



- **Features**
  - Core : 1 mm in diameter, 2 mm in length
  - Coil : 60 turns
  - Switching current : 15 A (reach by discharging a capacitor)

Block actuation

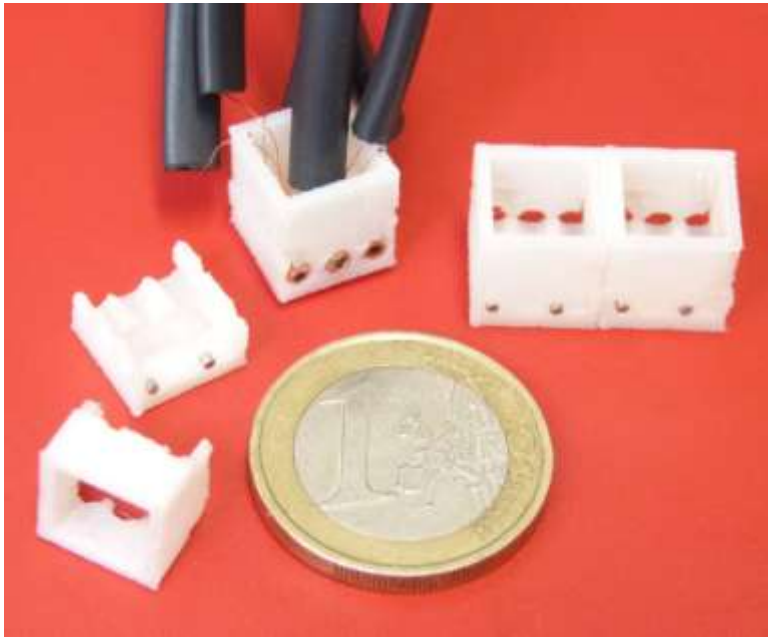
- **Linear motor principle**
  - One cycle = one move from one block to another = 6 states
- **Driving electronic**
  - dsPIC microcontroller
  - 100µF capacitor
  - H-bridges using MOSFETs

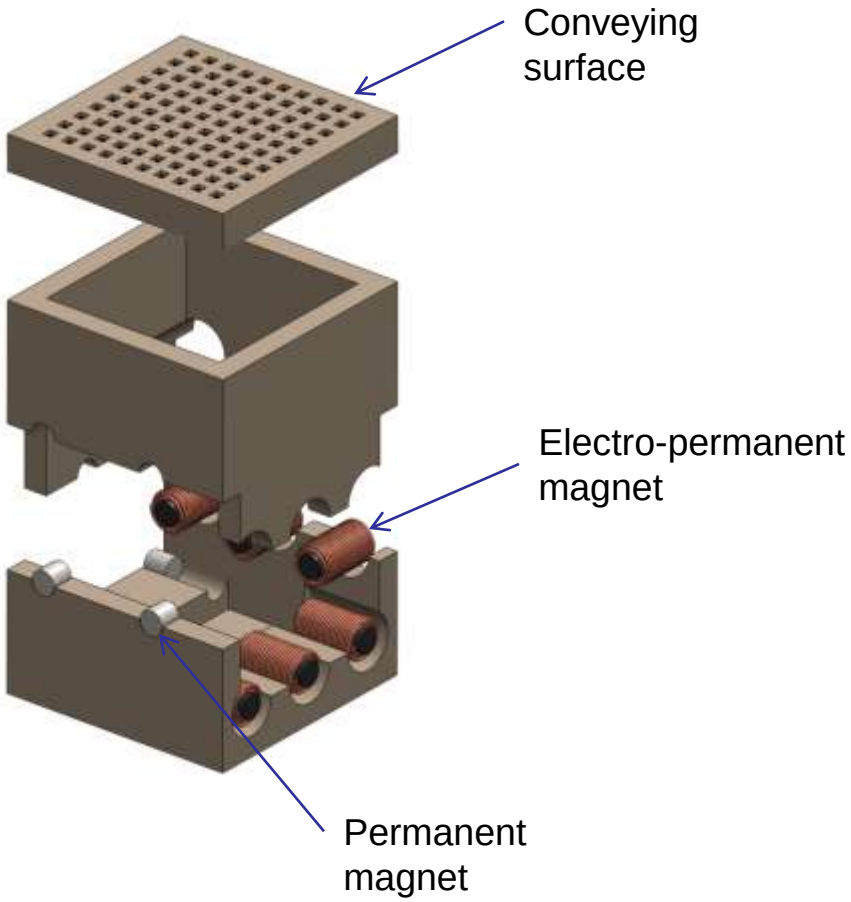


Top view of three blocks

Block design

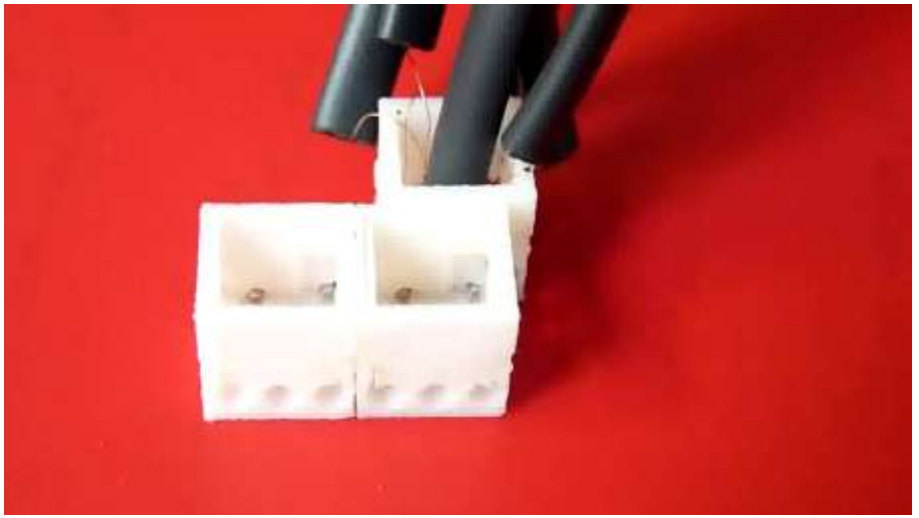
- 10x10x10 mm cube
- Two linear motors
- Rapid prototyping machine

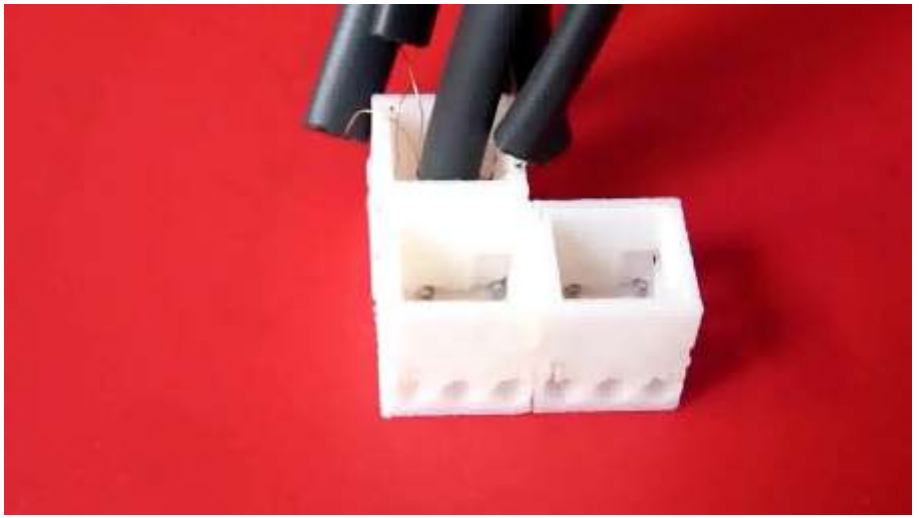




Experimental results

- Mean speed : 14 mm/s
- Holding force (power off) : 40 mN
- Pulse duration : 25 µs is sufficient
- Voltage : 15 V is sufficient





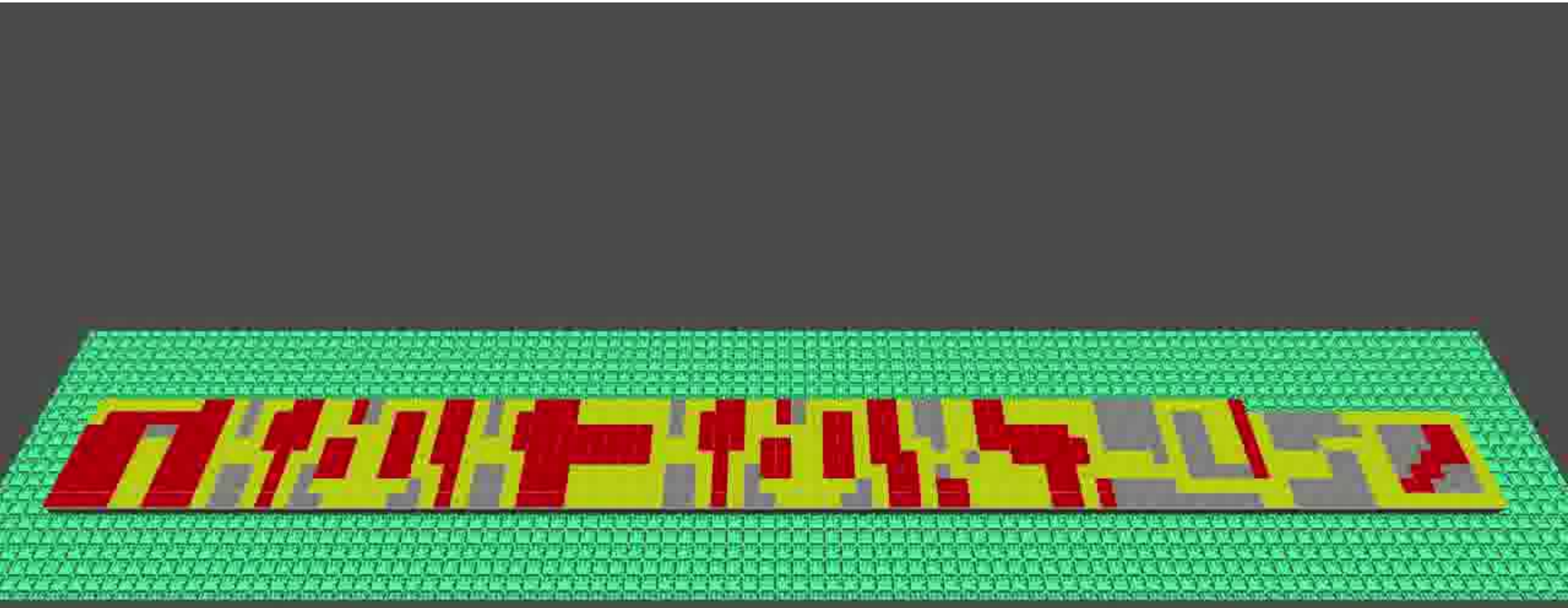
Toward a 2D Modular and Self-Reconfigurable Robot for Conveying Microparts

Block reconfiguration algorithm

Block reconfiguration algorithm

- **Goal of the reconfiguration process**
  - Blocks are placed in an initial configuration.
  - They have to move to a desired configuration.
- **Operating principle of the blocks**
  - Each block runs the same code.
  - Each block has local information only
    - State variables,
    - Neighbor blocks,
    - Local map of the final configuration.
  - Each block is connected to 1 to 4 other blocks.
    - Exchange messages
- **The Master Block**
  - A block is selected to manage the stages of the algorithm.
  - A the beginning of the process, it has the map of the goal configuration of blocks.

Example of reconfiguration

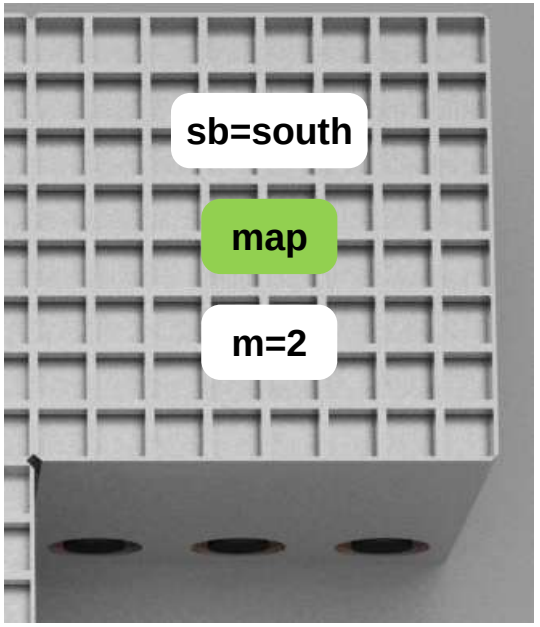


Block reconfiguration algorithm

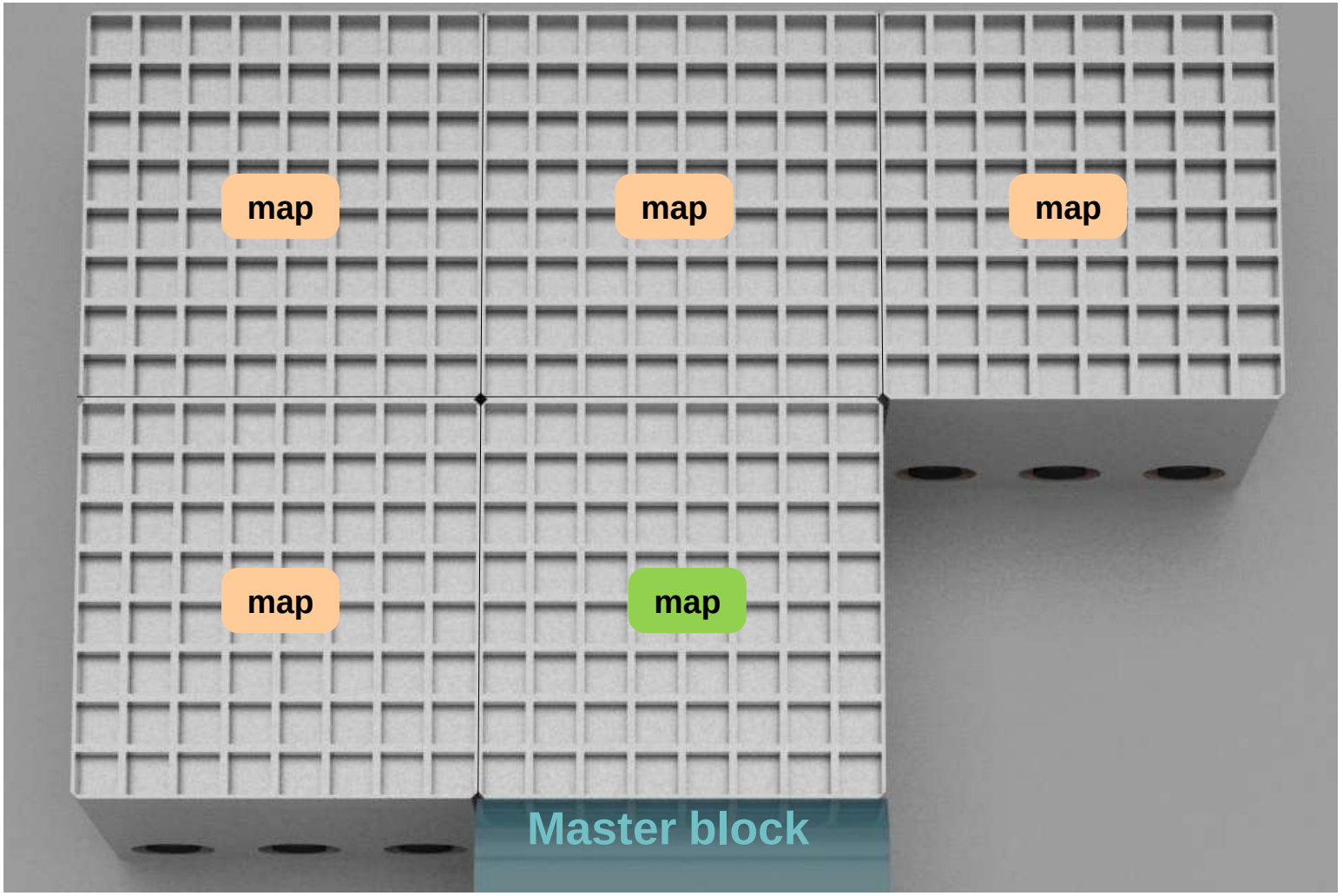
- **The 4 major stages of the algorithm**
  1. Preliminary step : sending the goal map to each block
     - To give the locations of free places to all blocks
  2. Calculation of the shortest distance between each block and a free place
  3. Determination of the direction of movement of each block
     - Many blocks may move at the same time
  4. Organize the motions of mobile blocks.
     - Without collision
- **If the goal is not reached, repeat from step 2**

1 : Sending the map to each block

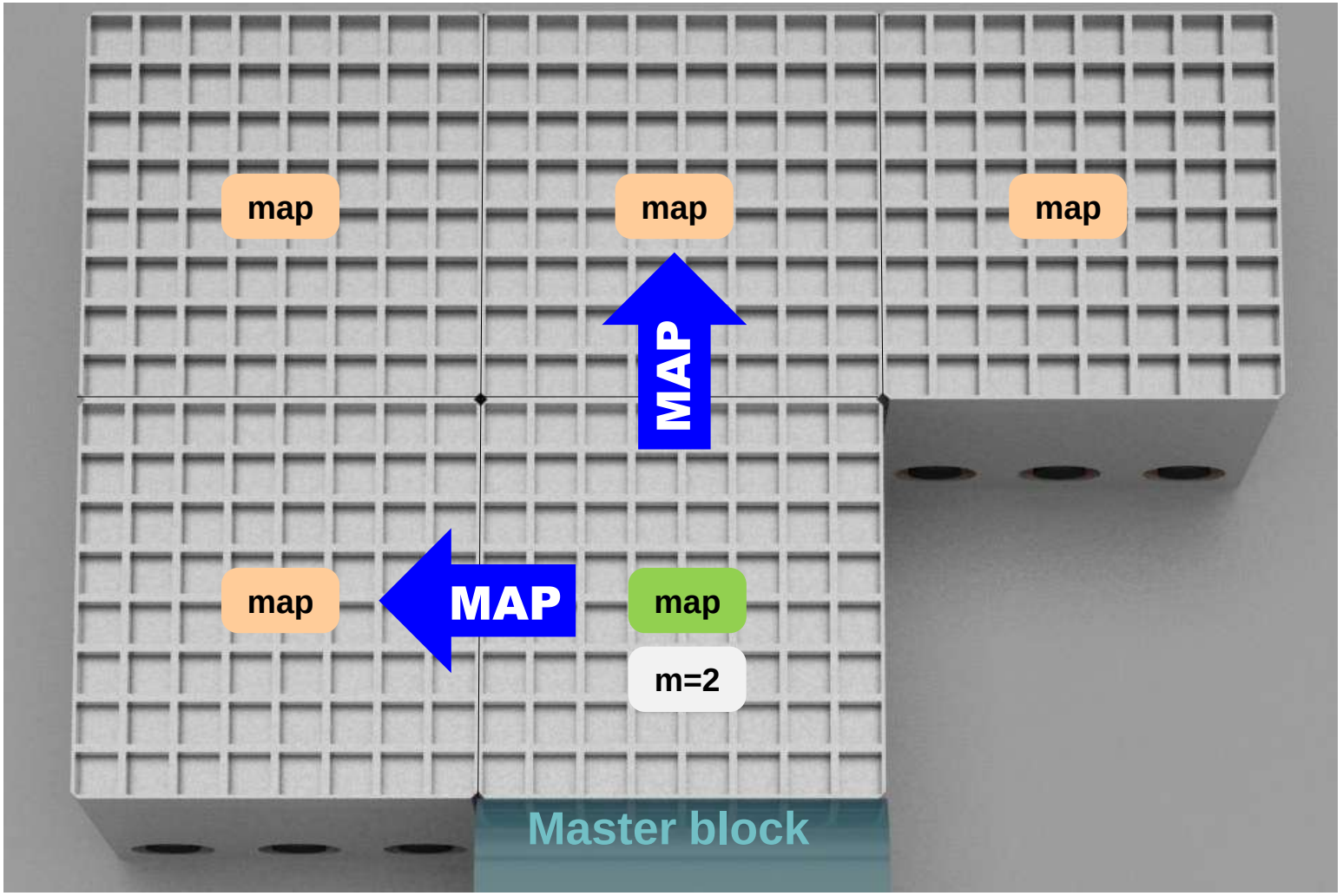
- **Sending the map to each block**
  - The master block sends the map
    - to its *m=n* neighbors
    - using a MAP message
    - Message containing the map data.
  - When a block receives a MAP message
    - If it already gets the map, it sends an ACQ message
    - else
      - It copies the map in its local memory
      - It memorizes the sender in *sb*
      - and sends a MAP message to its *m =n-1* other neighbors
  - When a block receives an ACQ message
    - It decrements *m,*
    - If all neighbors have answered (*m ==0*), it sends an ACQ message to *sb*.



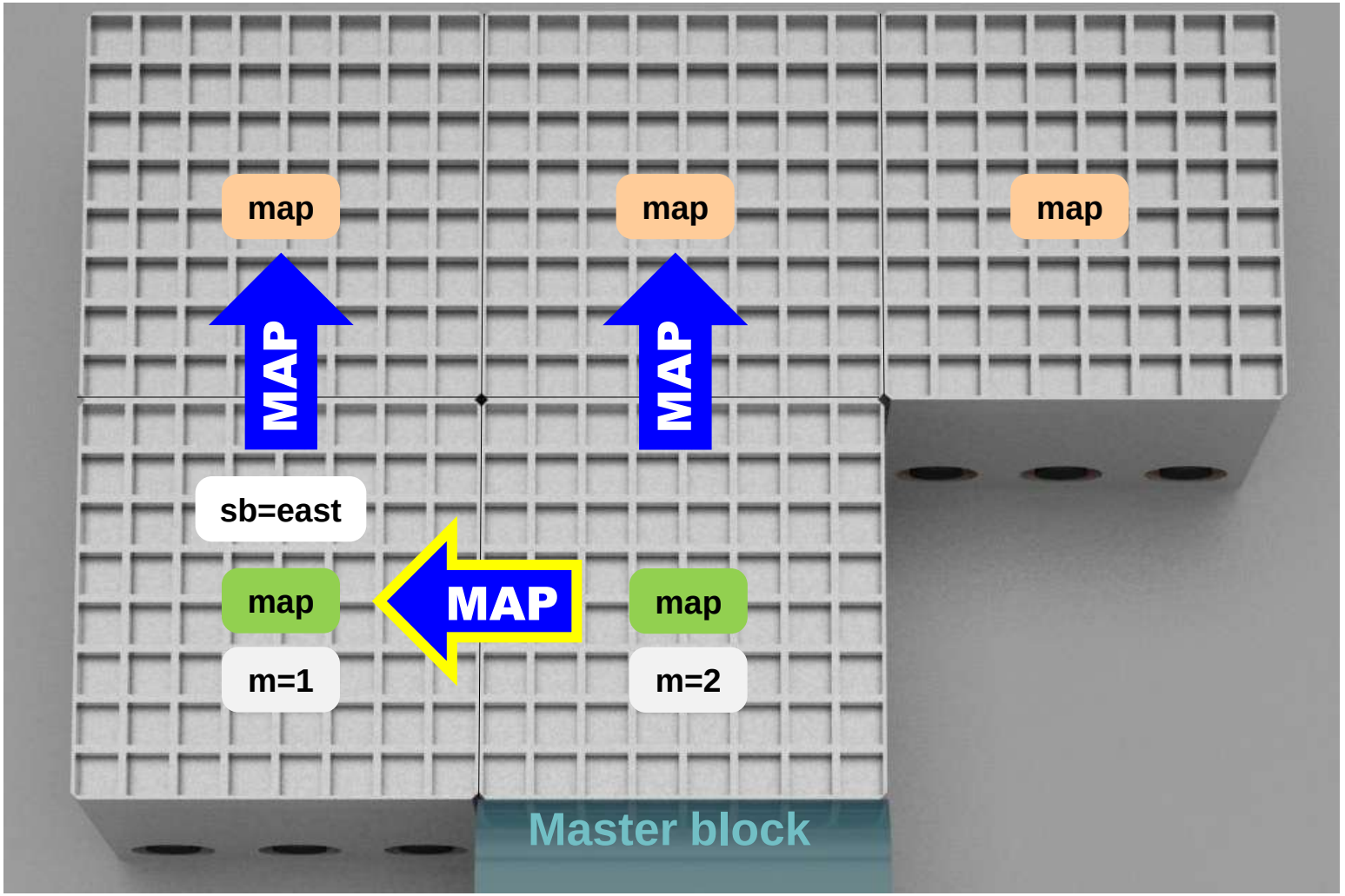
Sending the map to each block



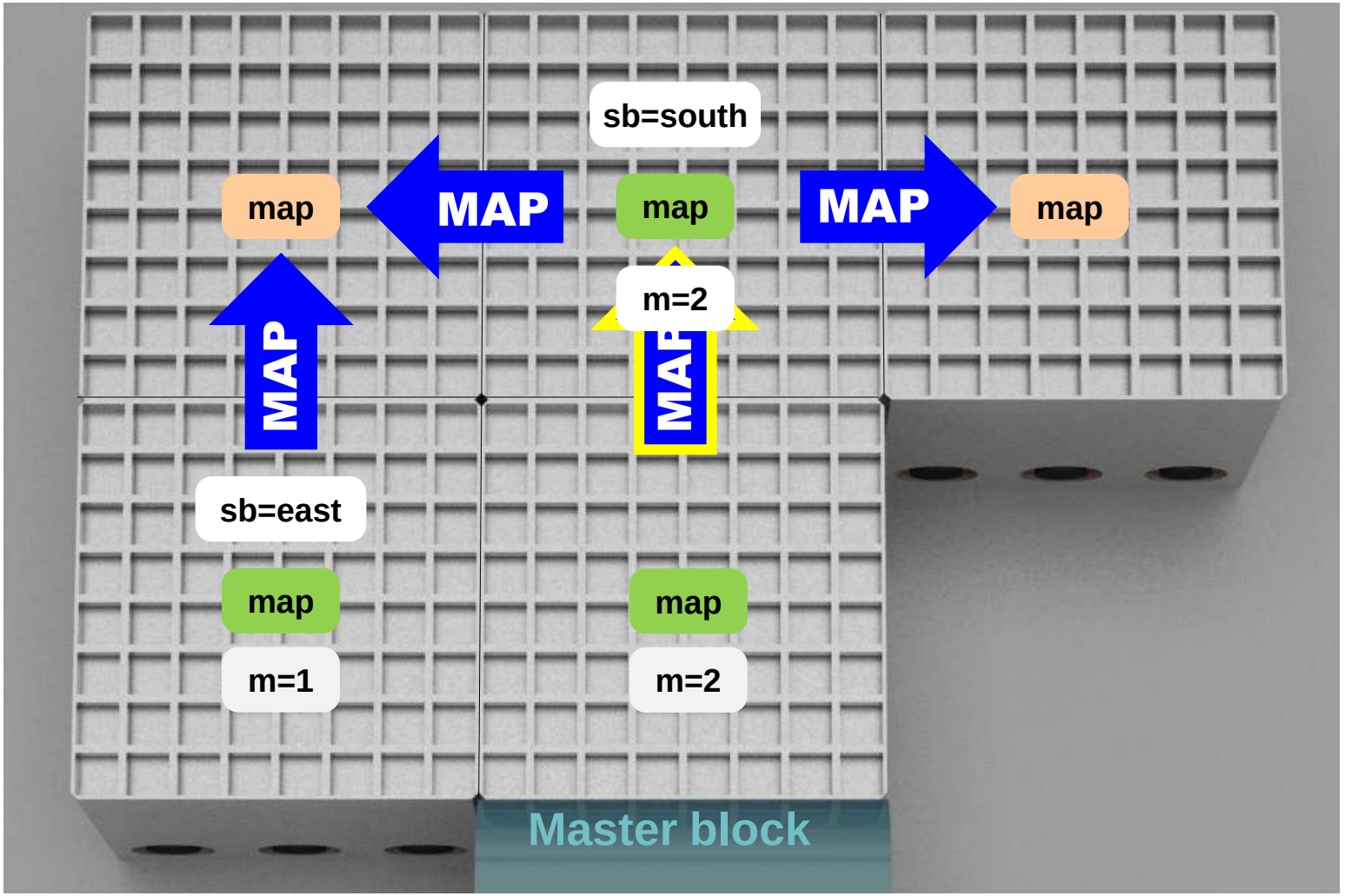
Sending the map to each block



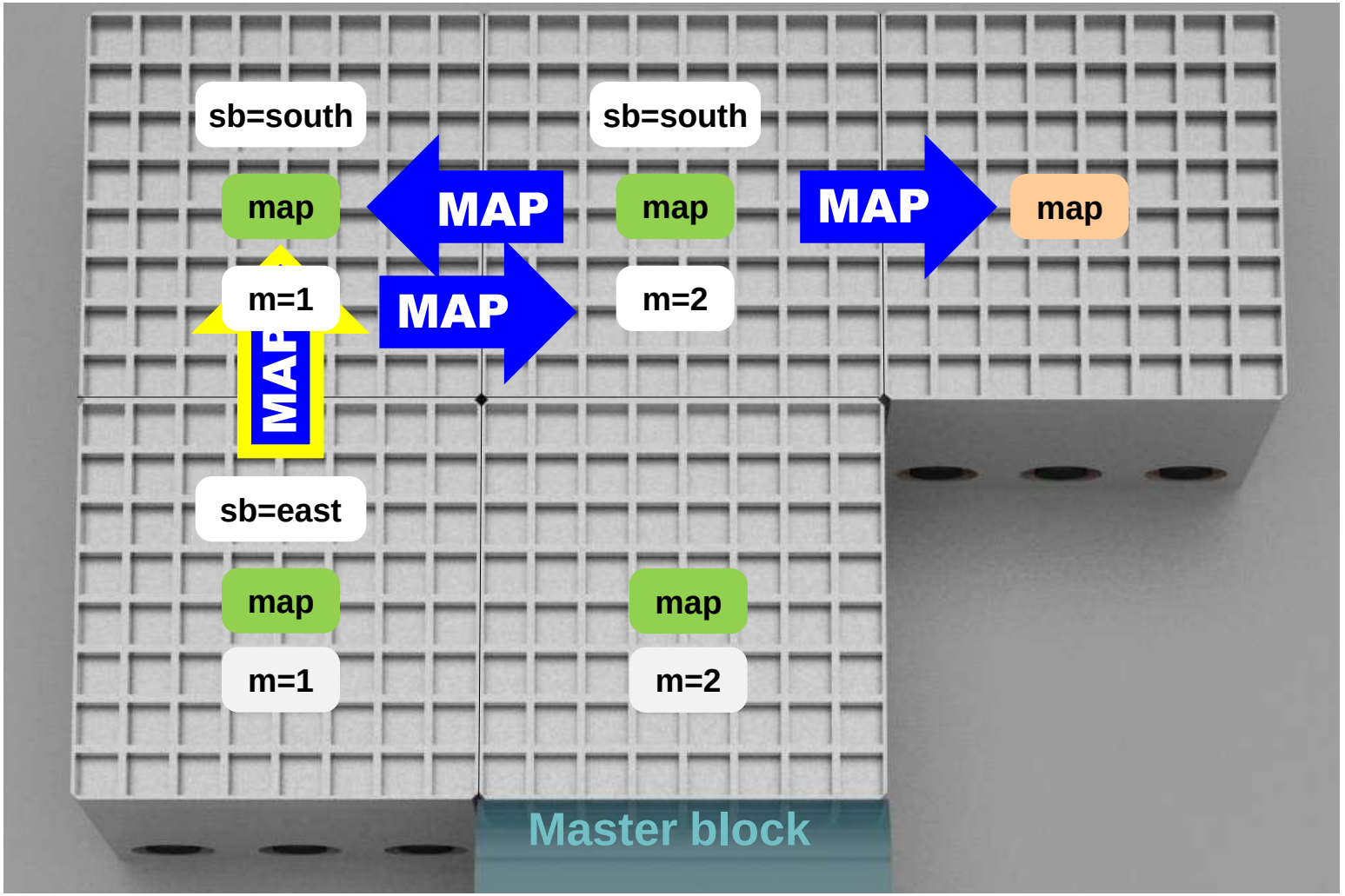
Sending the map to each block



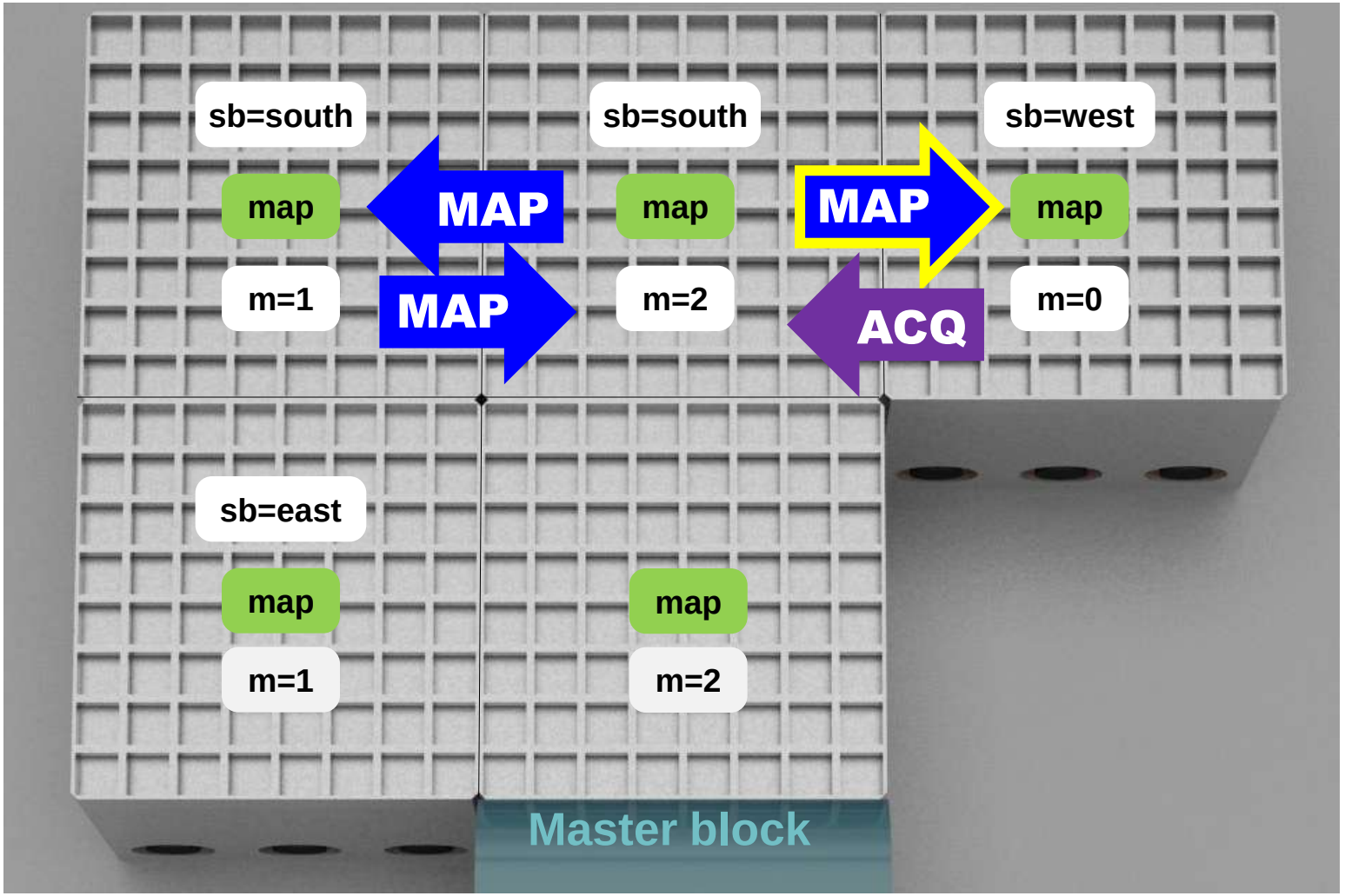
Sending the map to each block



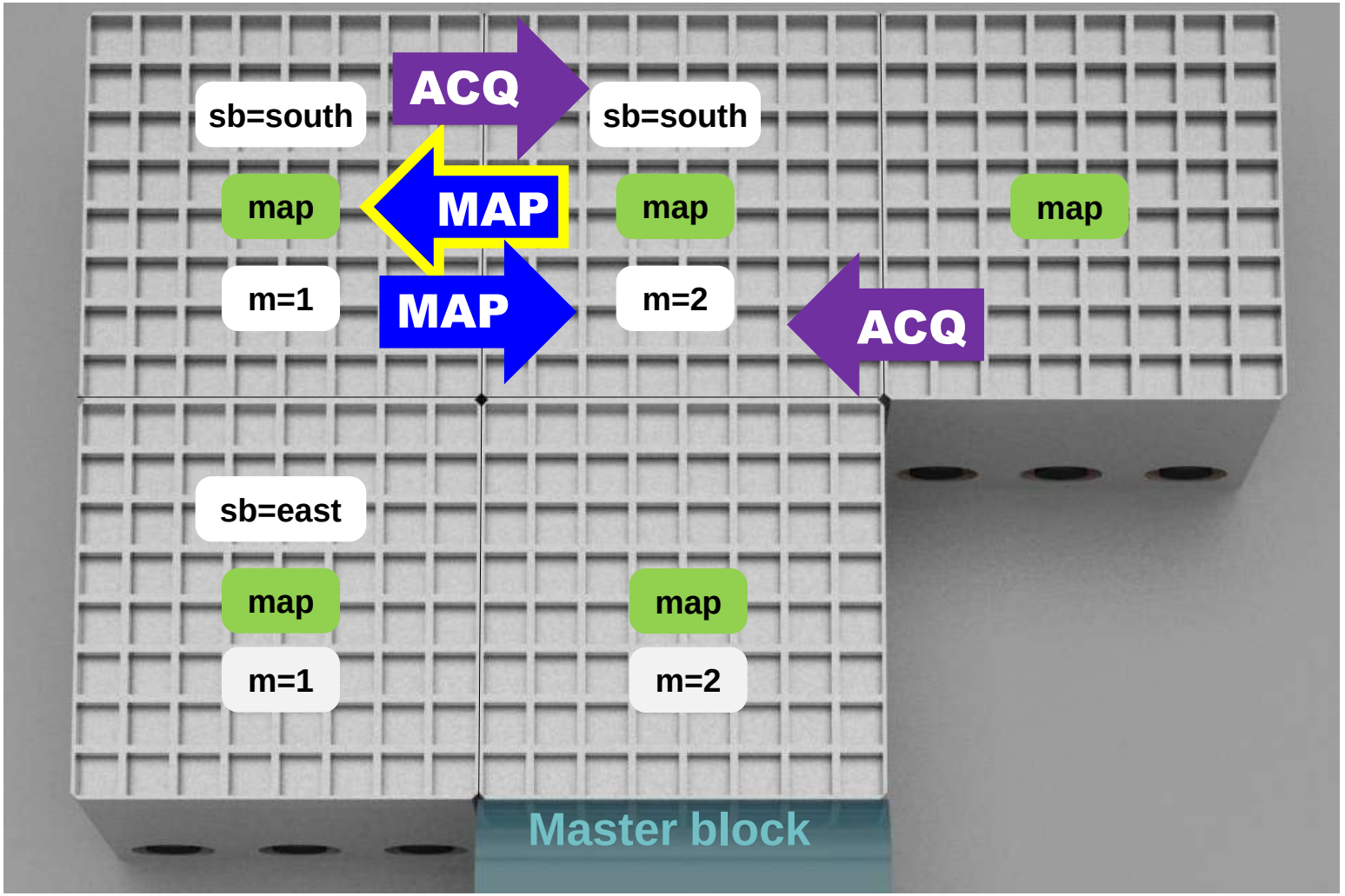
Sending the map to each block



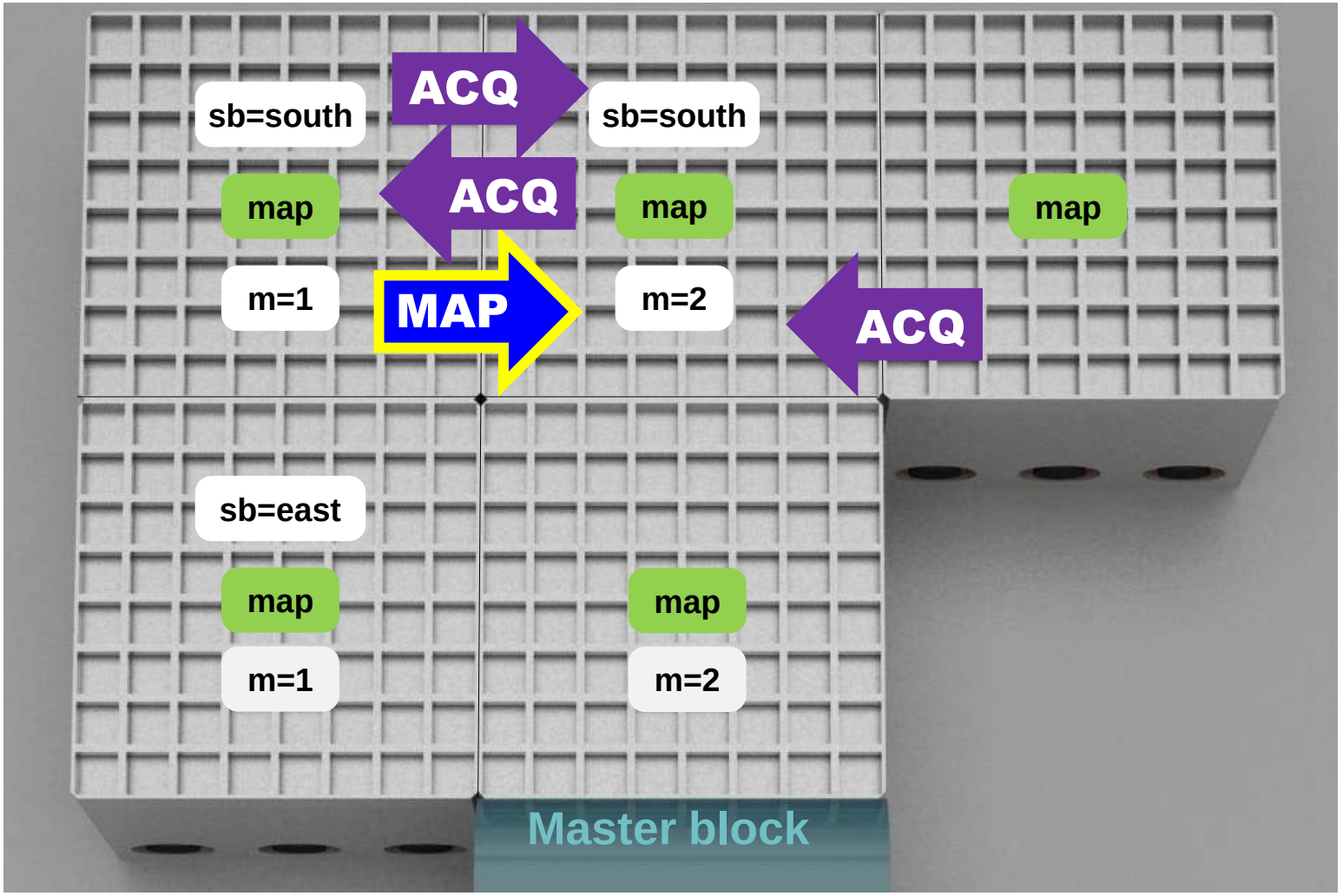
Sending the map to each block



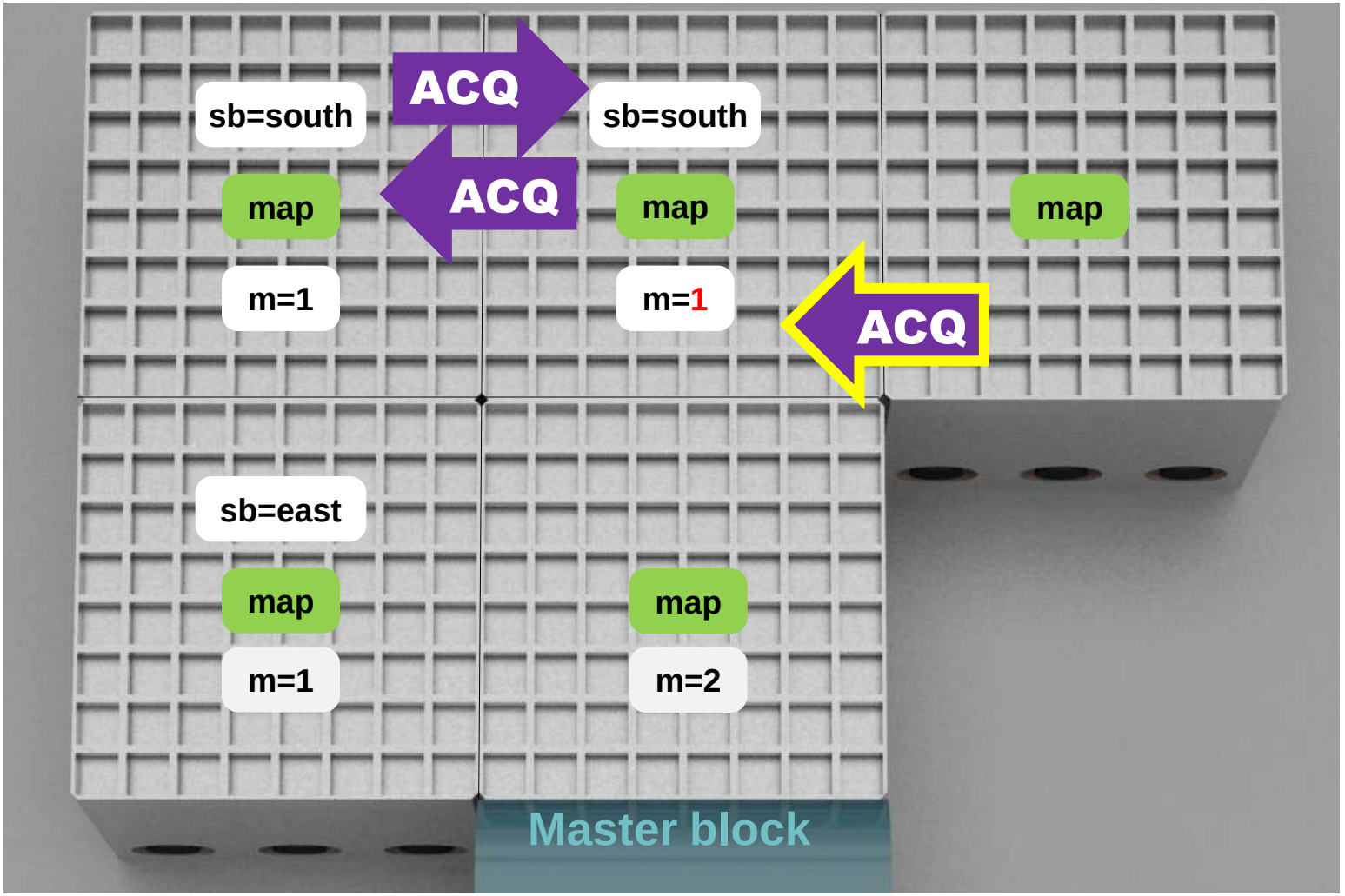
Sending the map to each block



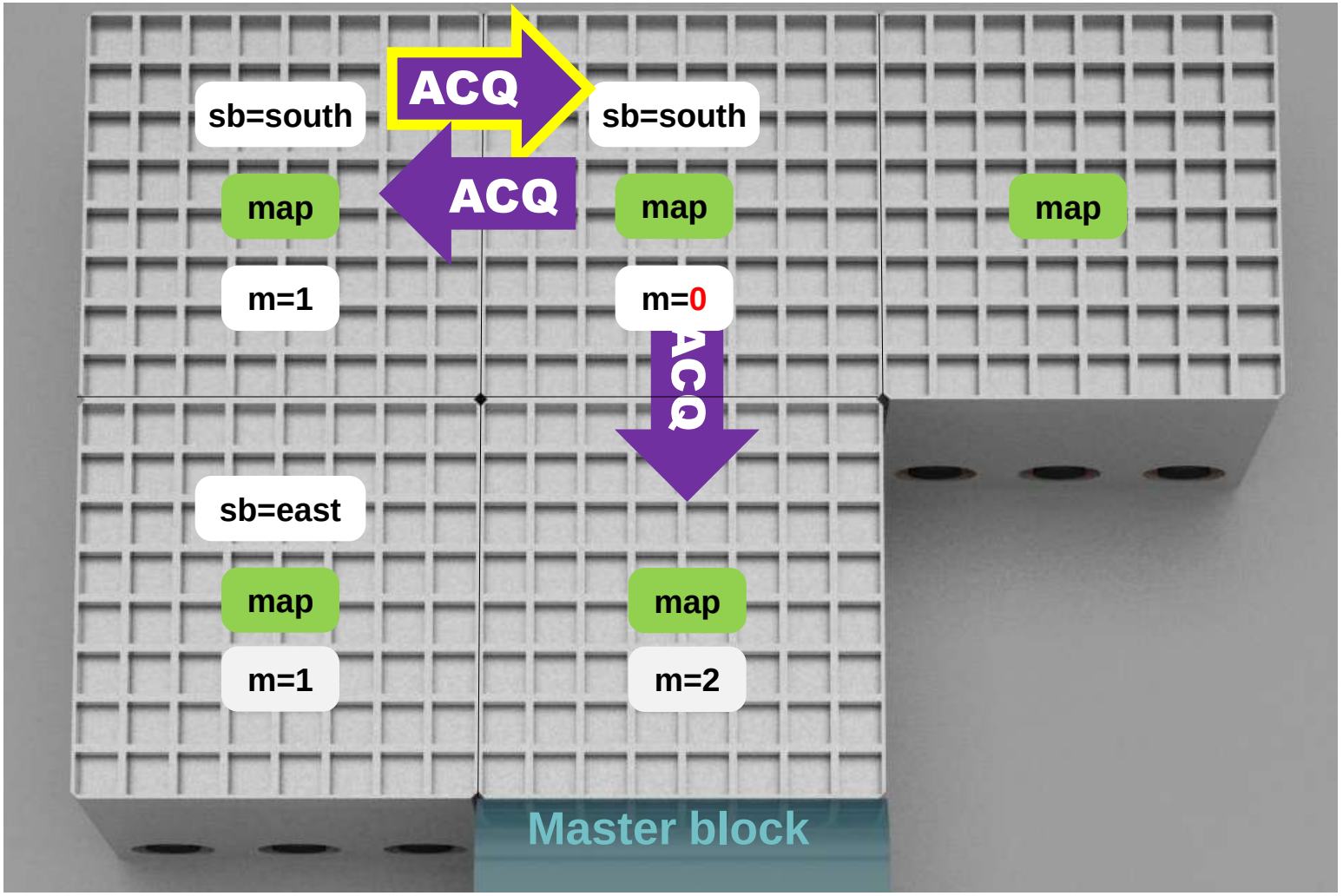
Sending the map to each block



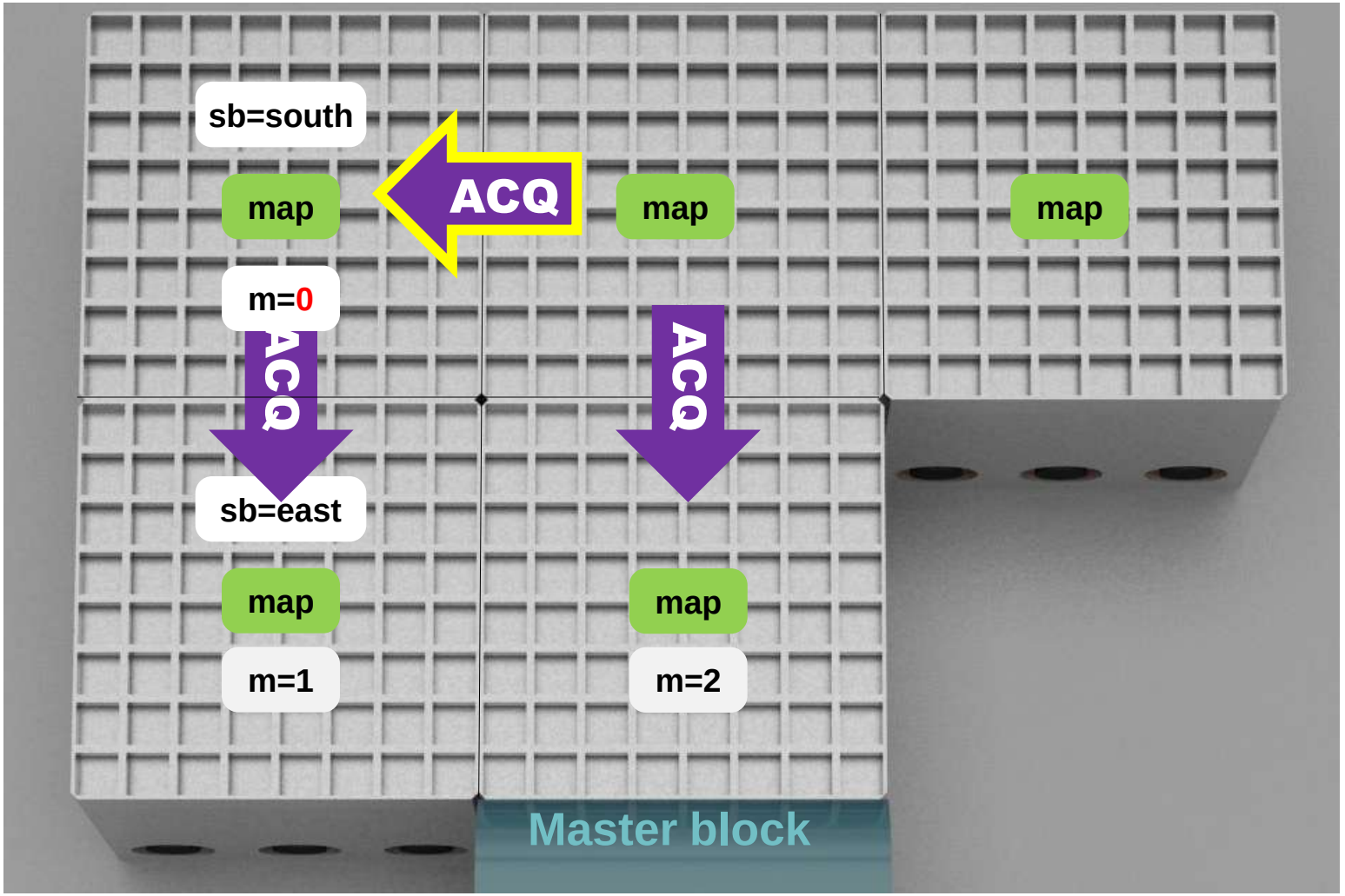
Sending the map to each block



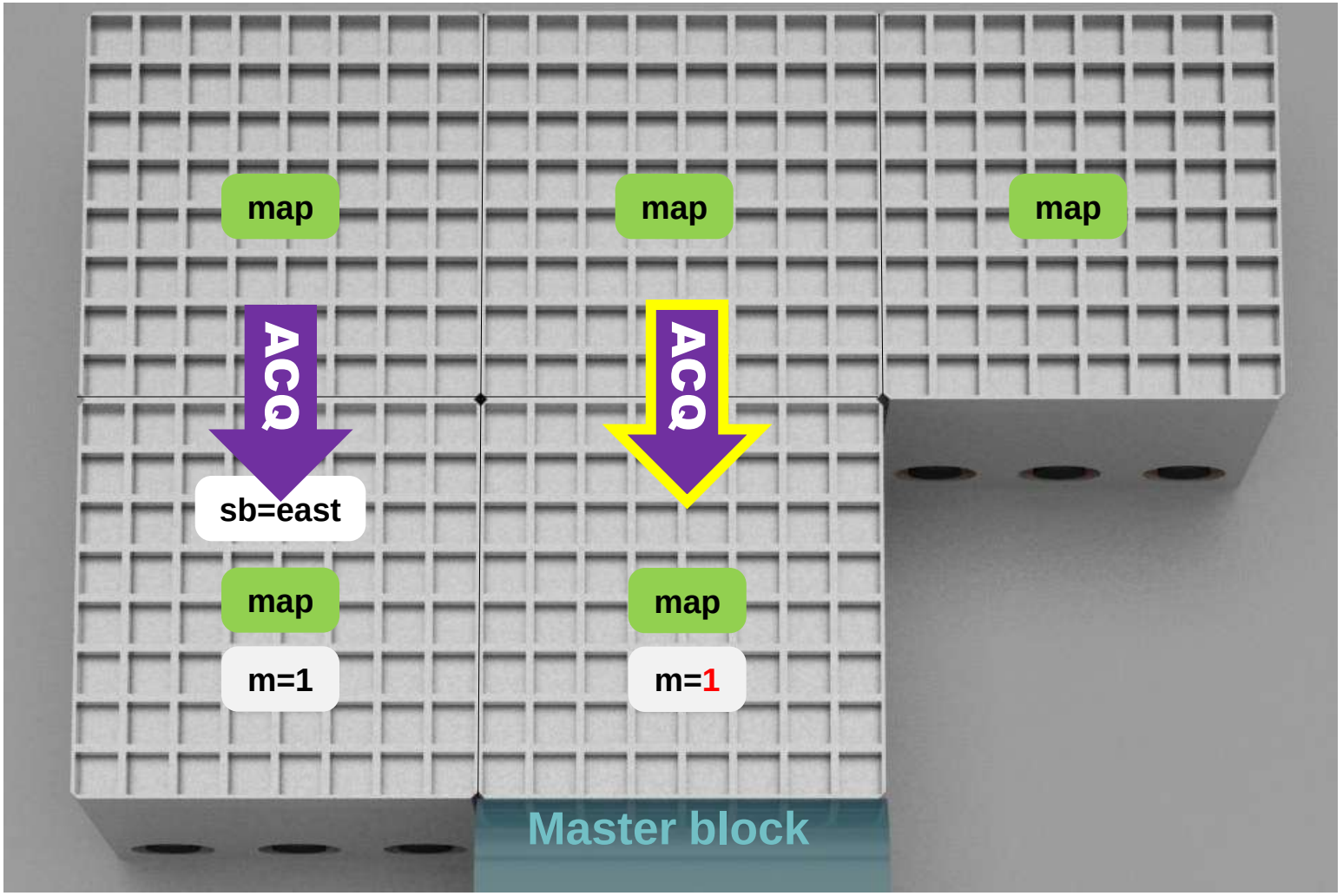
Sending the map to each block



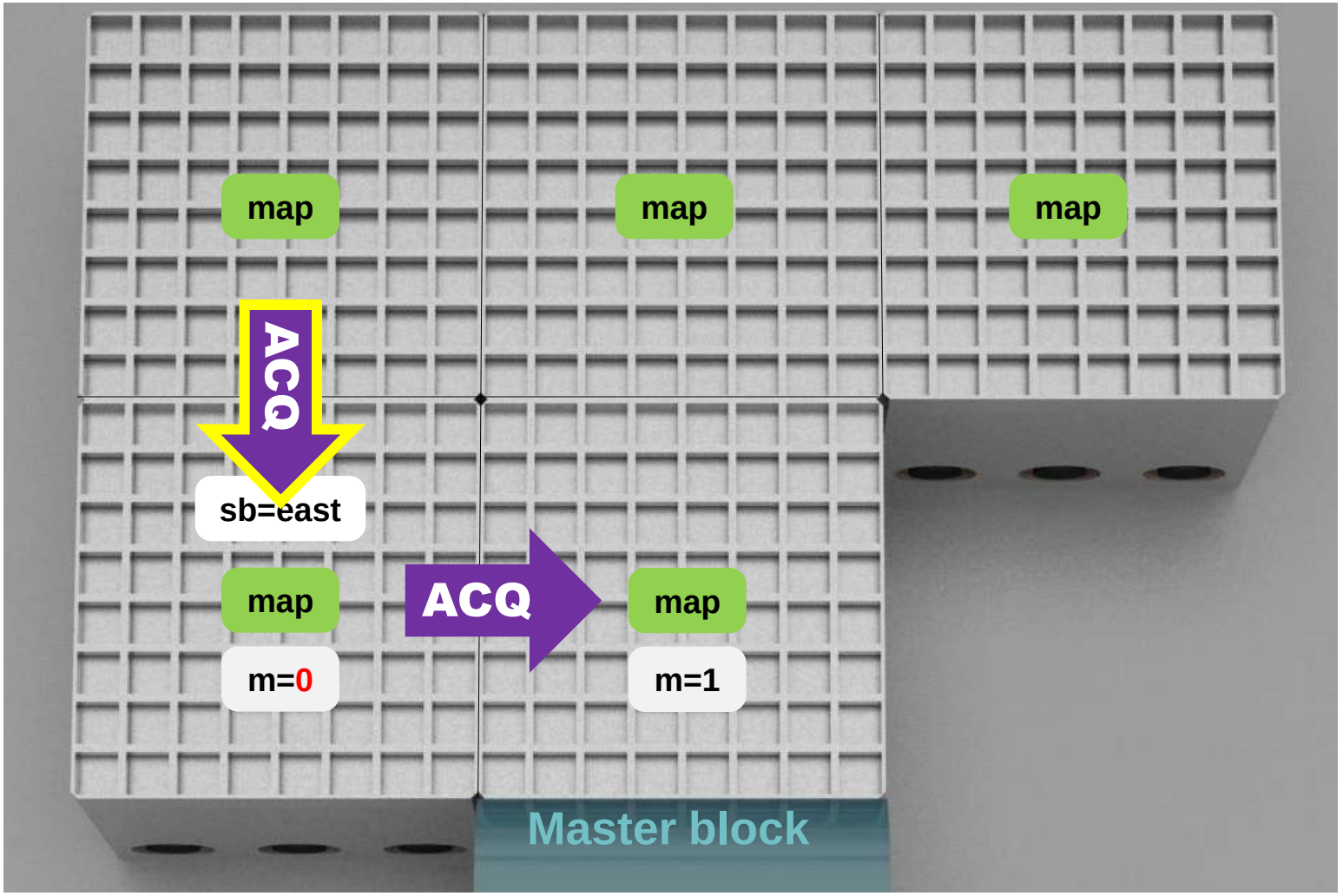
Sending the map to each block



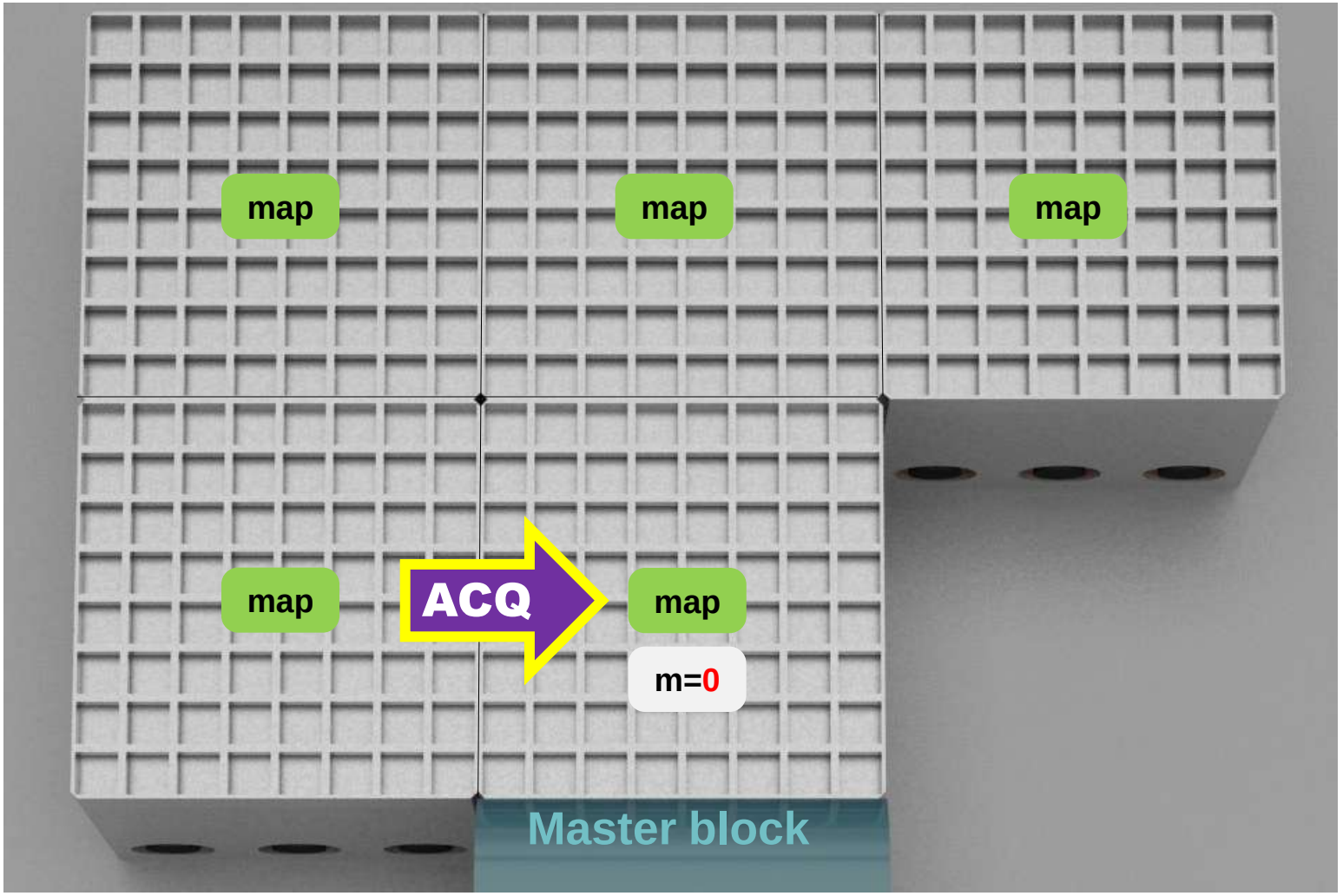
Sending the map to each block



Sending the map to each block

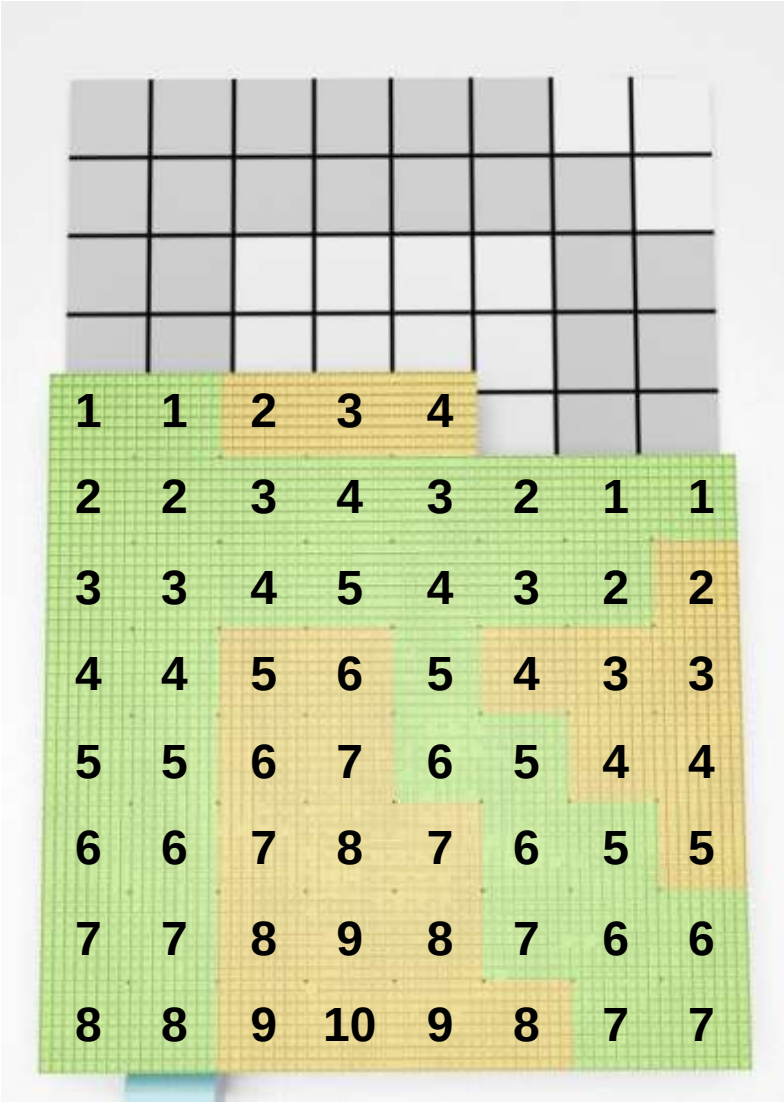


Sending the map to each block



Calculating the shortest distance

- **Calculating the shortest distance to an empty area**
  - If a block has an empty place on one of its 4 sides, its distance is 1.
  - For each other block, its distance is the minimum distance of its neighbors plus 1.
- **Example of results of distance calculation**
  - The goal configuration forms the R letter.
  - Green blocks are well placed.
  - Orange blocks have to move.

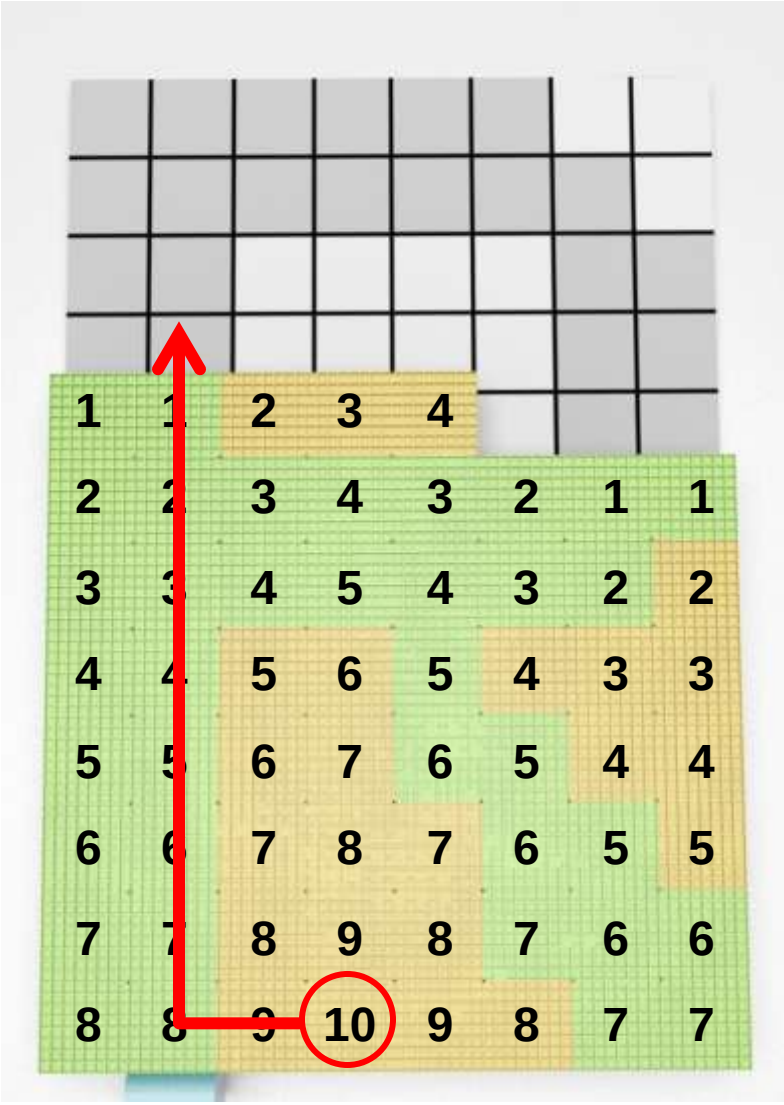


Calculating the shortest distance

- **Algorithm using messages**
  - The method is very similar to the previous one.
  - First, each block gets a distance parameter initialized to infinite.
  - The master block sends the distance requirement
    - to its *m=n* neighbors using a REQUIREMENT message.
  - When a block receives a REQUIREMENT message
    - If one of its sides touches a free place it writes 1 in the distance local variable and answers with a DISTANCE message
    - else
      - It memorizes the sender in *sb*
      - and sends a REQUIREMENT message to its $m = n-1$ other neighbors
  - When a block receives a DISTANCE message
    - It decrements m
    - It writes the minimum of its distance and the distance of the sender plus 1 in the distance variable
    - If all neighbors have answered ($m == 0$), it sends a DISTANCE message to *sb*.

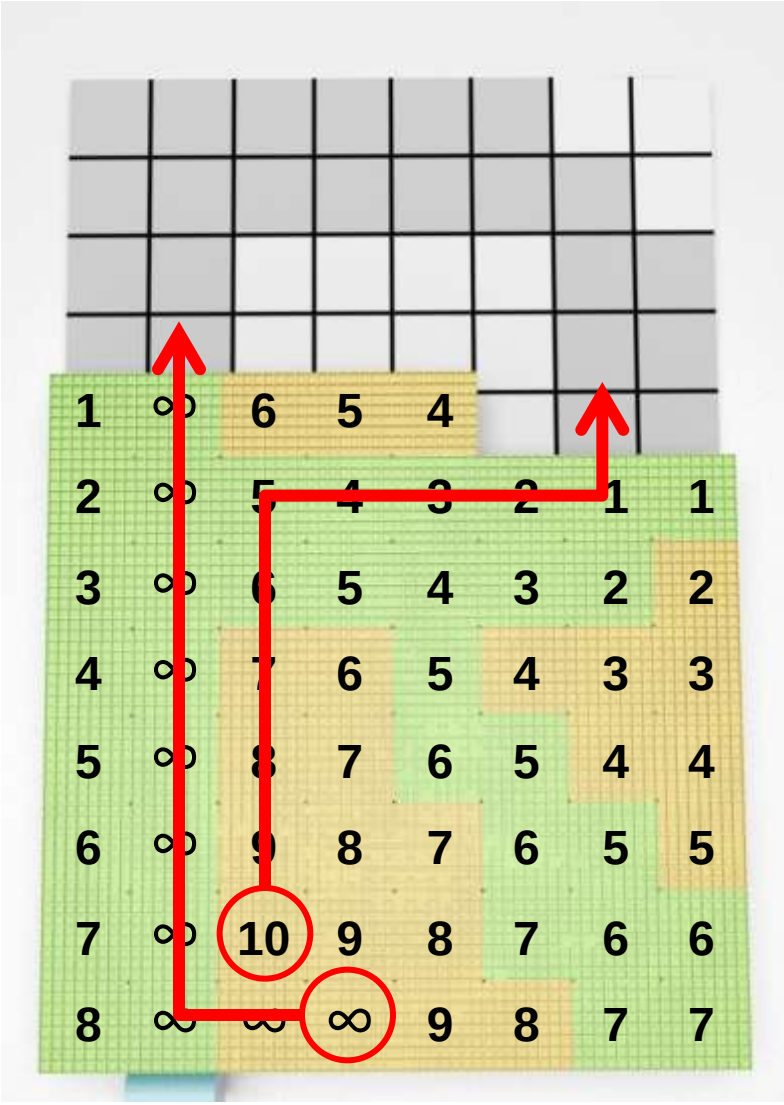
Moving blocks

- **Determining the moving blocks**
  - To reduce the global distance of blocks.
    - Converge to a solution
  - Find the block B_{max}
    - That admits the greatest distance value
    - And is in a place that must be emptied
  - Find a path from B_{max} to an empty place



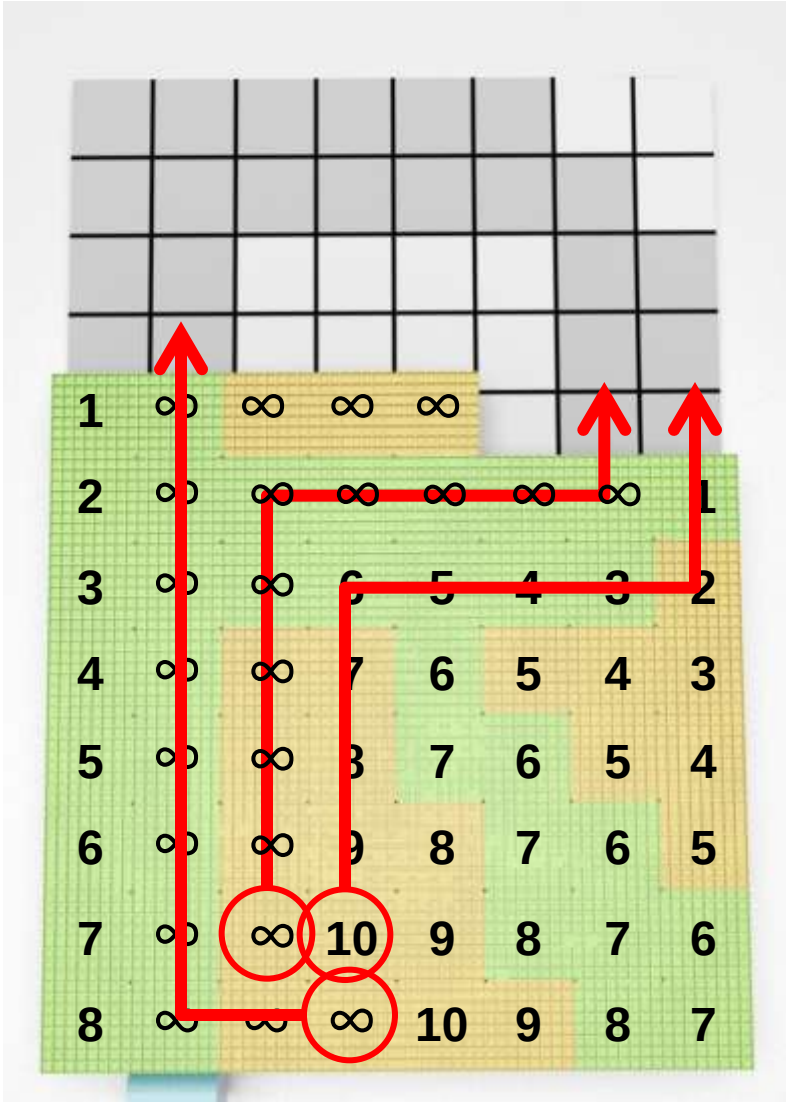
Moving blocks

- **Repeat the two last steps :**
  - The distance of cells crossed by the paths is set to infinite
  - Recalculate the distance
  - Find a new path
- **Until all distances are infinite**



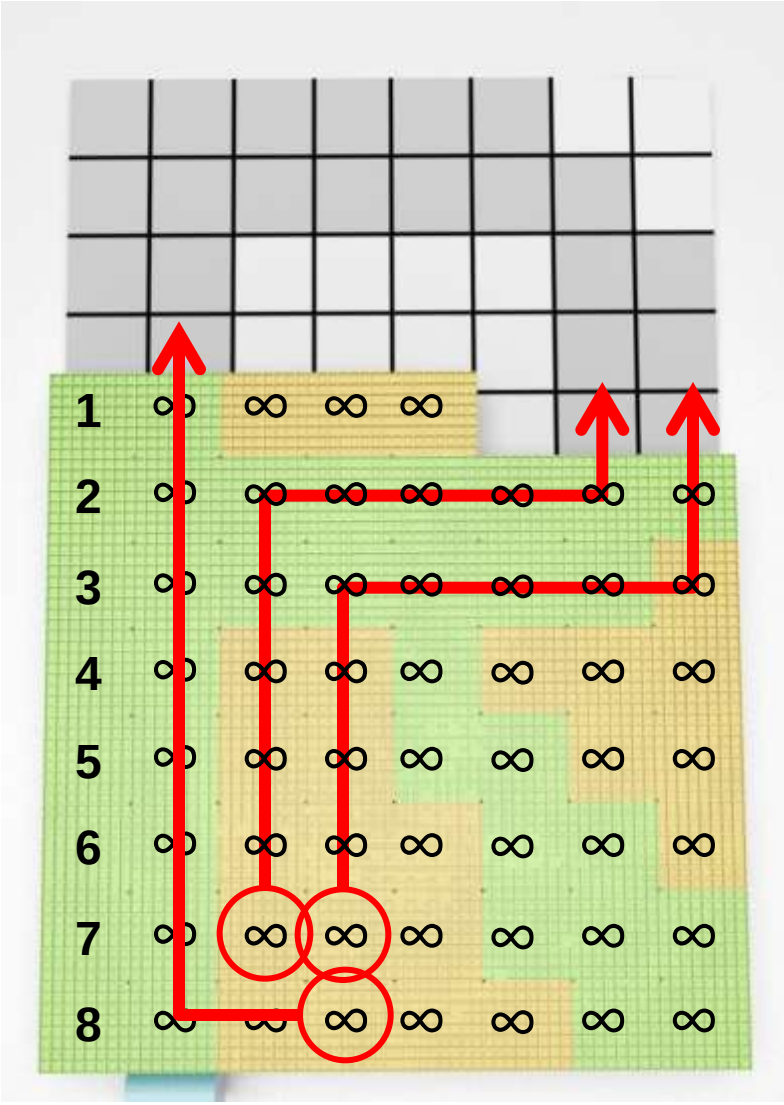
Moving blocks

- **Repeat the two last steps :**
  - The distance of cells crossed by the paths is set to infinite
  - Recalculate the distance
  - Find a new path
- **Until all distances are infinite**

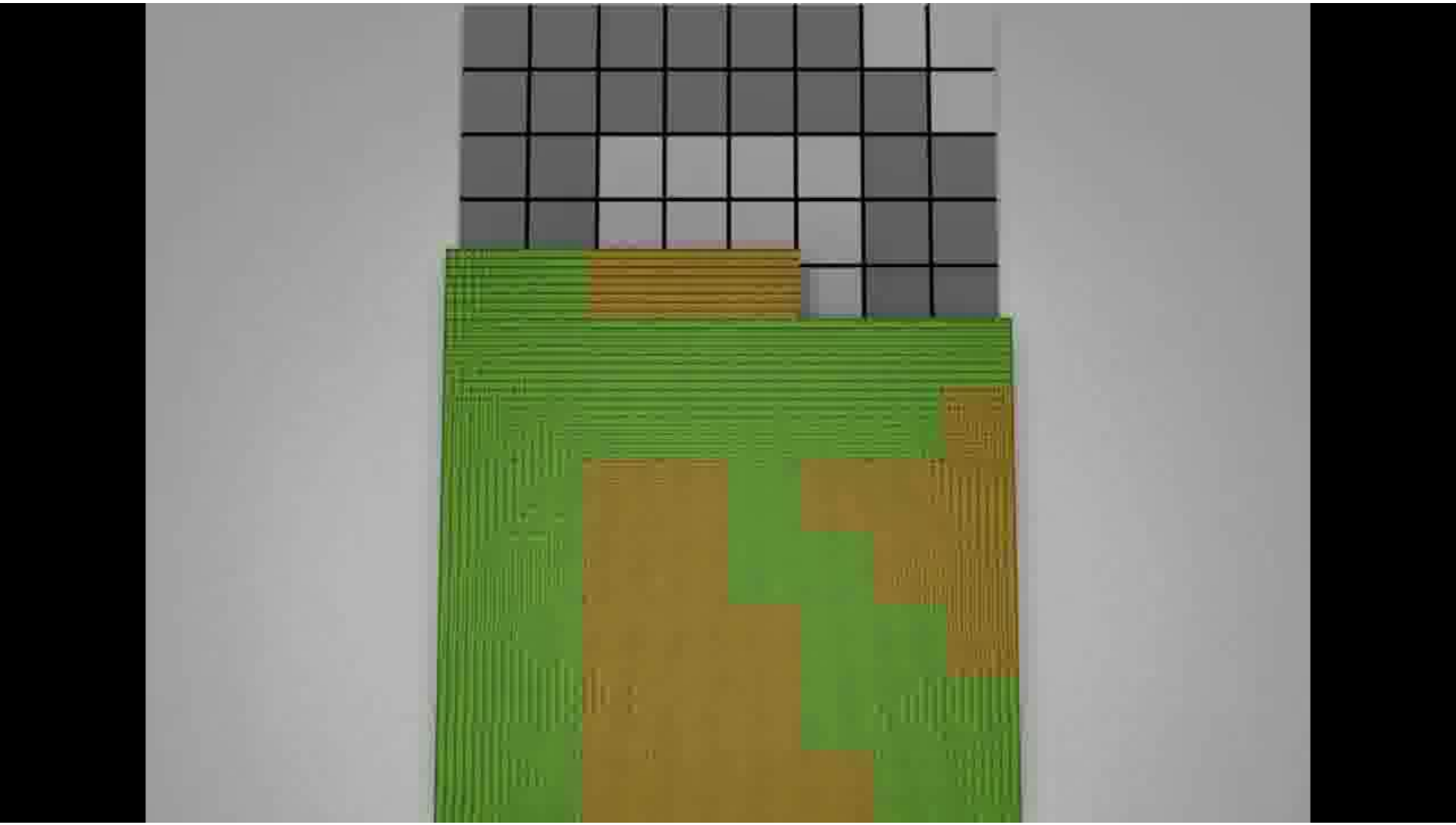


Moving blocks

- **Repeat the two last steps :**
  - The distance of cells crossed by the paths is set to infinite
  - Recalculate the distance
  - Find a new path
- **Until all distances are infinite**



Motions of blocks



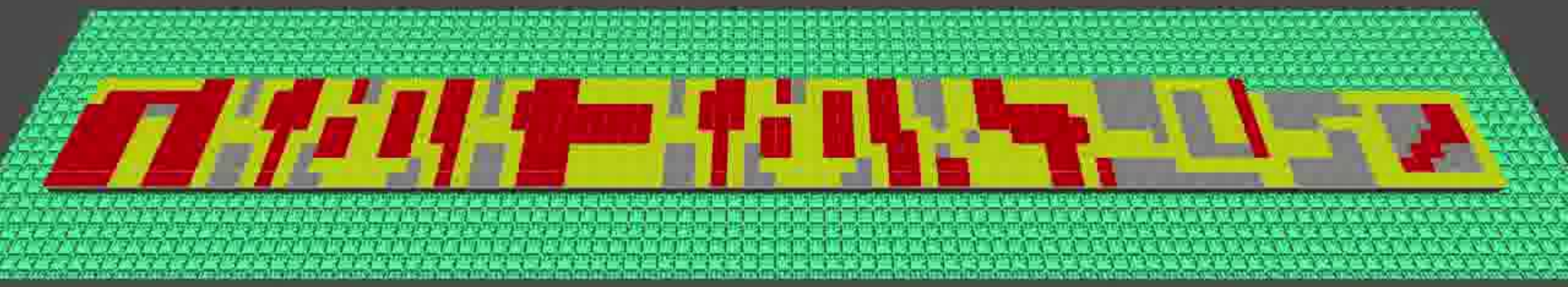
The real-time simulator

- **Discrete events simulator**
  - Deterministic
  - Is able to reproduce the exact same list of events.
- **Capabilities**
  - Blocks movements
  - Exchanges of messages between neighbor blocks.
- **Interface**
  - Free visualization of blocks
  - Selection of a block to get its state.
  - Colorization of blocks to show states of blocks

Example of reconfiguration

Initial configuration : horizontal line

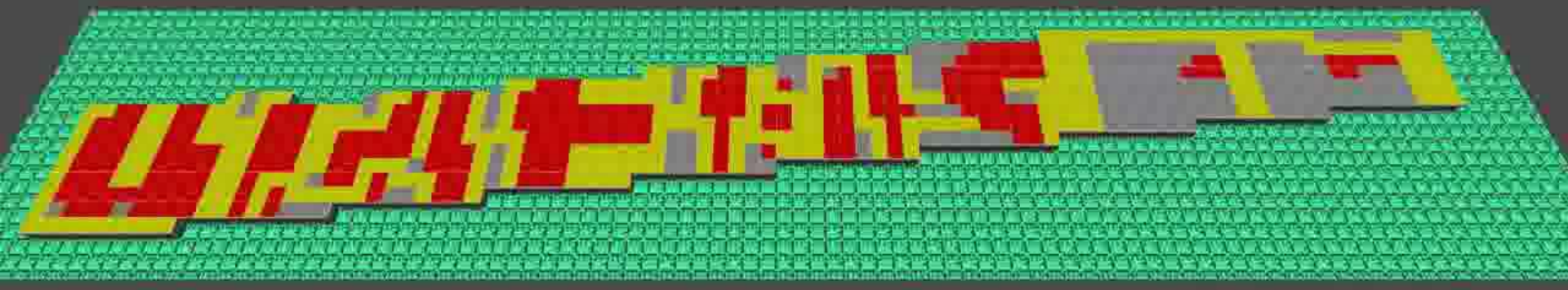
706 blocks / 66 steps



Example of reconfiguration

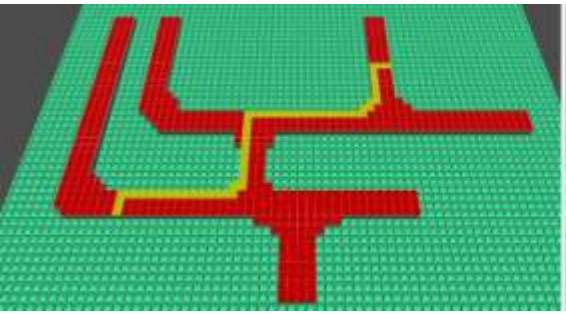
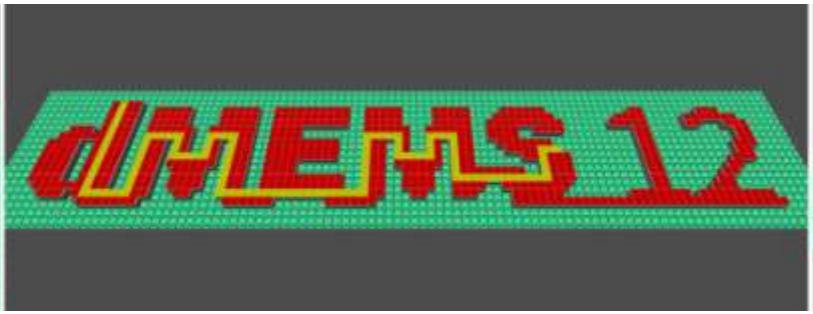
Initial configuration : diagonal line

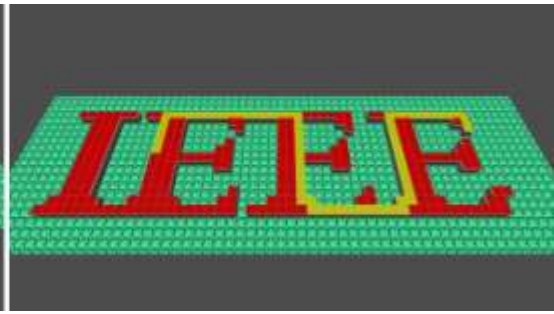
706 blocks / 65 steps



Some results

- **The speed of convergence of the algorithm to reach a configuration is highly dependent on the initial configuration.**



Initial configuration \ Final configuration	Conveyor (546 blocks)	dMEMS 12 (706 blocks)	IEEE (281 blocks)
Horizontal line	121	66	23
2 horizontal lines	136	64	18
Diagonal line	108	65	27
Sinusoïdal line	71	71	21
2 boxes	132	177	67

Future works

- **Block actuators**
  - Increase the size of blocks to 25 mm
    - Higher holding forces
    - More place for electronics.
  - Integrate conveyance actuators on the top
- **Algorithm**
  - Modifying the algorithm to better meet the constraints of lateral displacement of the blocks.
  - Defining a hierarchy of virtual levels
    - Software level over real blocks: agents.
    - Agents allowing to realize macro processing on a set of blocks.
- **Simulator**
  - Add physical simulation of
    - Air-jet forces
    - Micro-parts displacements
  - Add debugging and tracing tools

Questions ?