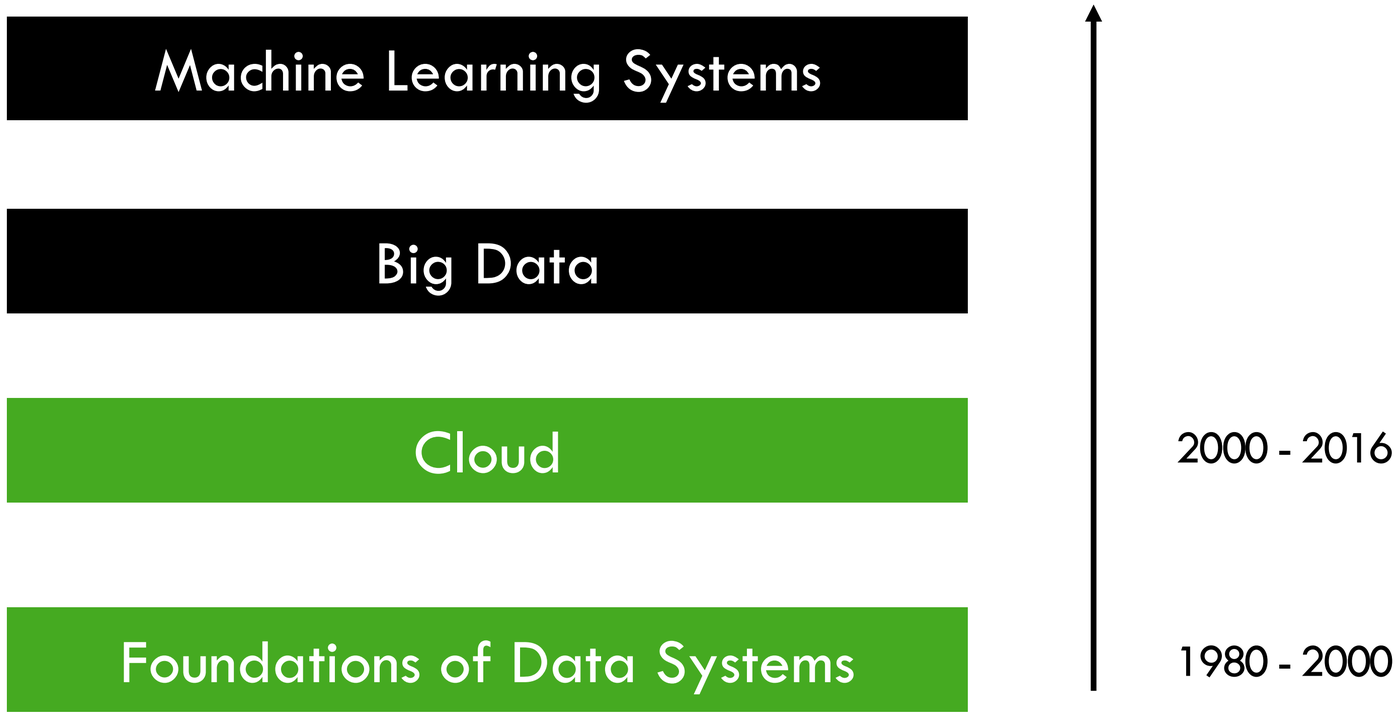
https://hao-ai-lab.github.io/dsc204a-f25/

DSC 204A: Scalable Data Systems
Fall 2025

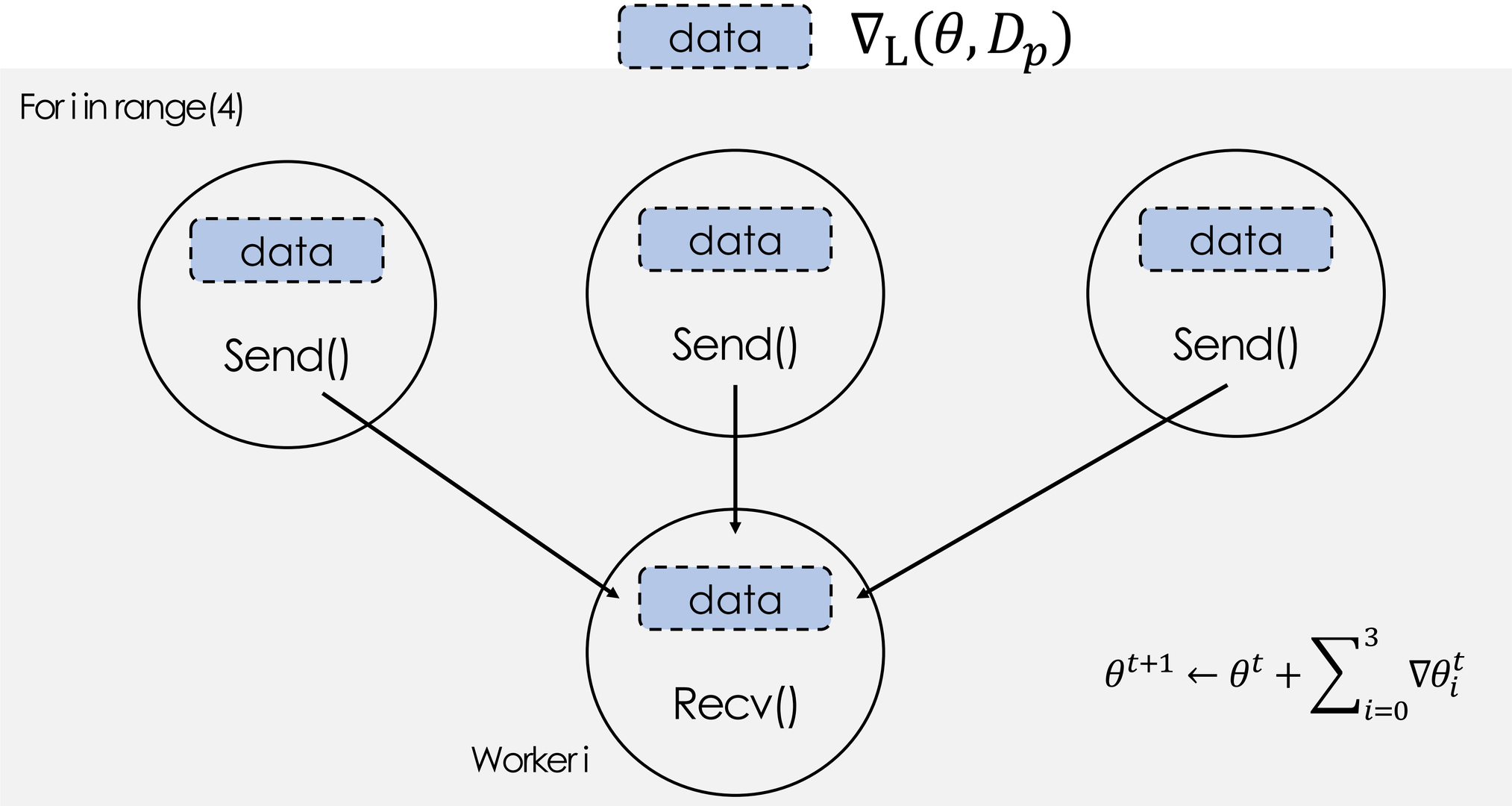
Staff
Instructor: Hao Zhang
TAs: Mingjia Huo, Yuxuan Zhang

@haozhangml @haoailab
haozhang@ucsd.edu

Where We Are


Machine Learning Systems
Big Data
Cloud
Foundations of Data Systems
2000 - 2016
1980 - 2000

Problem: We need **All-**Reduce



Program This? Will be in PA2!

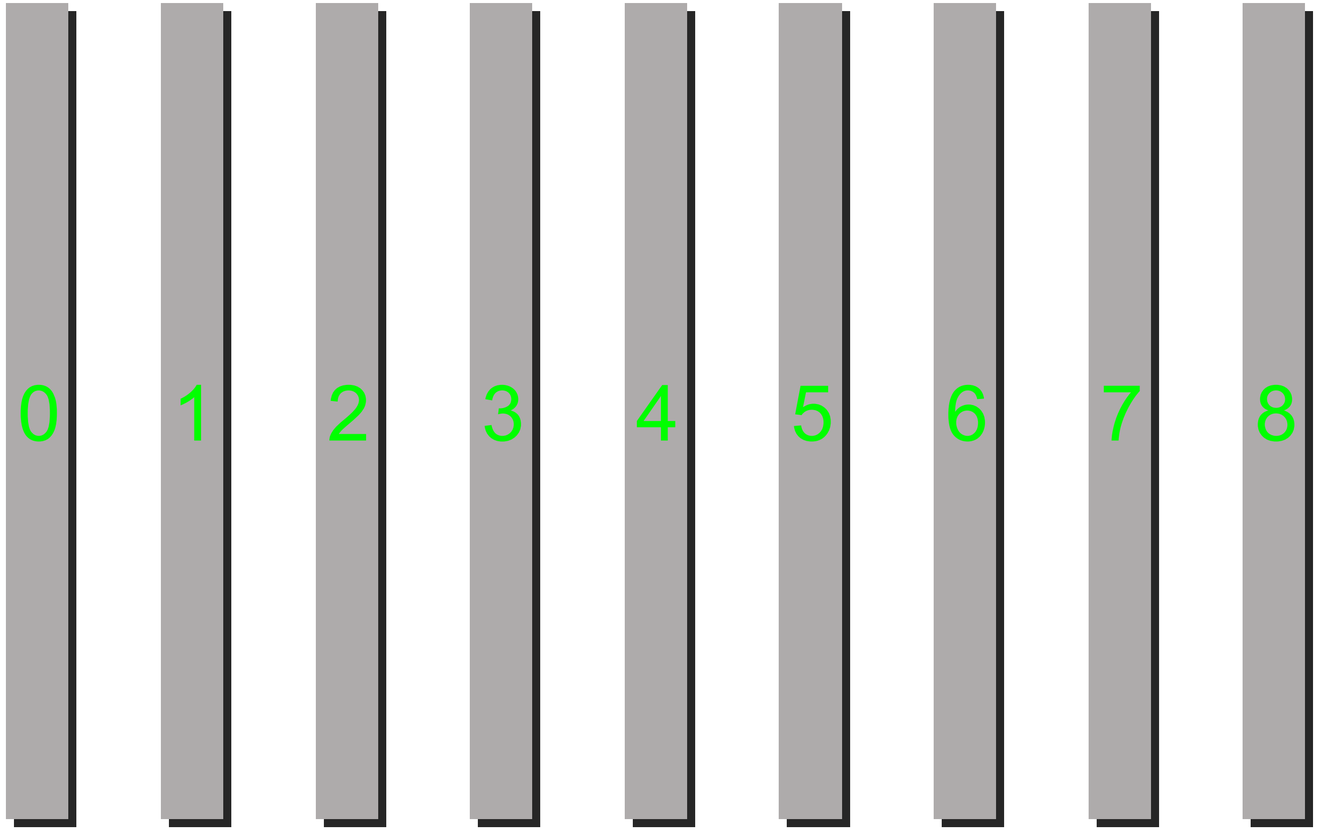
Performance

- Message size over networks:
  - Sum: 3N
  - Send Sum back: 3N
  - = 6N
- Can we do better?
  - Hint: we cannot do better than 3N

Why Collective Communication?

- Programming Convenience
  - Use a set of well-defined communication primitives to express complex communication patterns
- Unification and Performance
  - Since they are well defined and well structured, we can optimize them to the extreme
- ML Systems ❤️ Collective communication

Make it Formal

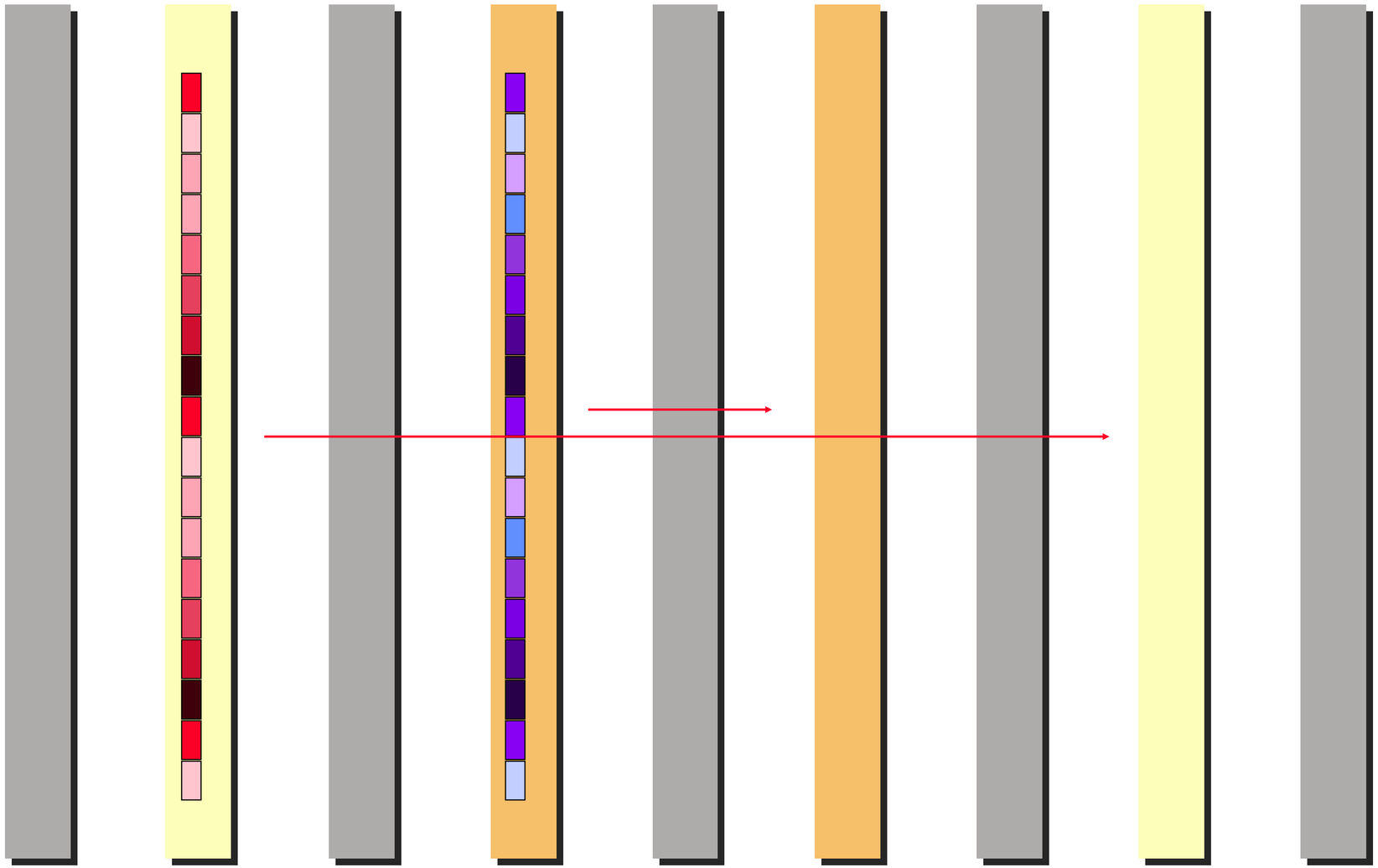


- A 1D **Mesh** of workers (or devices, or nodes)

Model of Parallel Computation

- a node can send directly to any other node (maybe not true)
- a node can simultaneously receive and send
- cost of communication
  - sending a message of length *n* between any two nodes

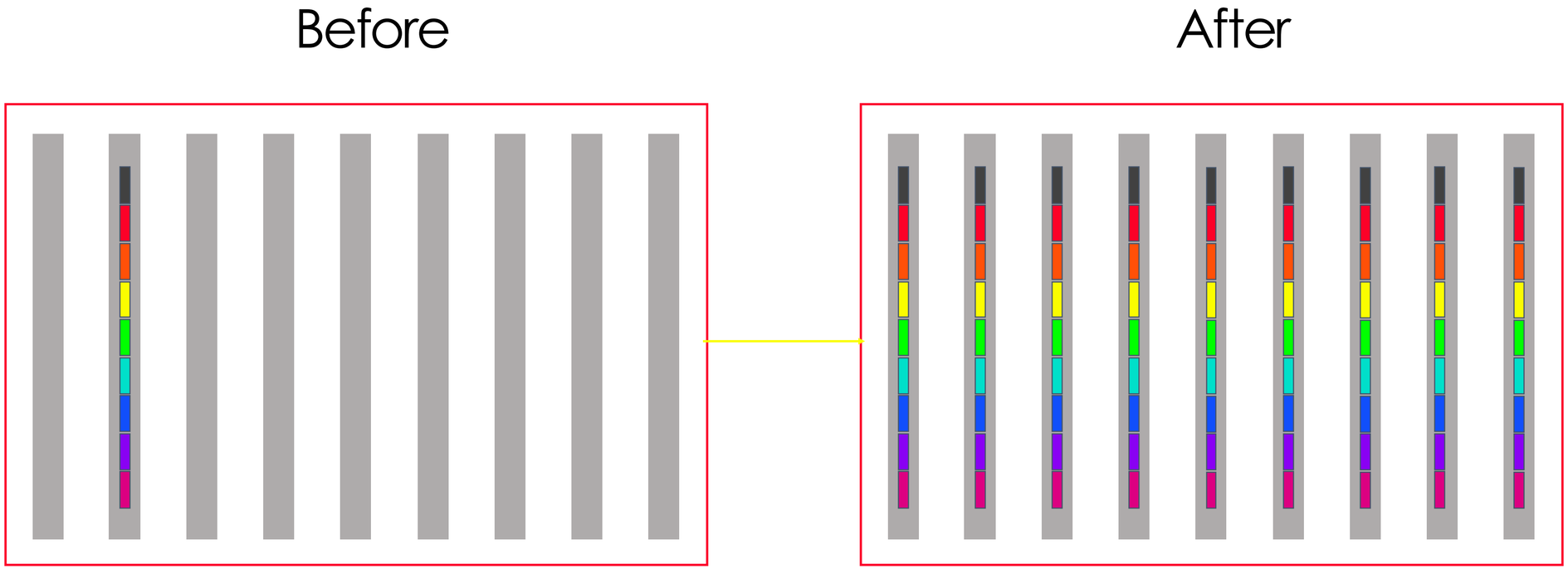
$$\alpha + n\,\beta$$



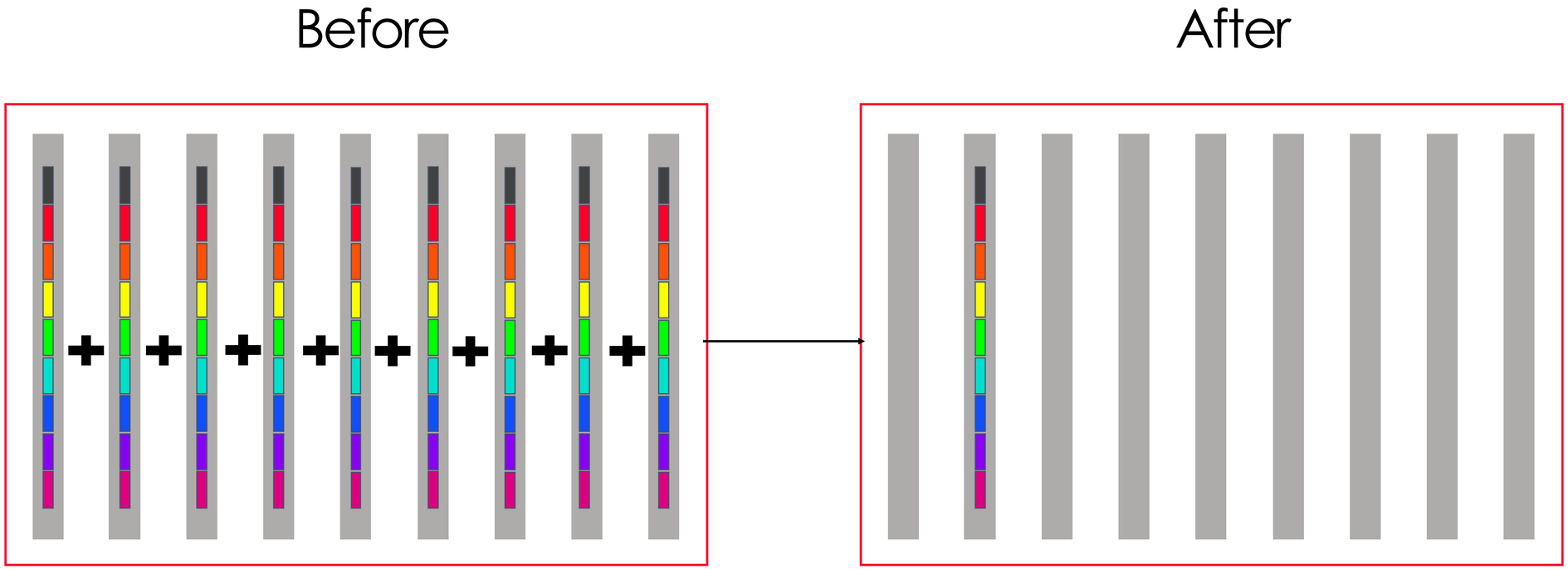
Collective Communications

- Broadcast
- Reduce(-to-one)
- Scatter
- Gather
- Allgather
- Reduce-scatter
- Allreduce
- All-2-All

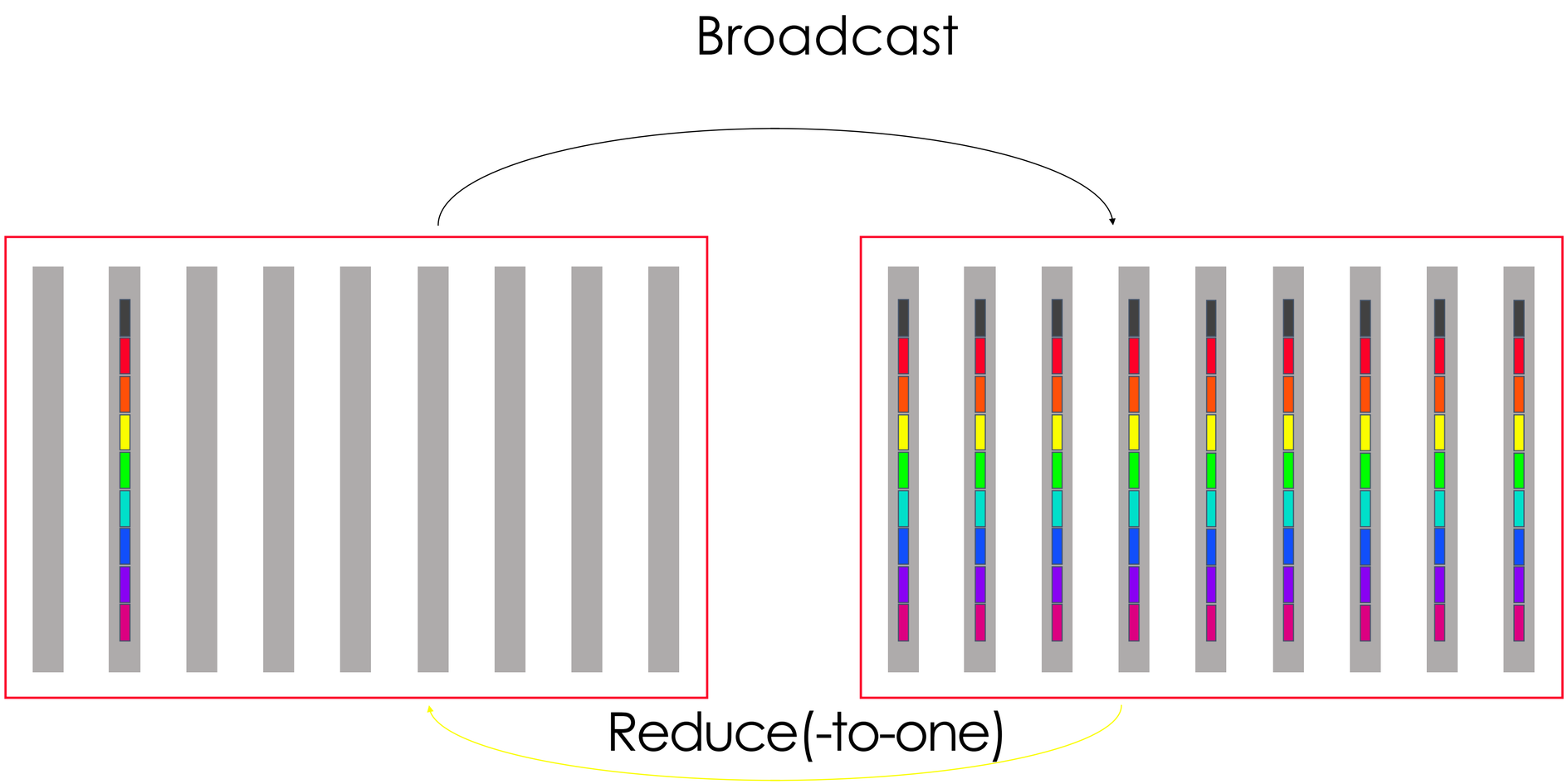
Broadcast



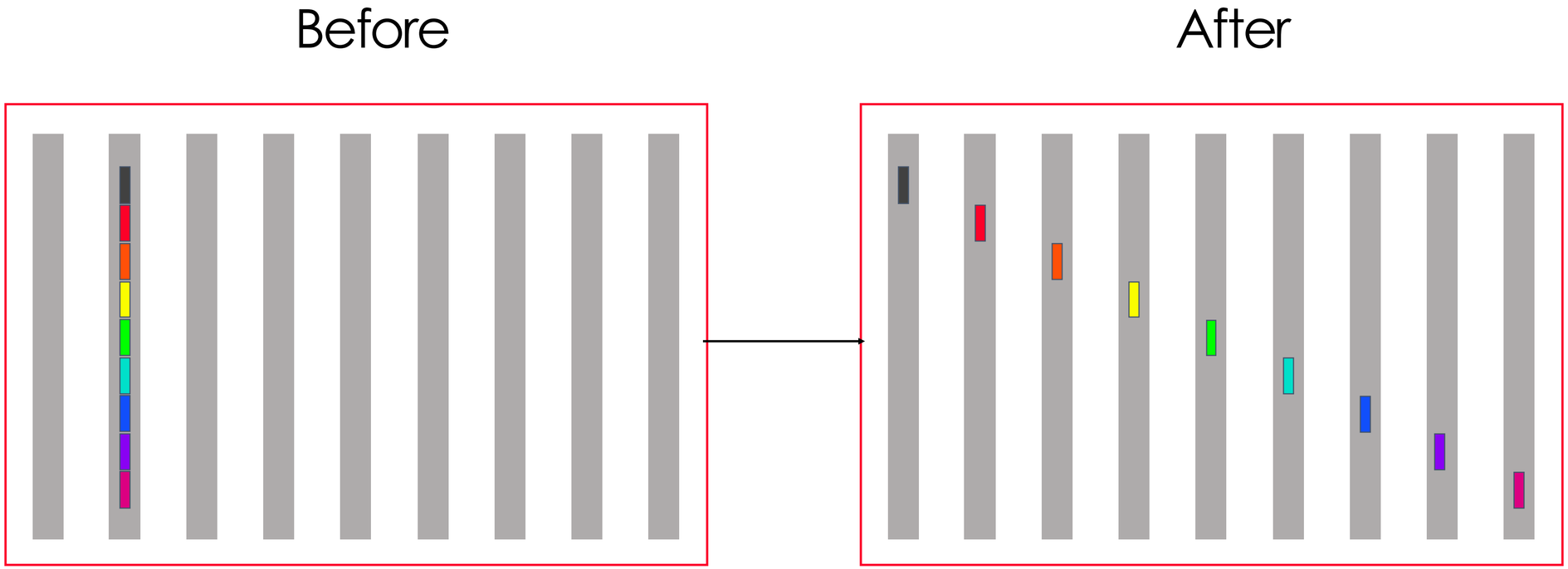
Reduce(-to-one)



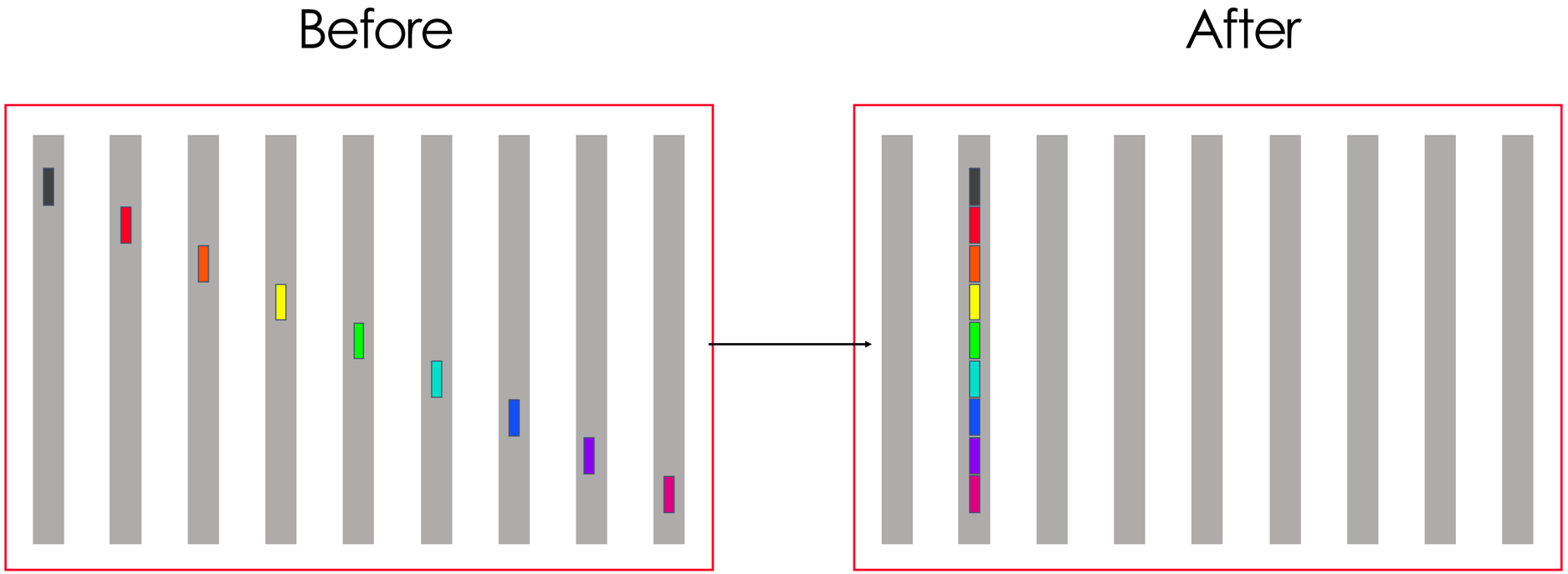
Broadcast/Reduce(-to-one)



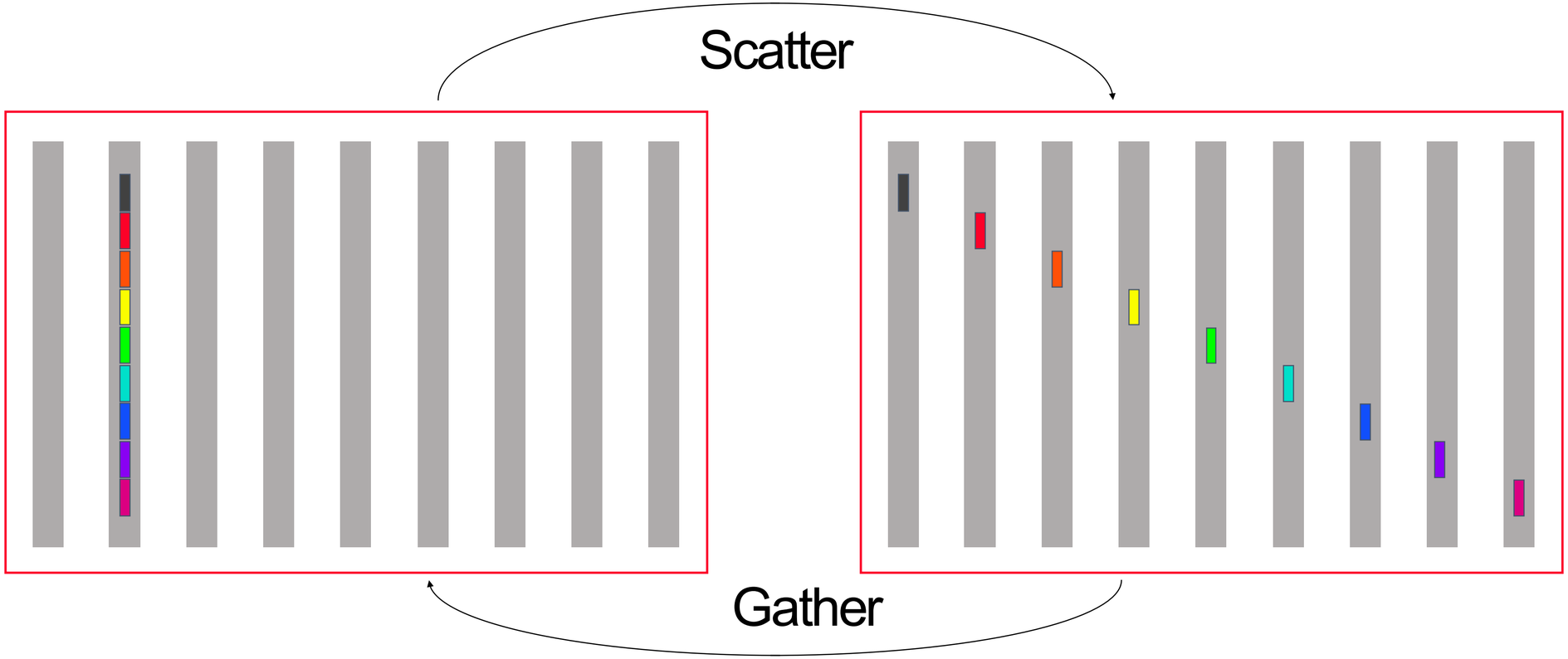
Scatter



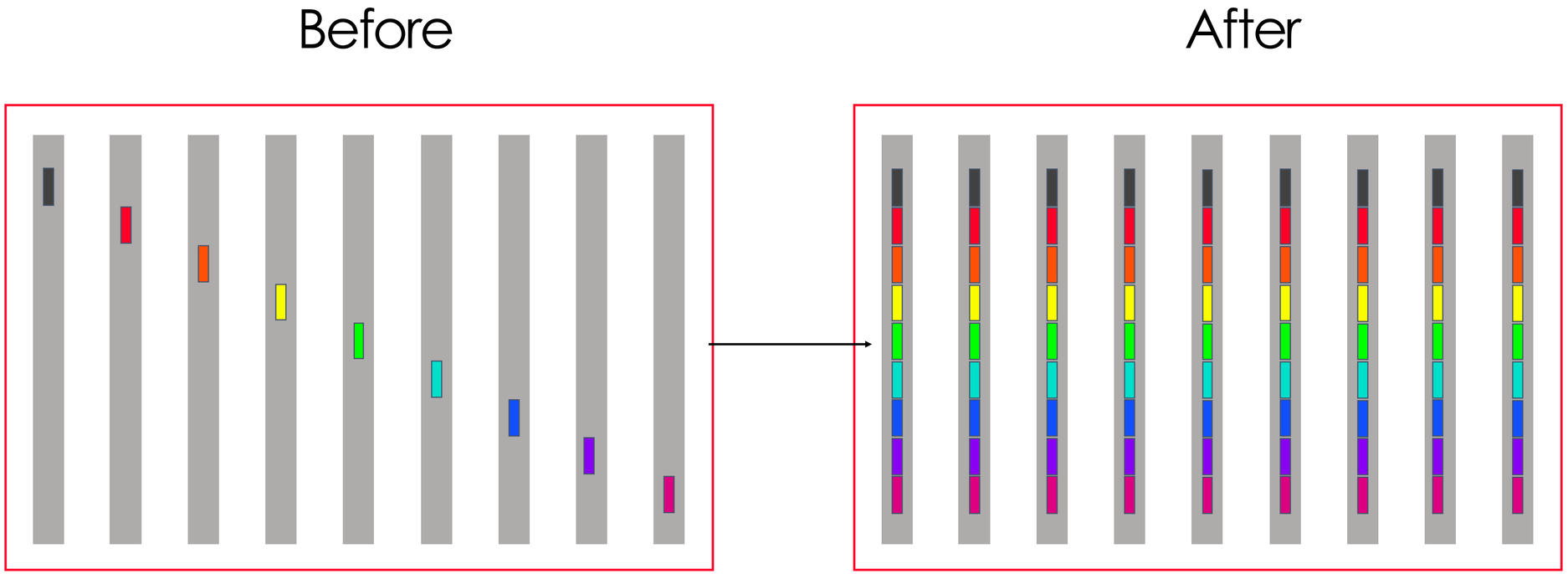
Gather



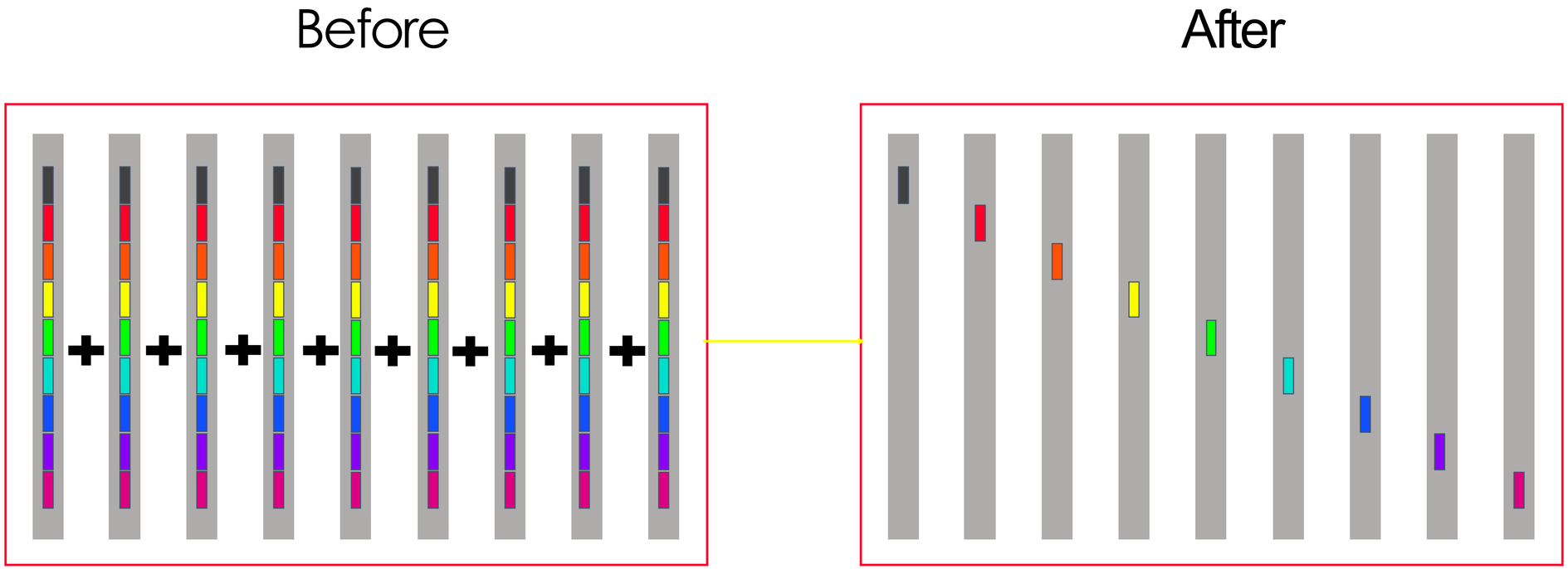
Scatter/Gather



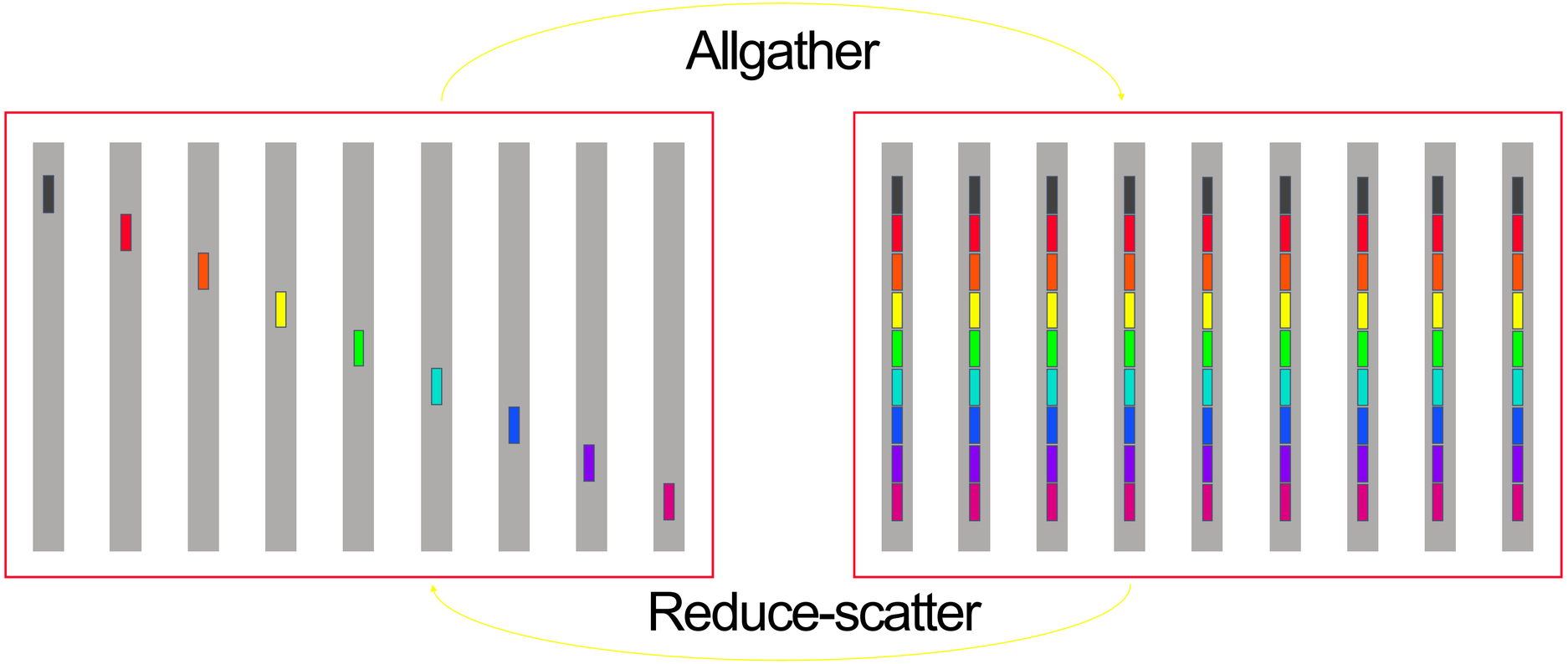
Allgather



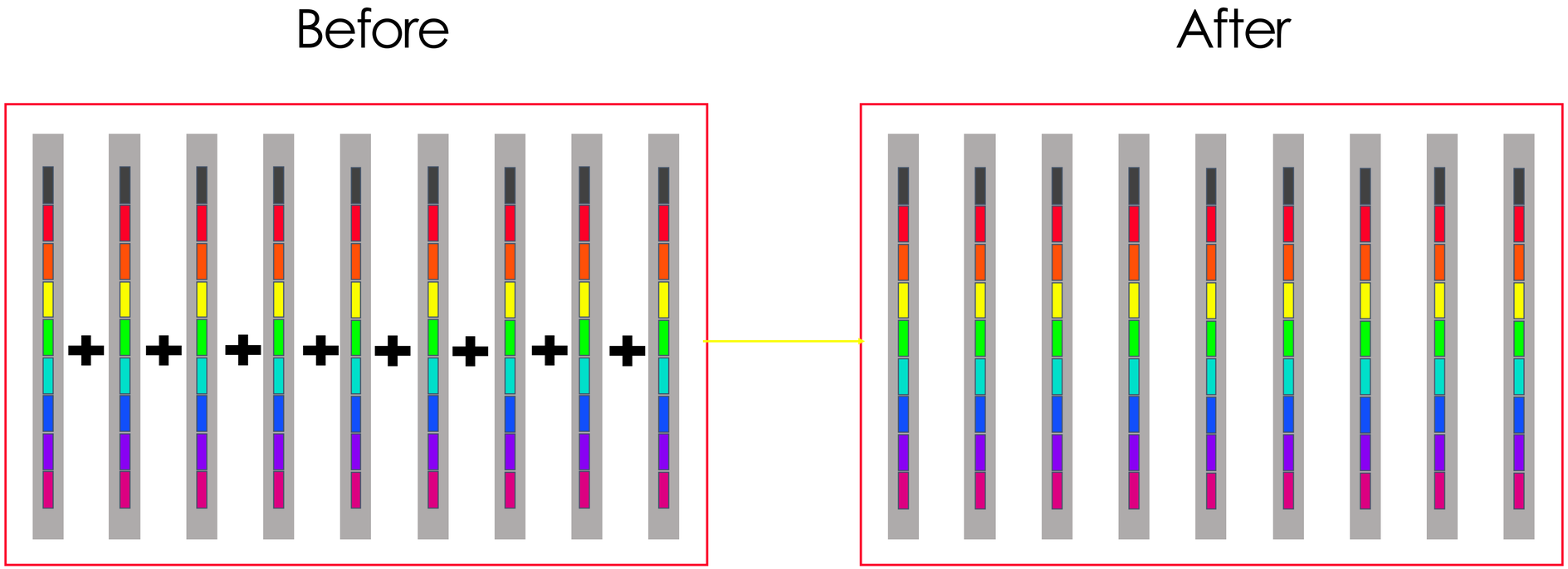
Reduce-scatter



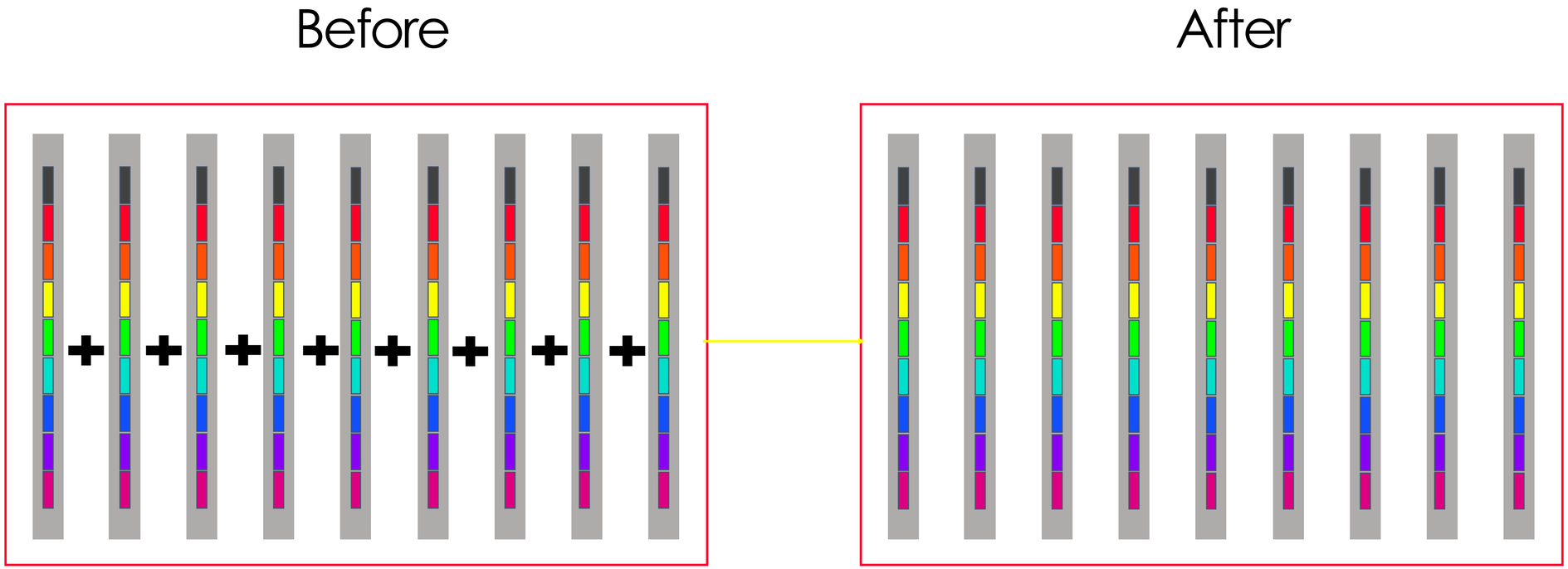
Allgather/Reduce-scatter



Allreduce



All2All



Some Facts

- Collective is much more expensive than P2P
  - Collective can be assembled using many P2P
  - Collective is cheaper than realizing collective using P2P (we'll see)
- Collective is highly optimized in the past 20 years
  - Look out for "X"CCL libraries
    - NCCL, MCCL, OneCCL, UCCL
- Collective is not fault-tolerant
  - A major sources of faults in ML systems

Communication Model: $\alpha\beta$ model

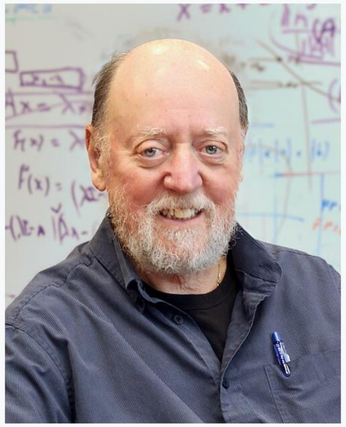
Communication Model: $\alpha + n\,\beta, \beta = \frac{1}{B}$

- Small Message size ($n \to 0$): α dominates, emphasize latency
- Large Message Size ($n \to +\infty$): $n\,\beta$ dominate, emphasize bandwidth utilization

Two Family of Mainstream Algorithms/Implementations

- Small message: Minimum Spanning Tree algorithm
  - Emphasize **low latency**

- Large Message: Ring algorithm
  - Emphasize **bandwidth utilization**



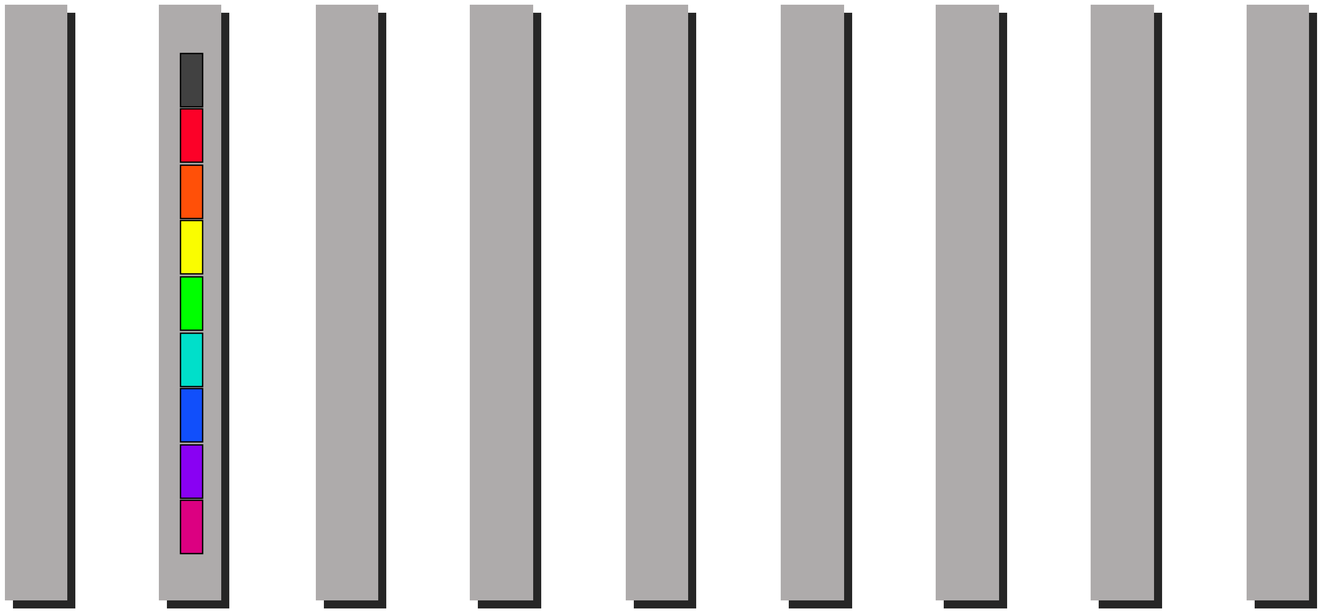
Dongarra in 2022

- There are 50+ different algorithms developed in the past 50 years by a community called “High-performance computing”
  - 2021 Turing award

General principles: Low Latency

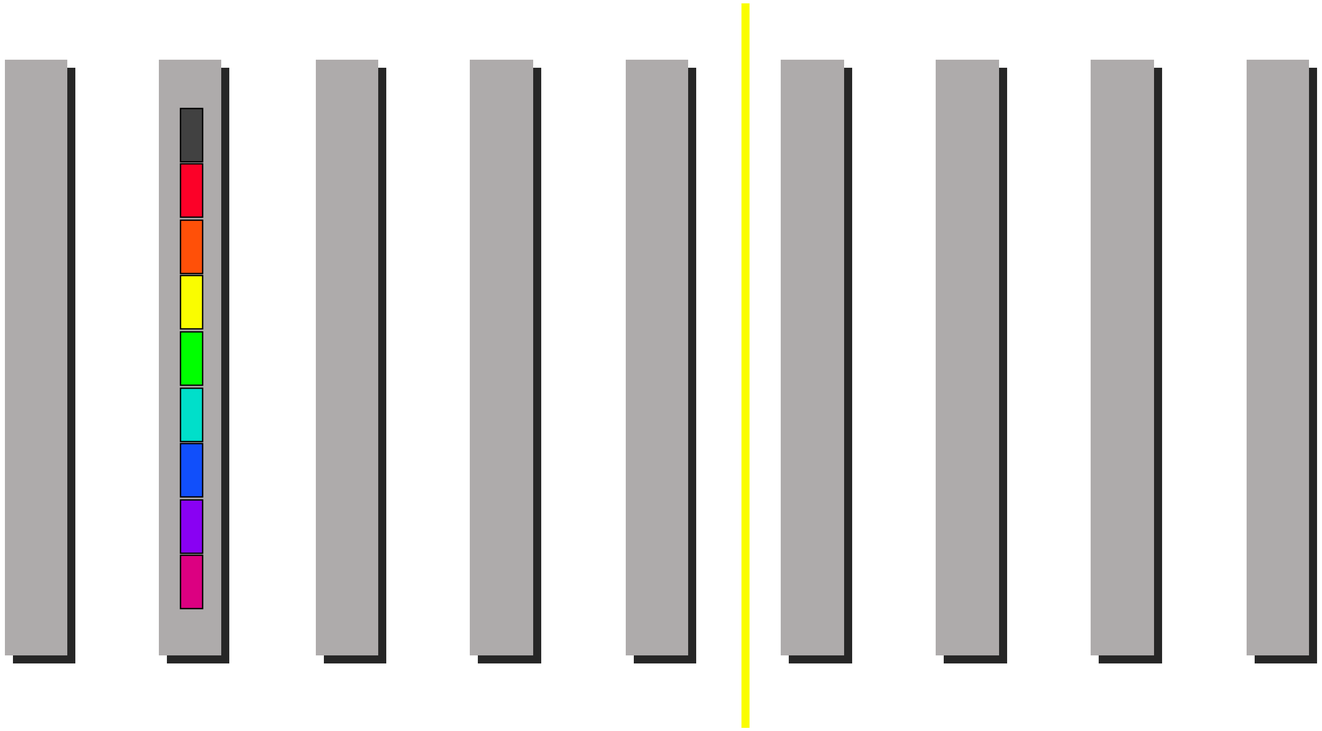
- Minimize the number of rounds needed for communication
- Minimal-spanning tree algorithm

General principles



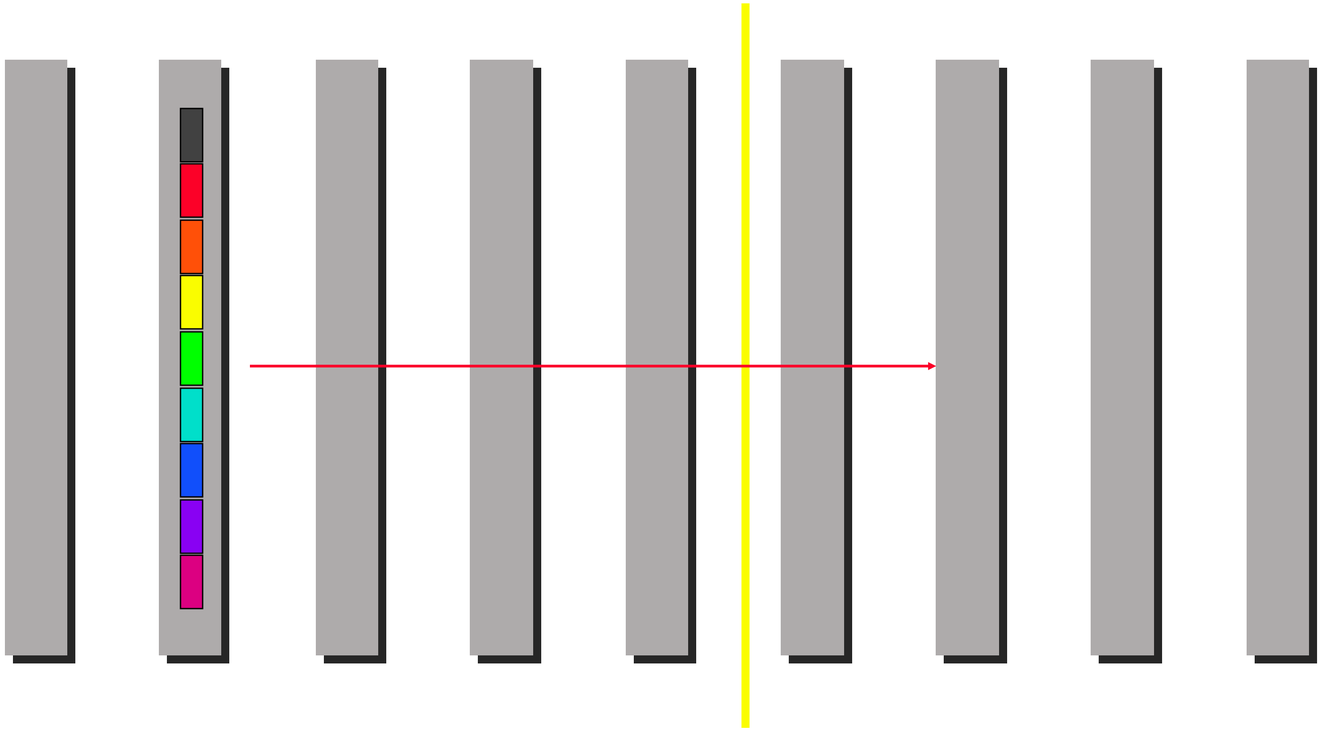
- message starts on one processor

General principles



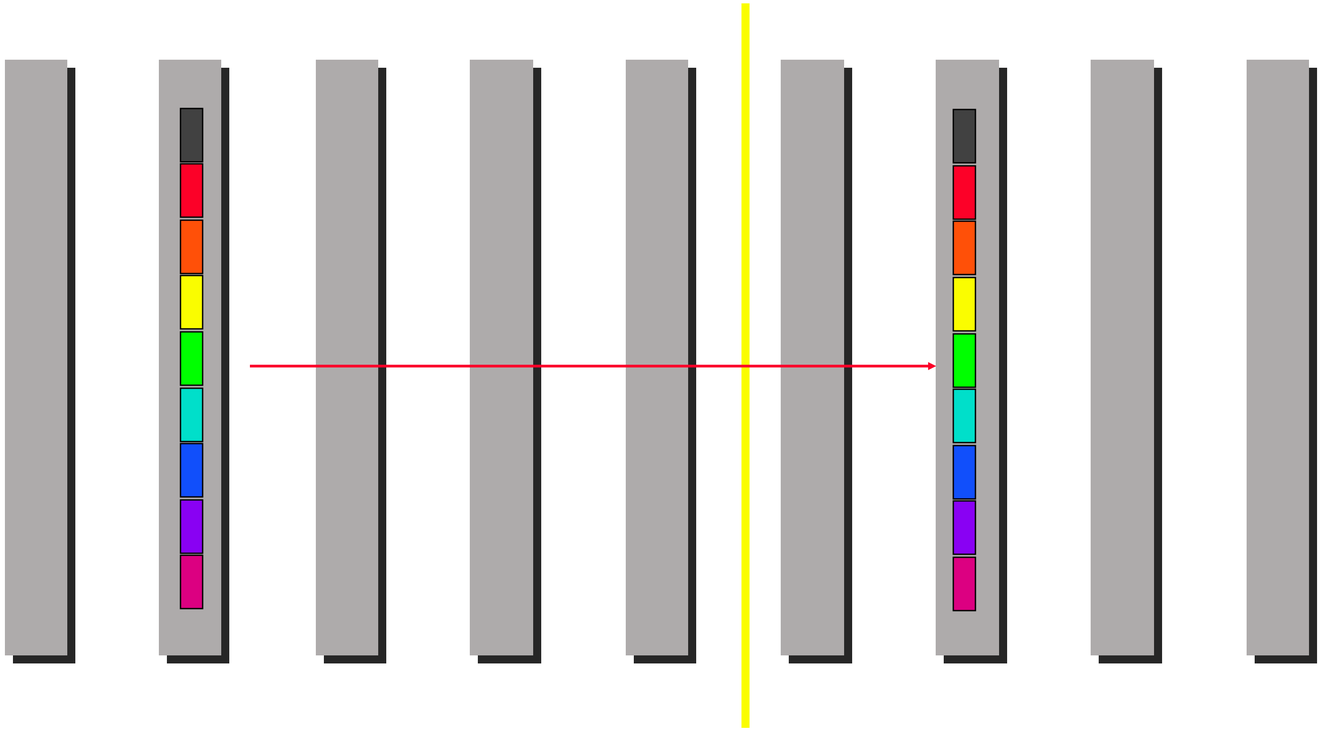
- divide logical linear array in half

General principles



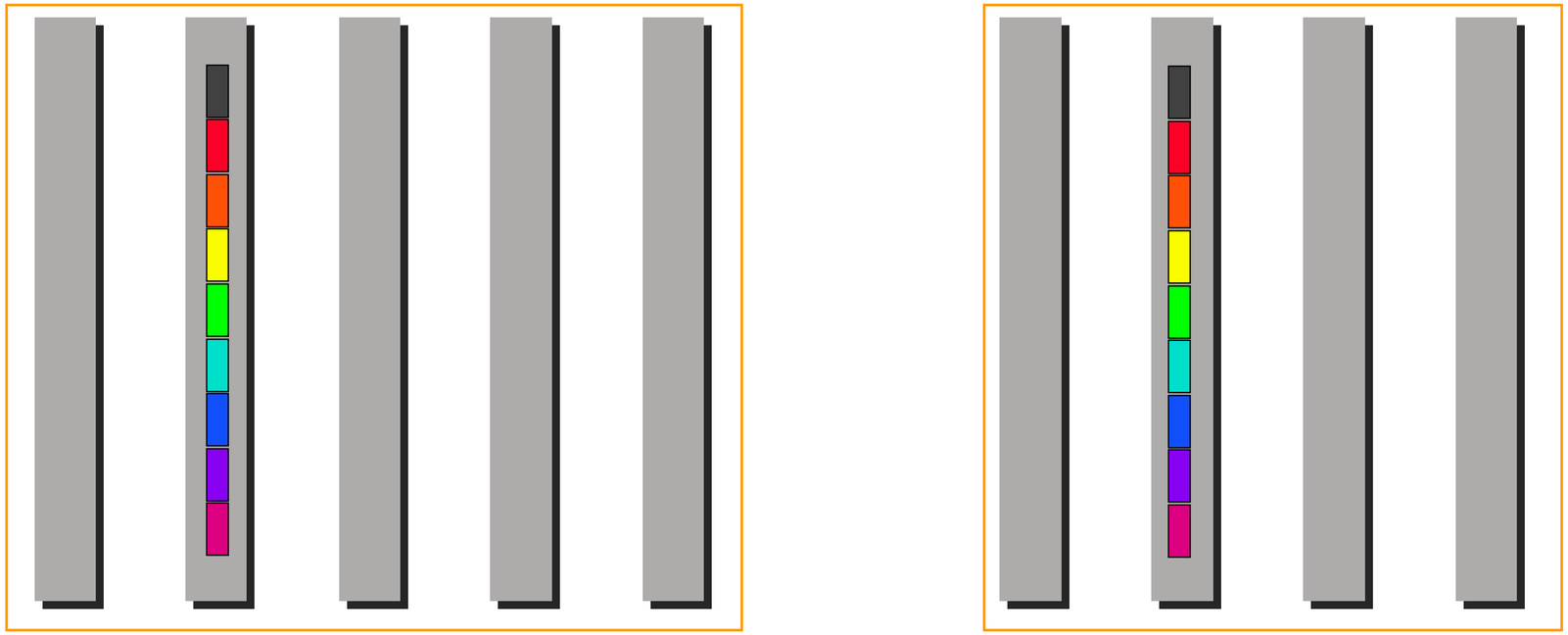
- send message to the half of the network that does not contain the current node (root) that holds the message

General principles



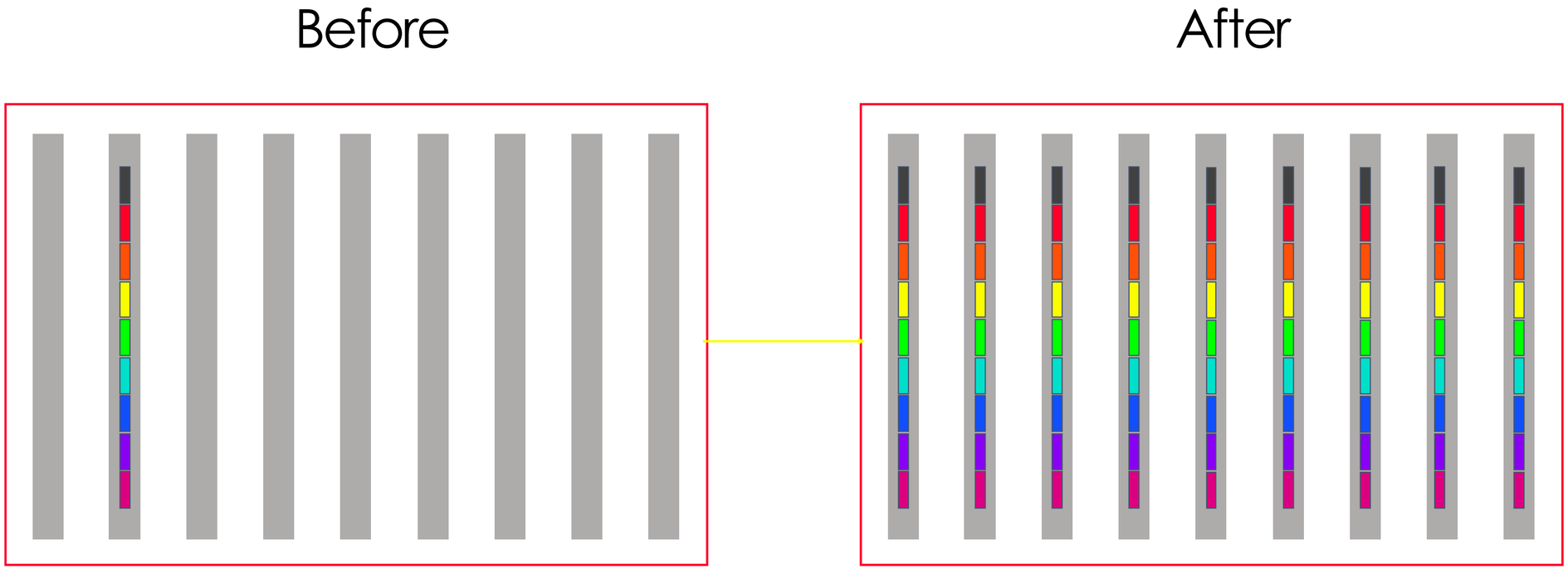
- send message to the half of the network that does not contain the current node (root) that holds the message

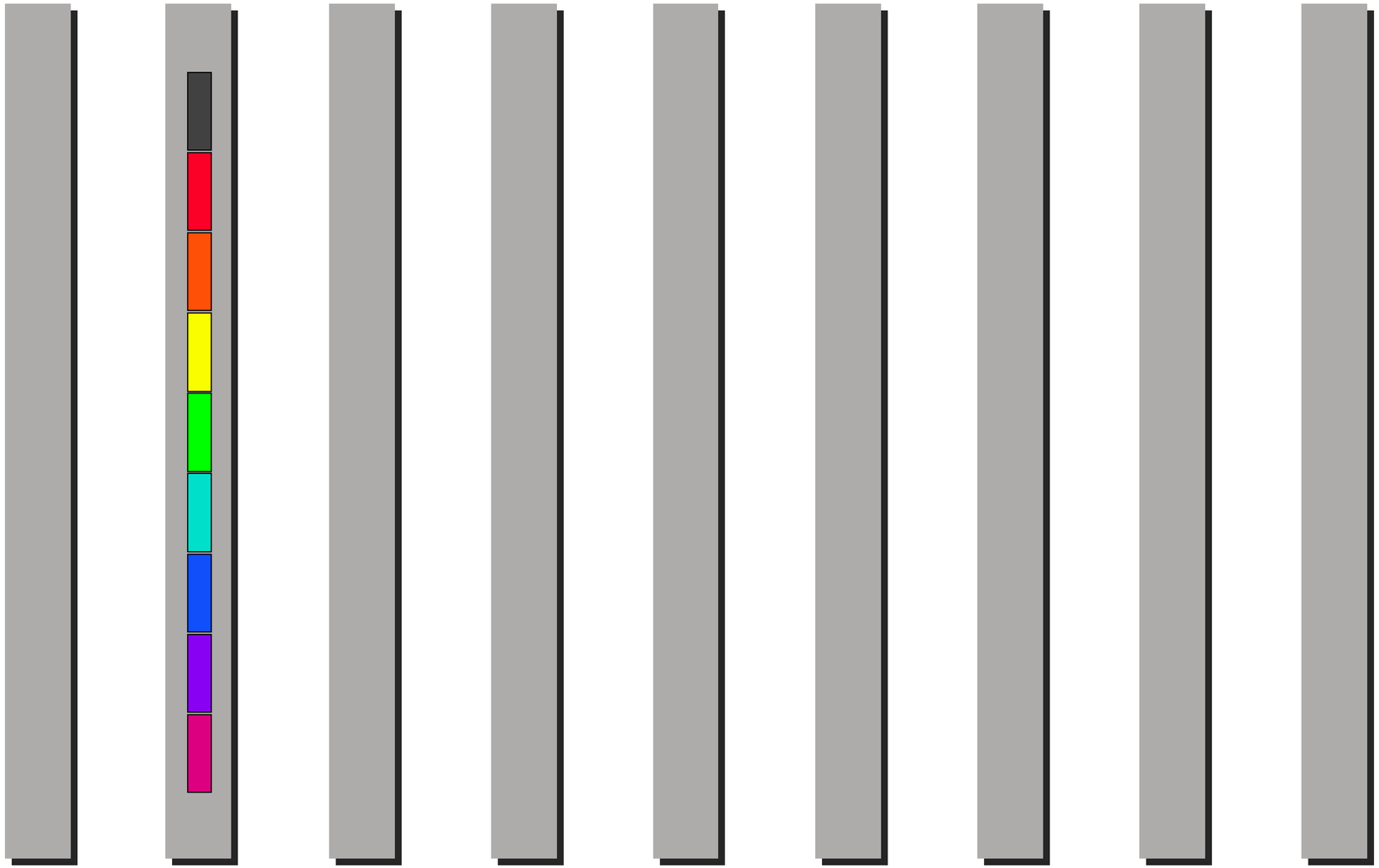
General principles

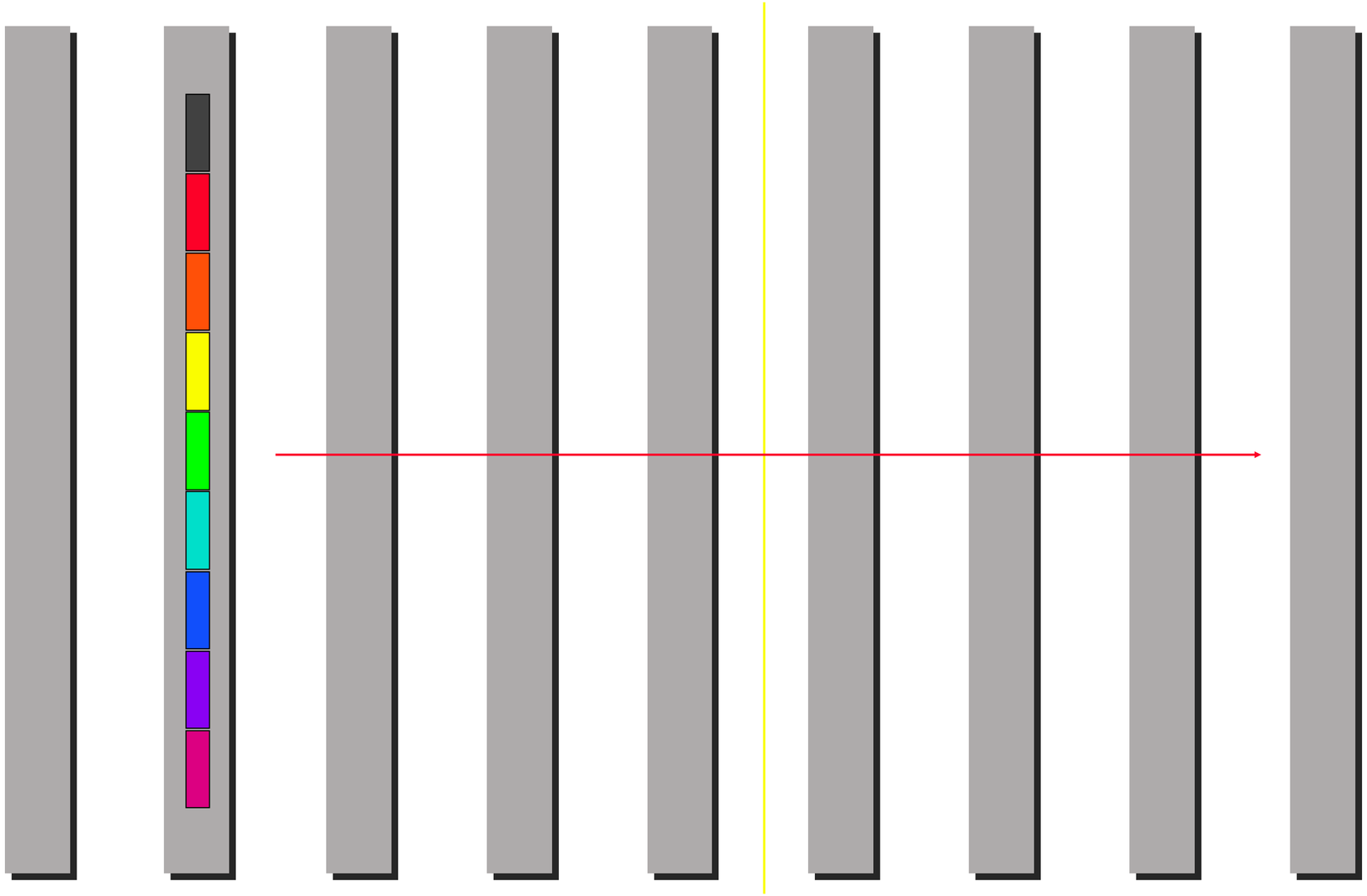


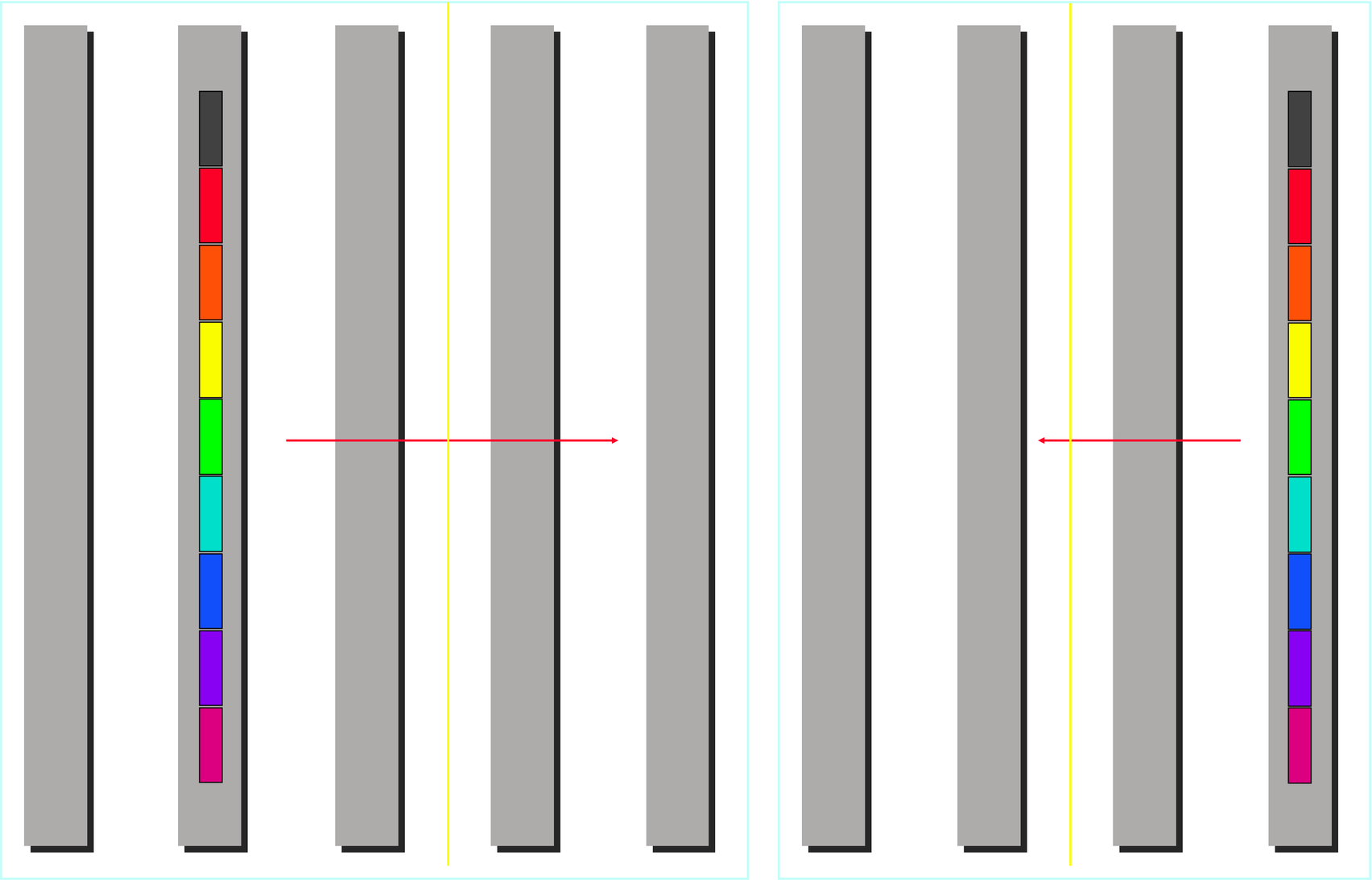
- continue recursively in each of the two halves

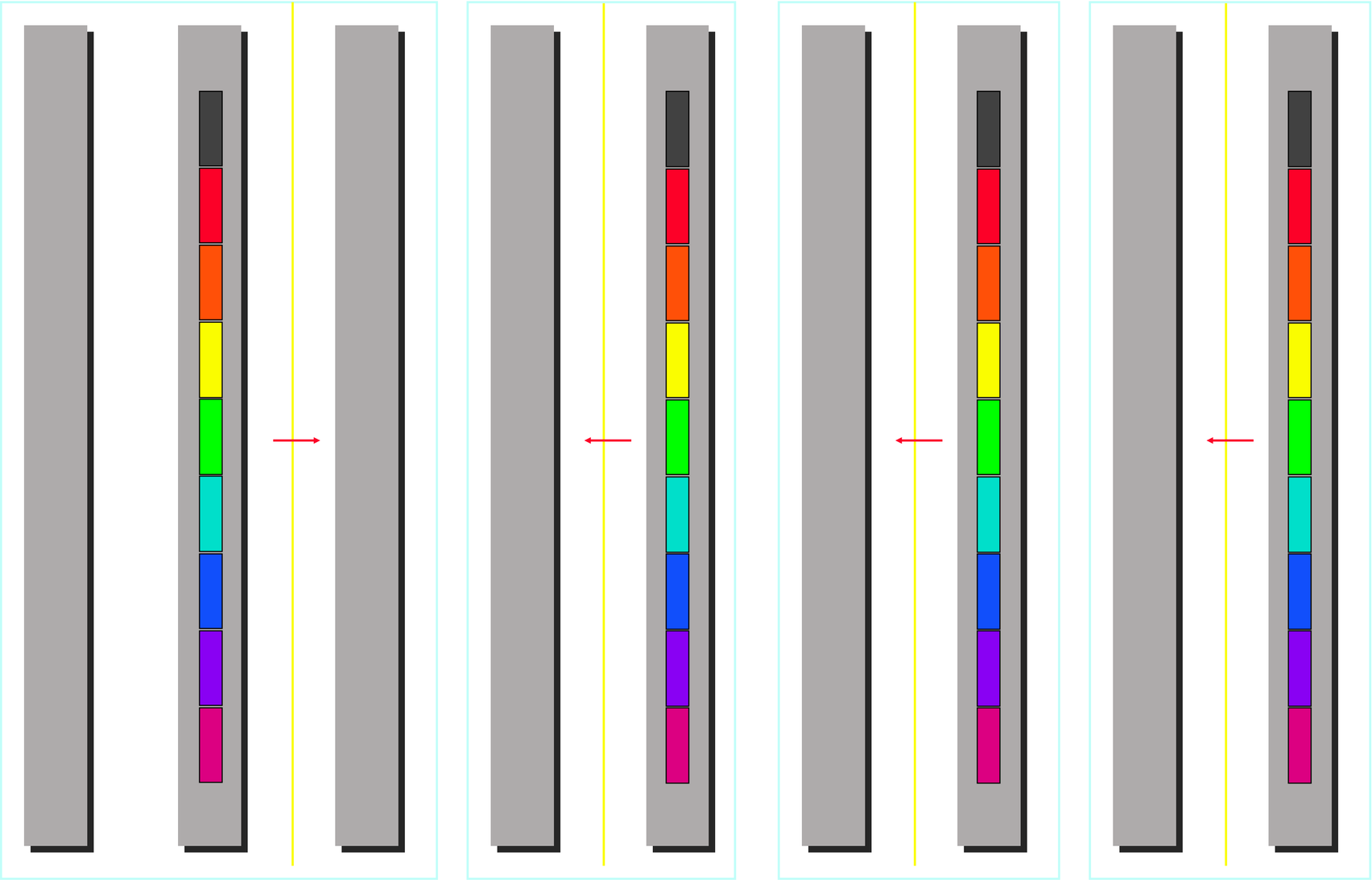
Broadcast

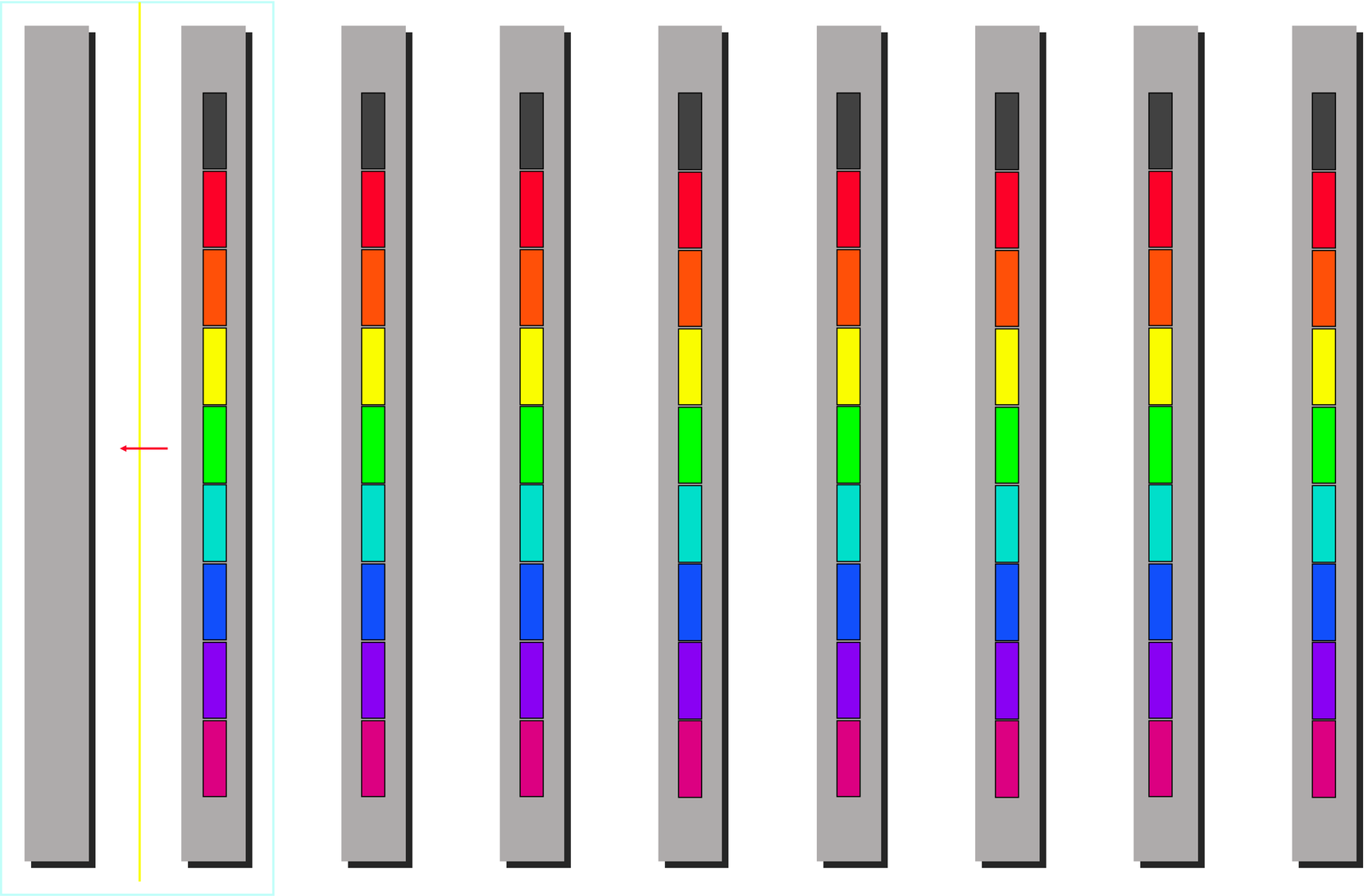


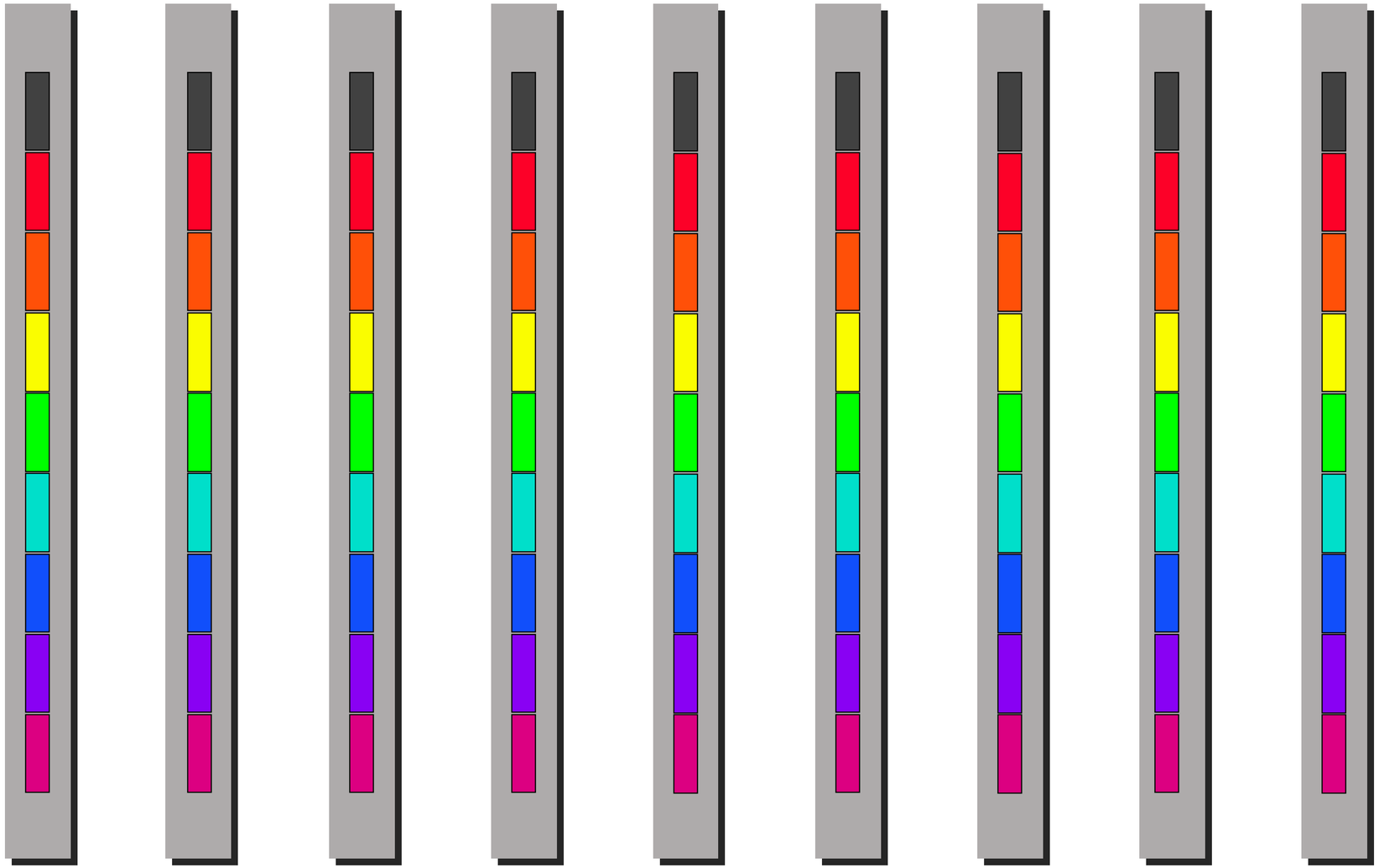








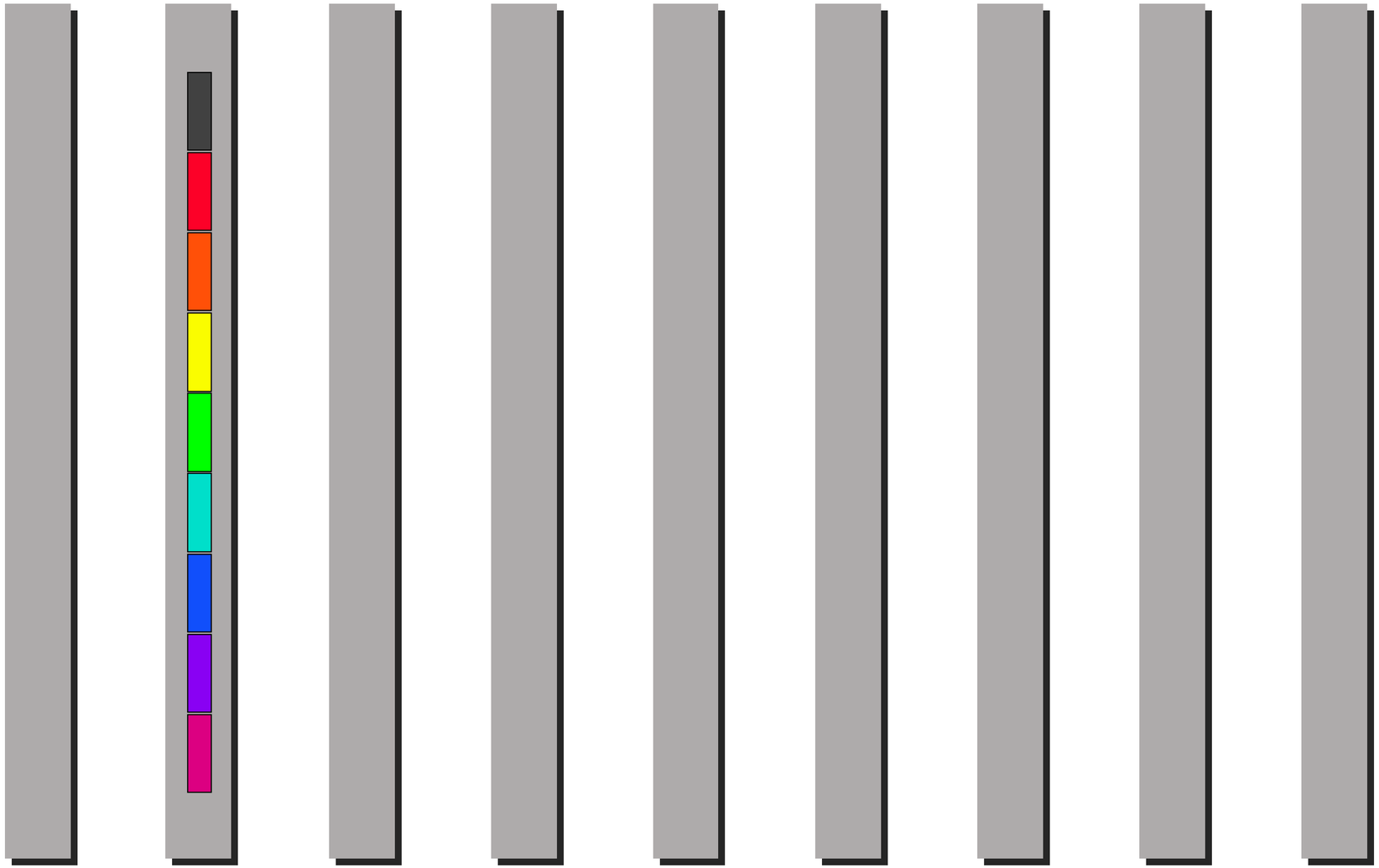


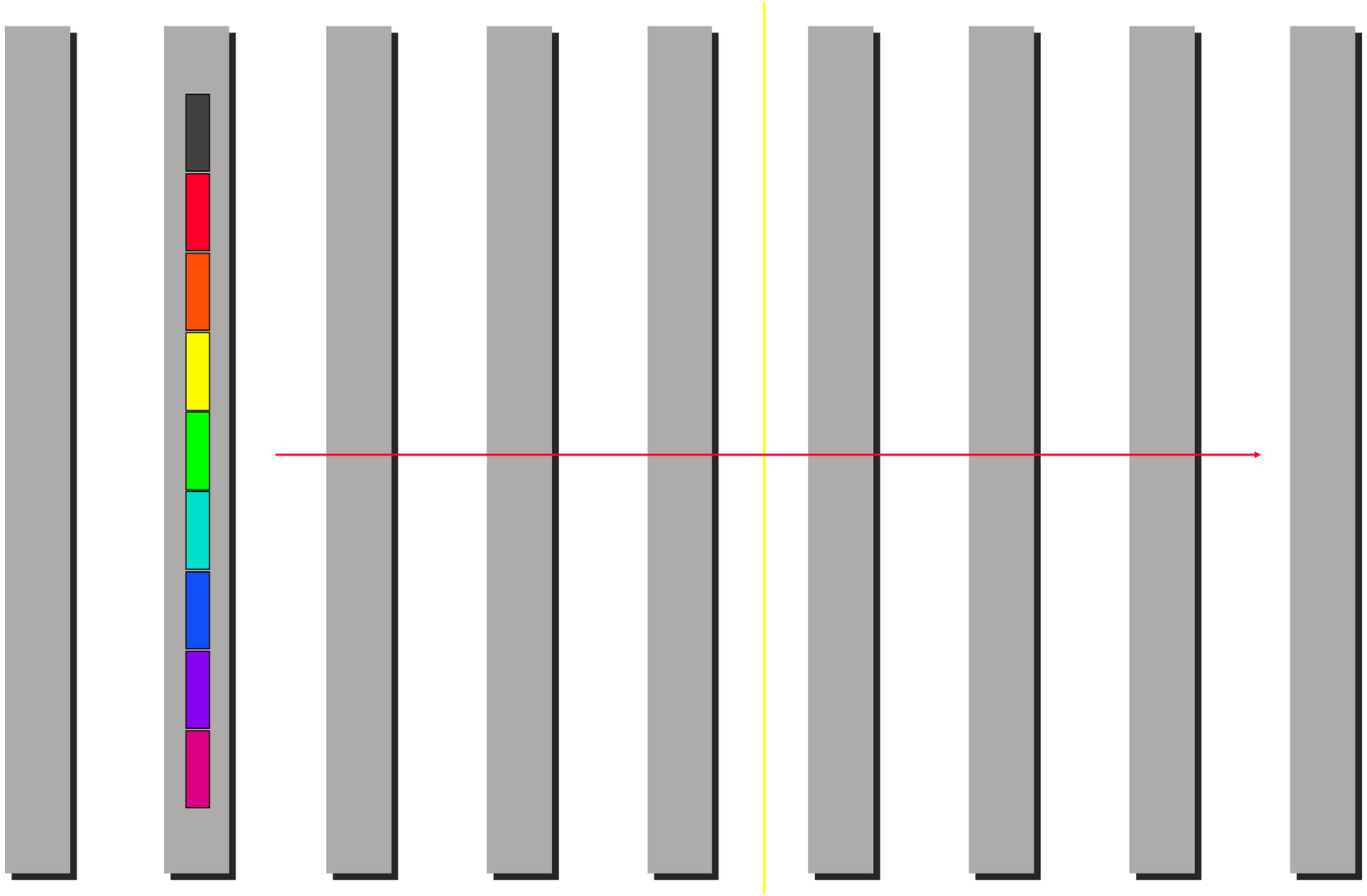


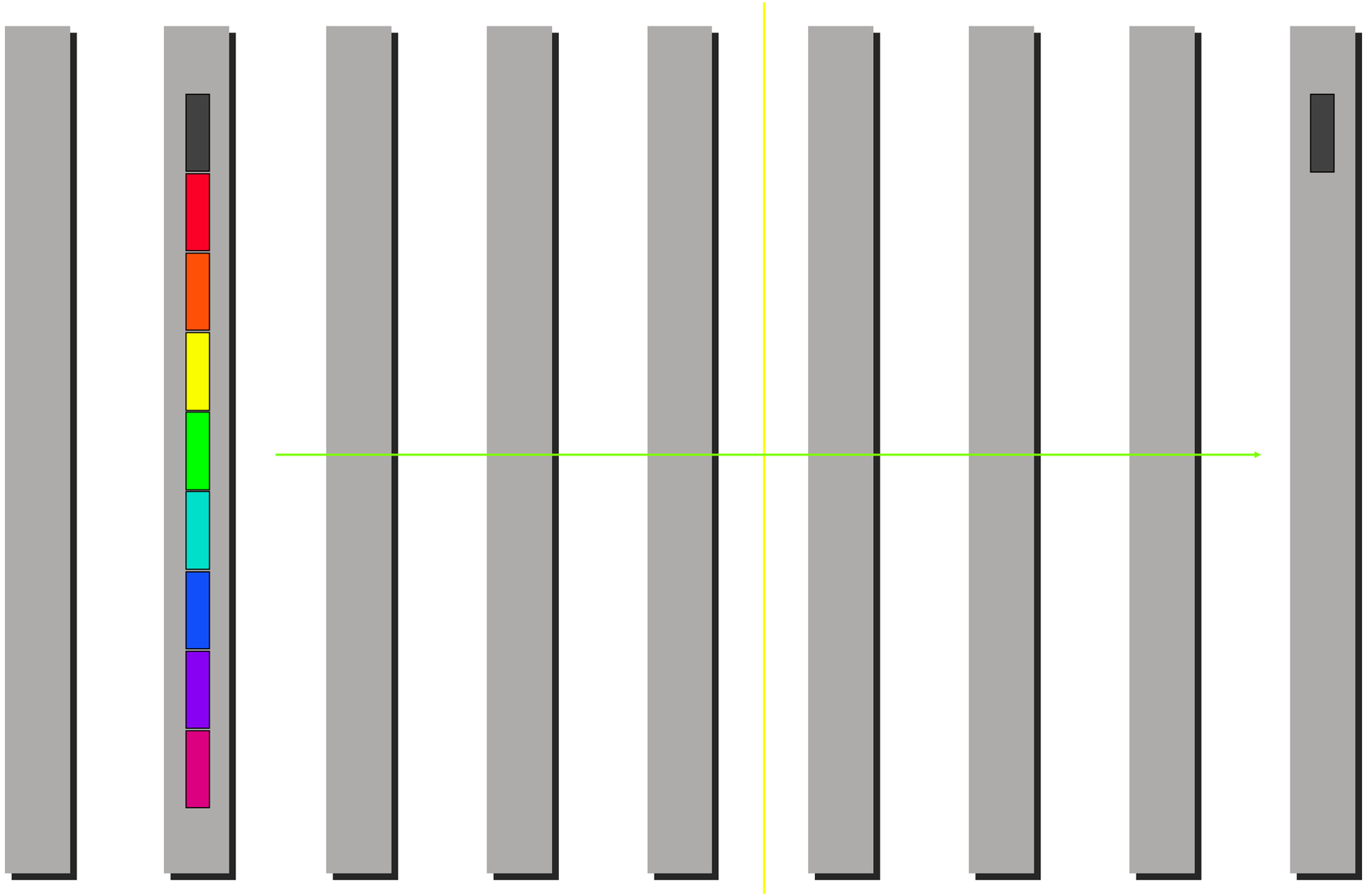
Let us view this more closely

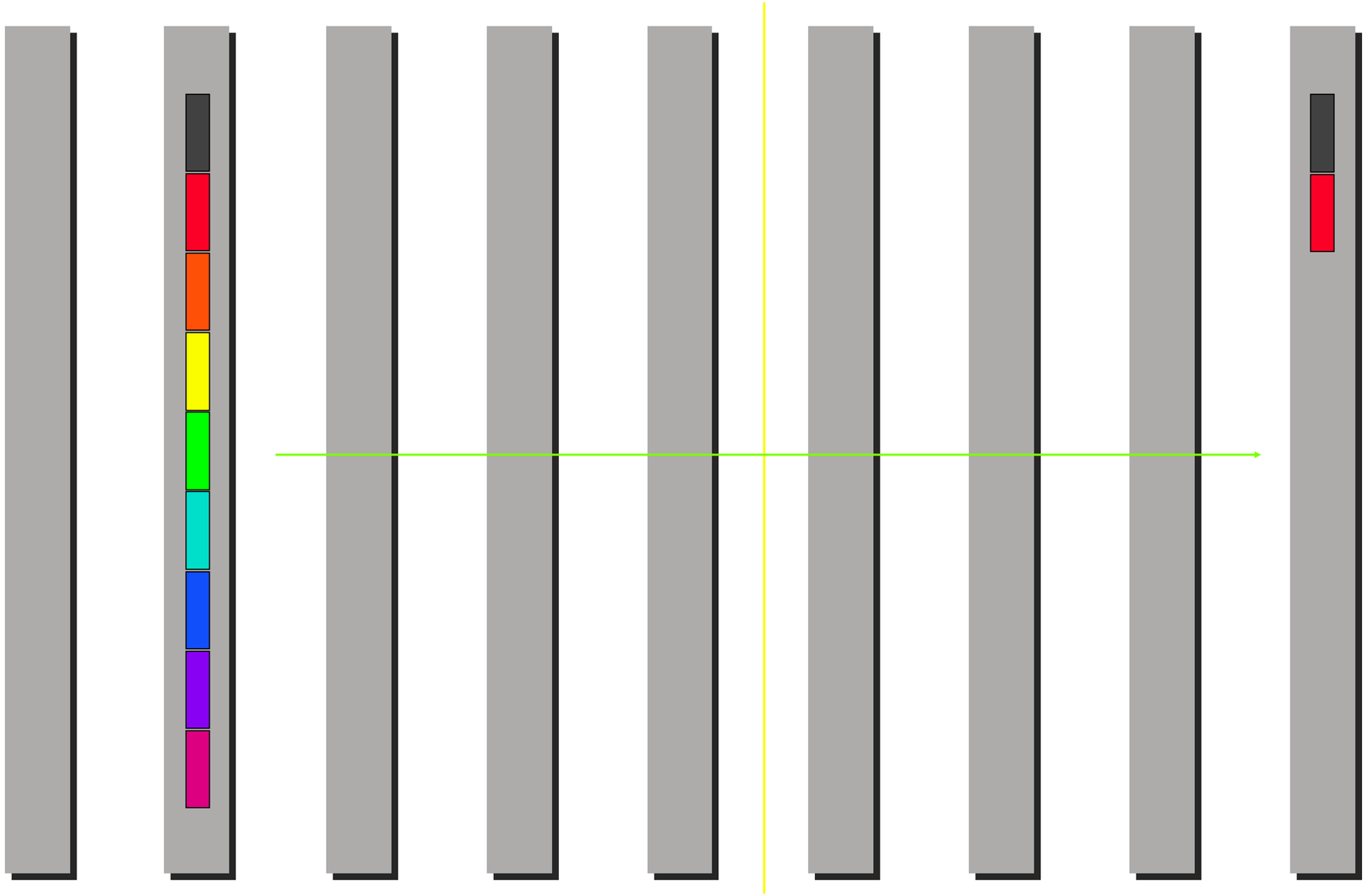
- Red arrows indicate startup of communication (leading to latency, α)

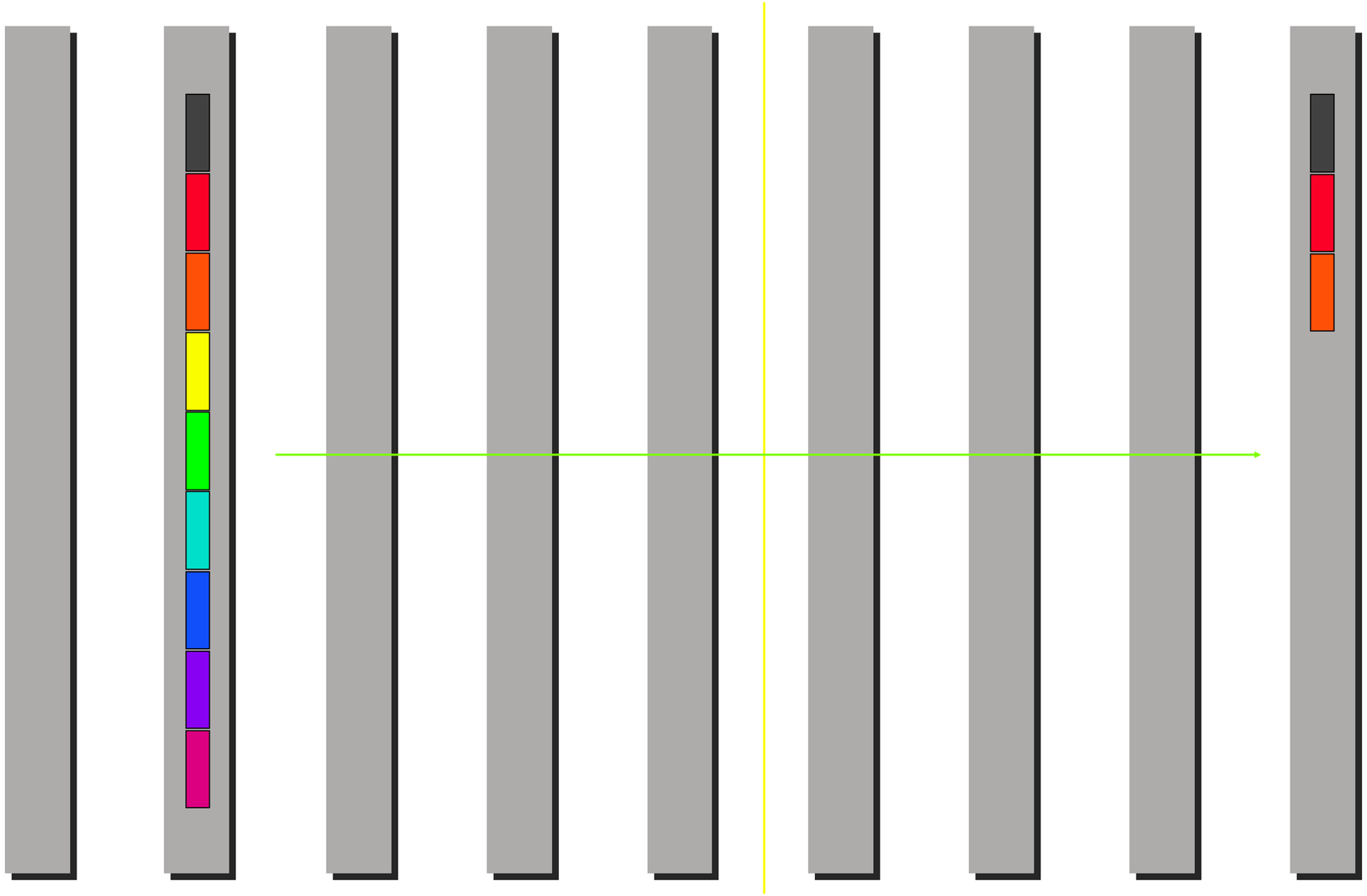
- Green arrows indicate packets in transit (leading to a bandwidth related cost proportional to β and the length of the packet

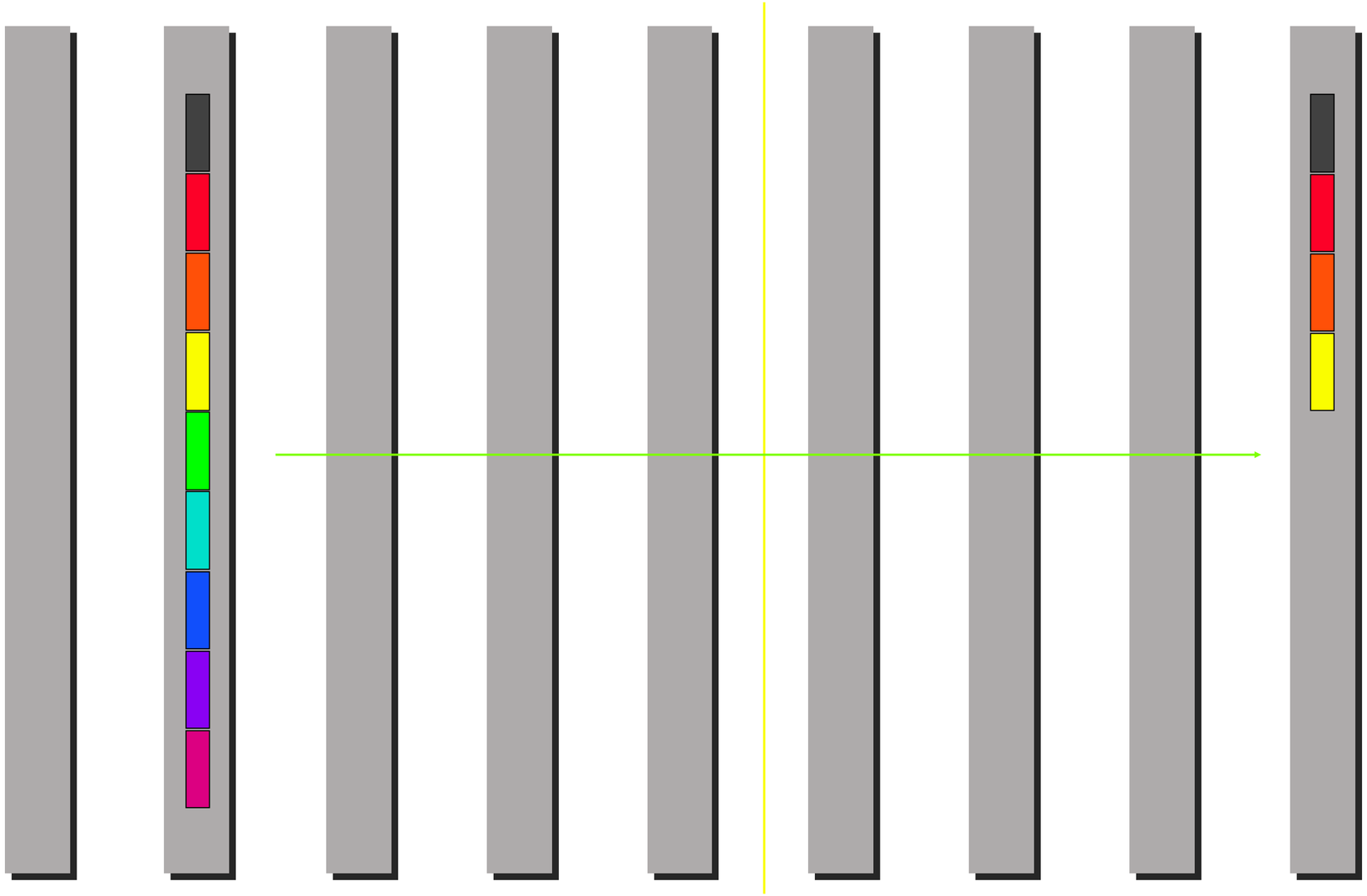


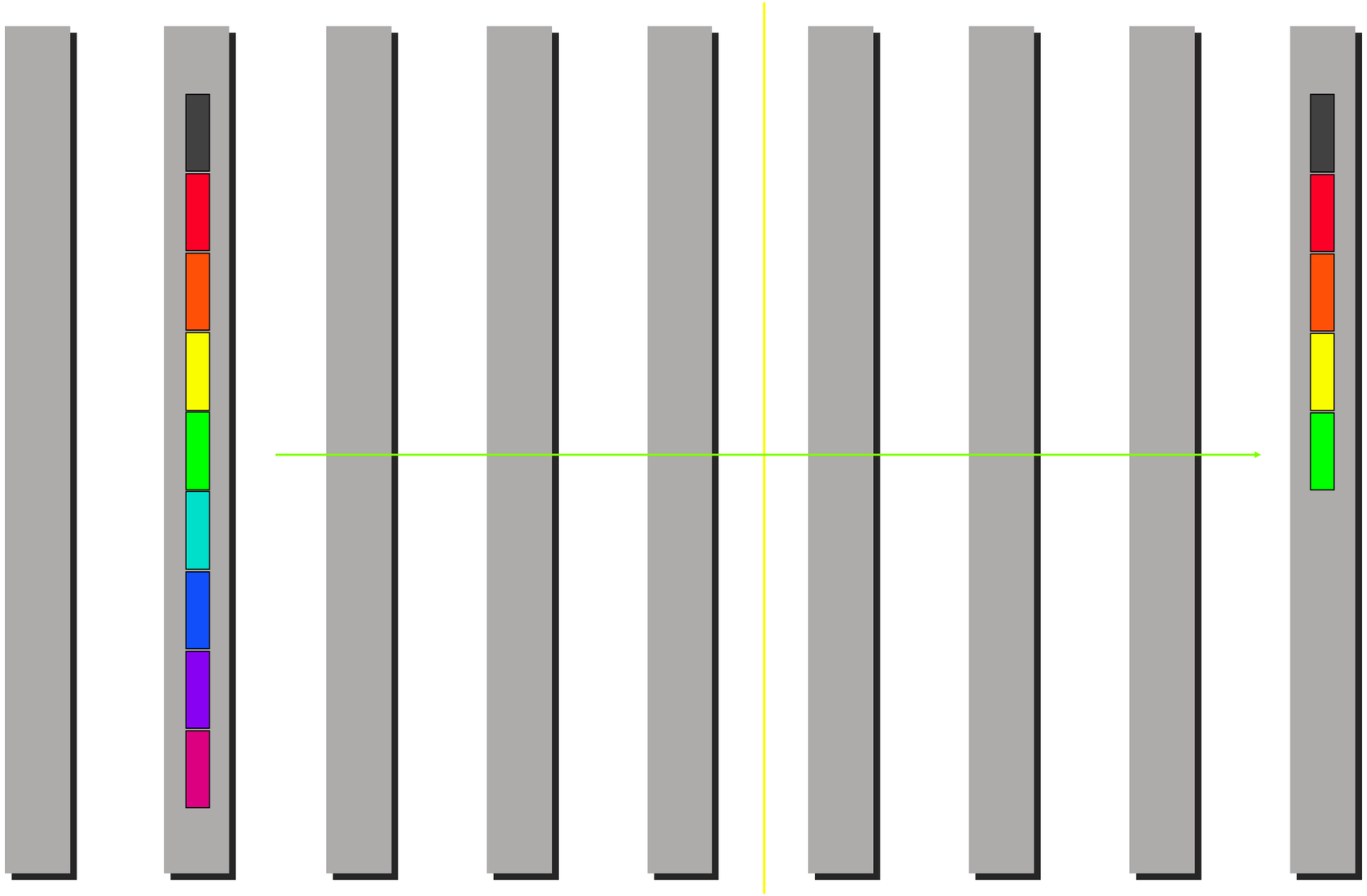


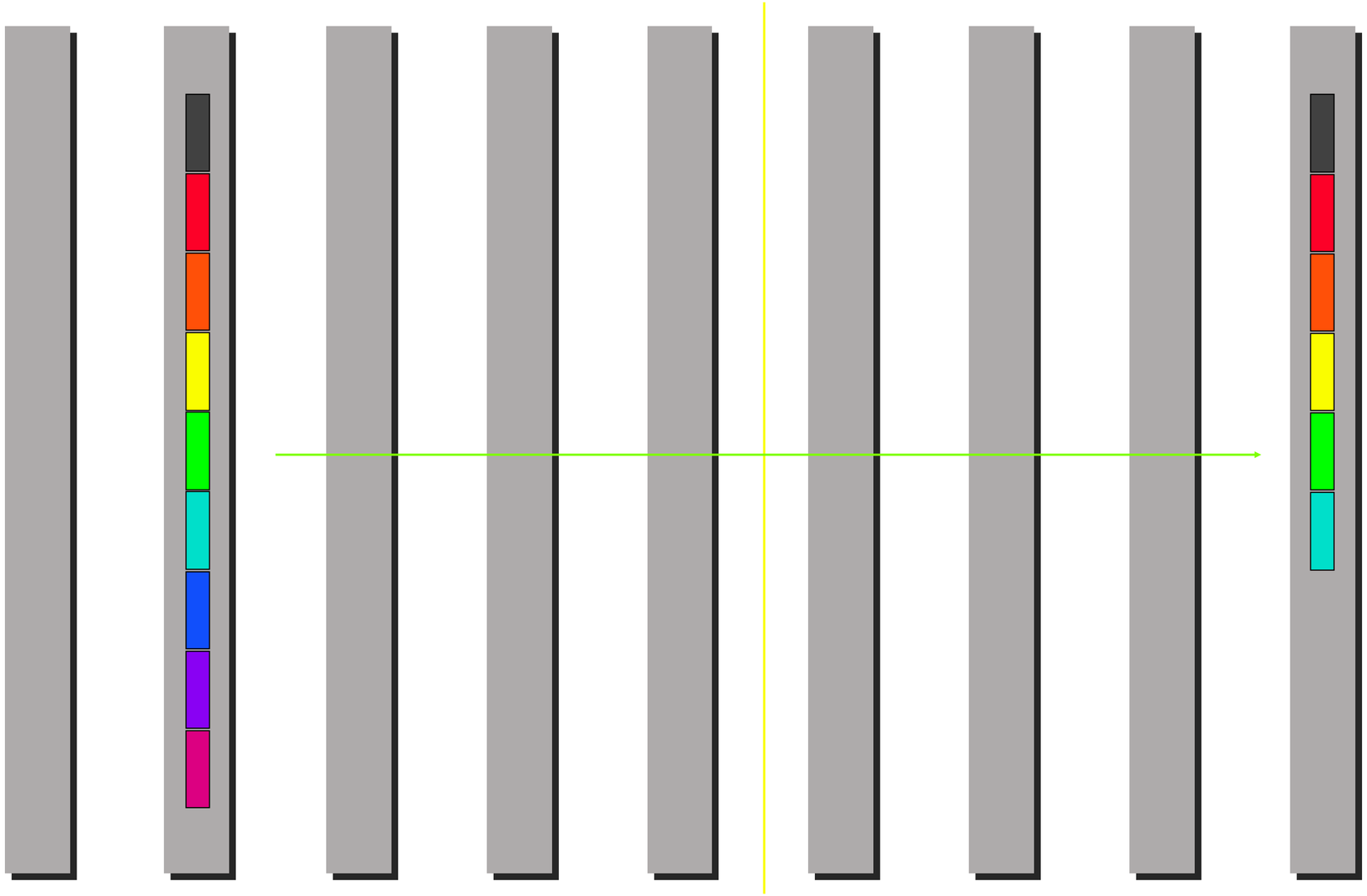


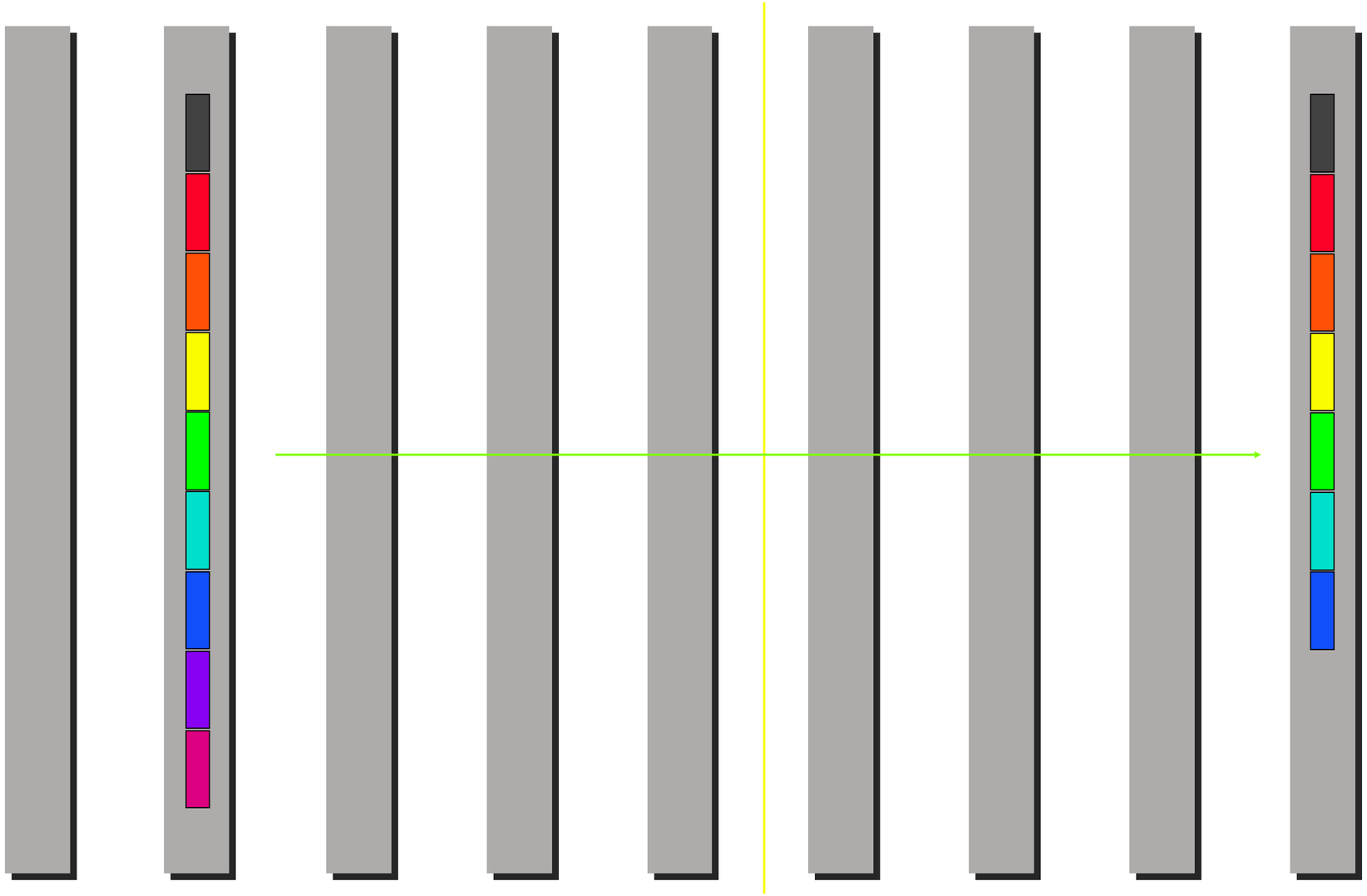


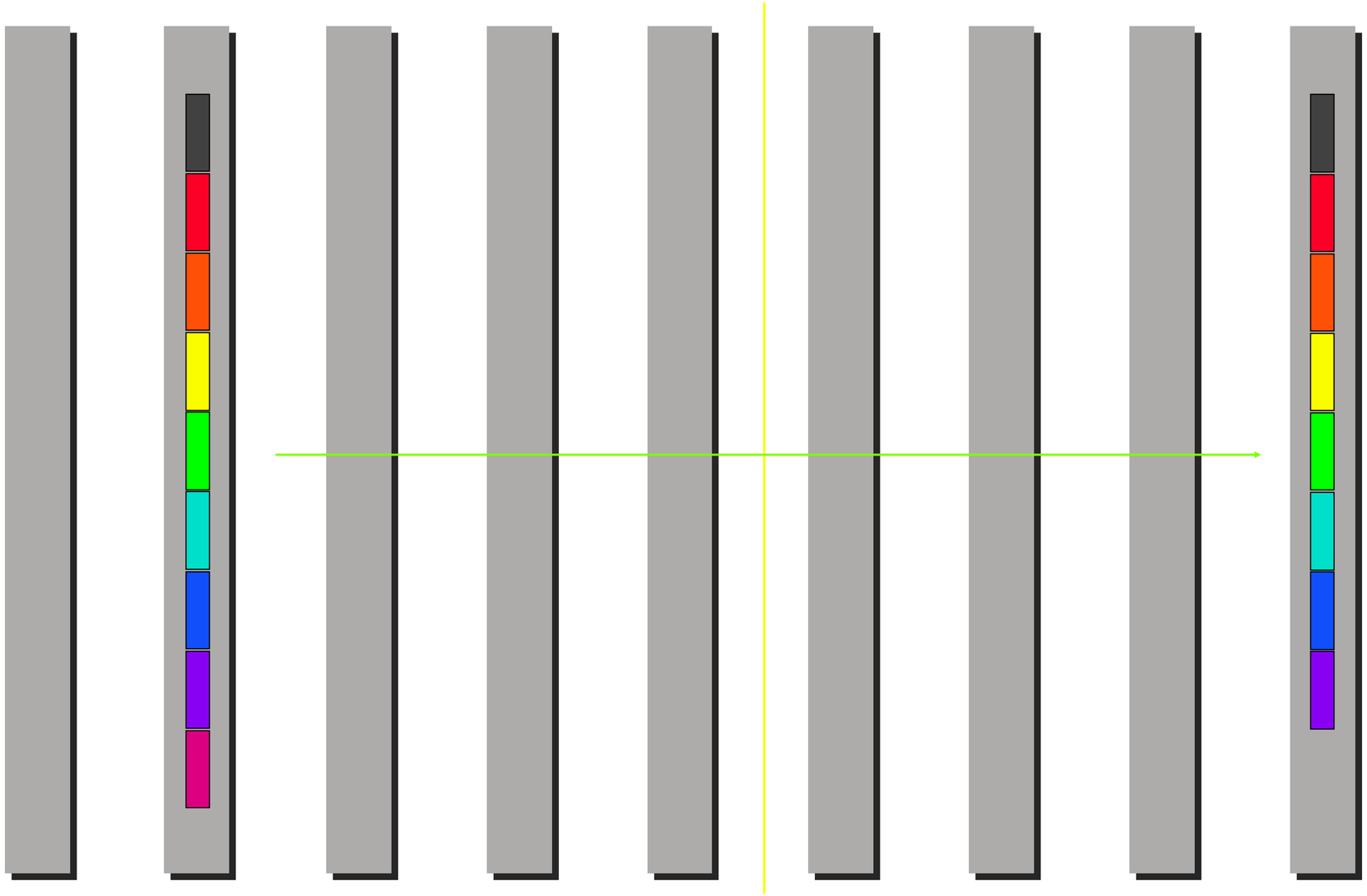


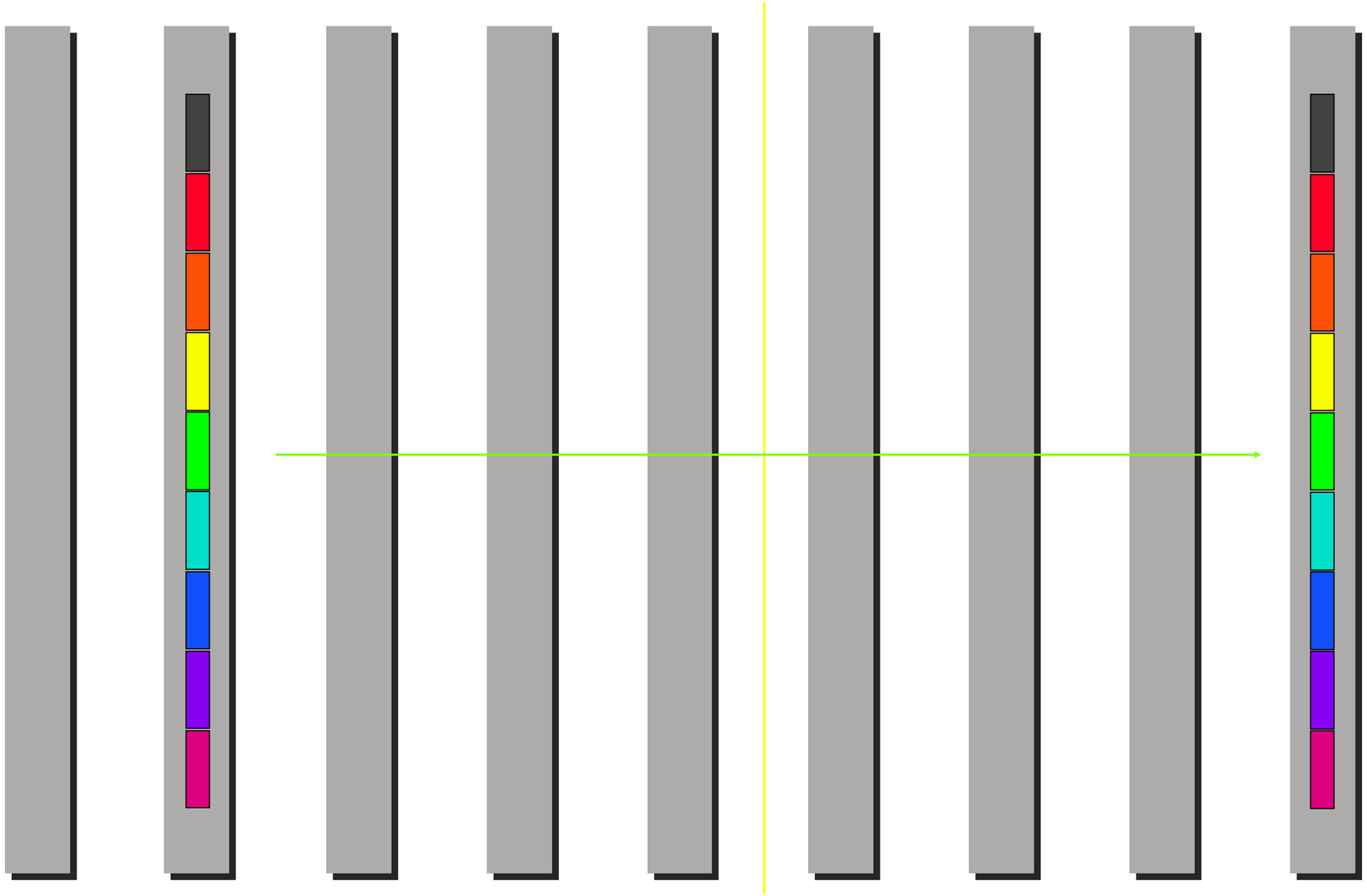


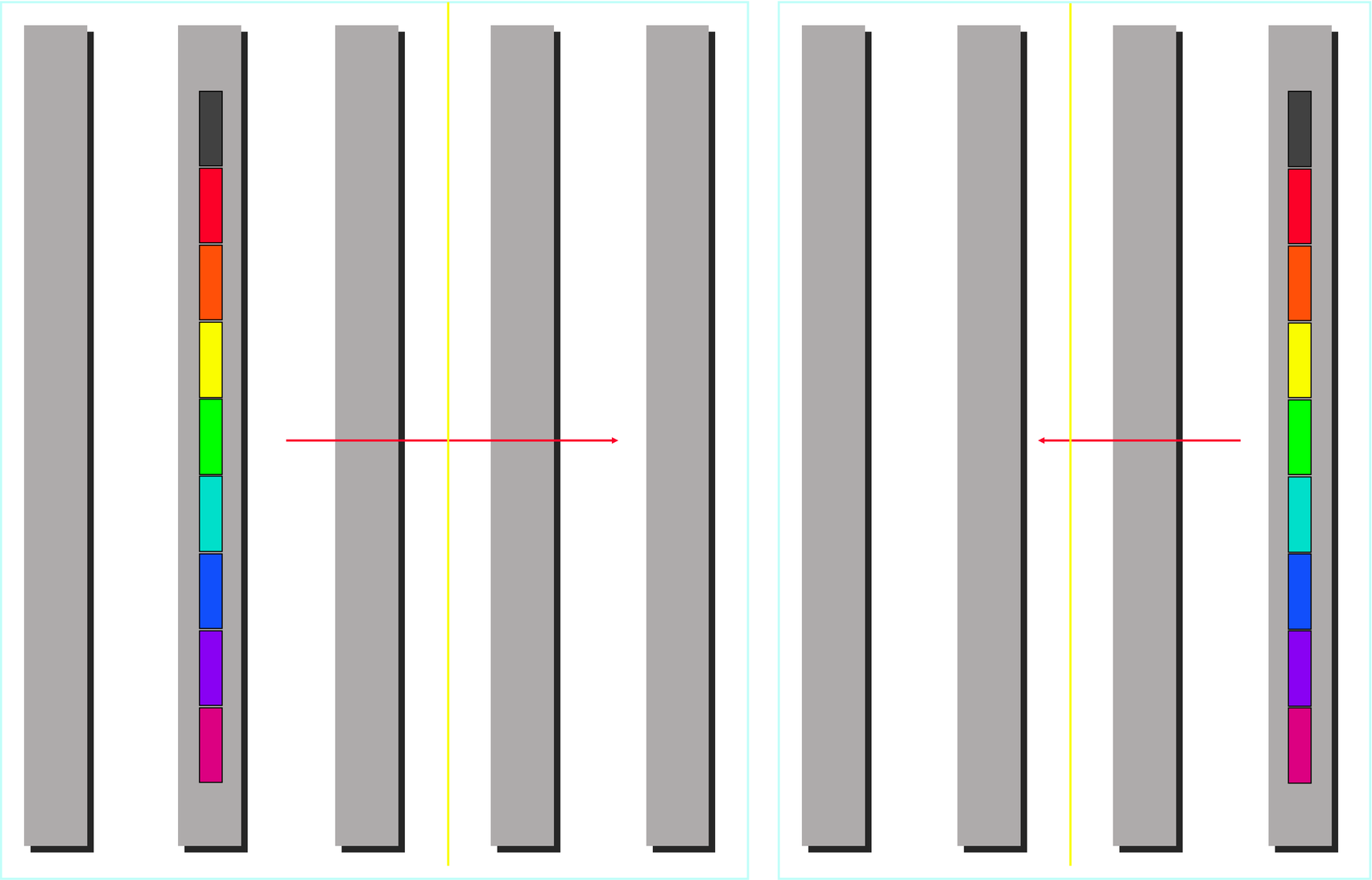


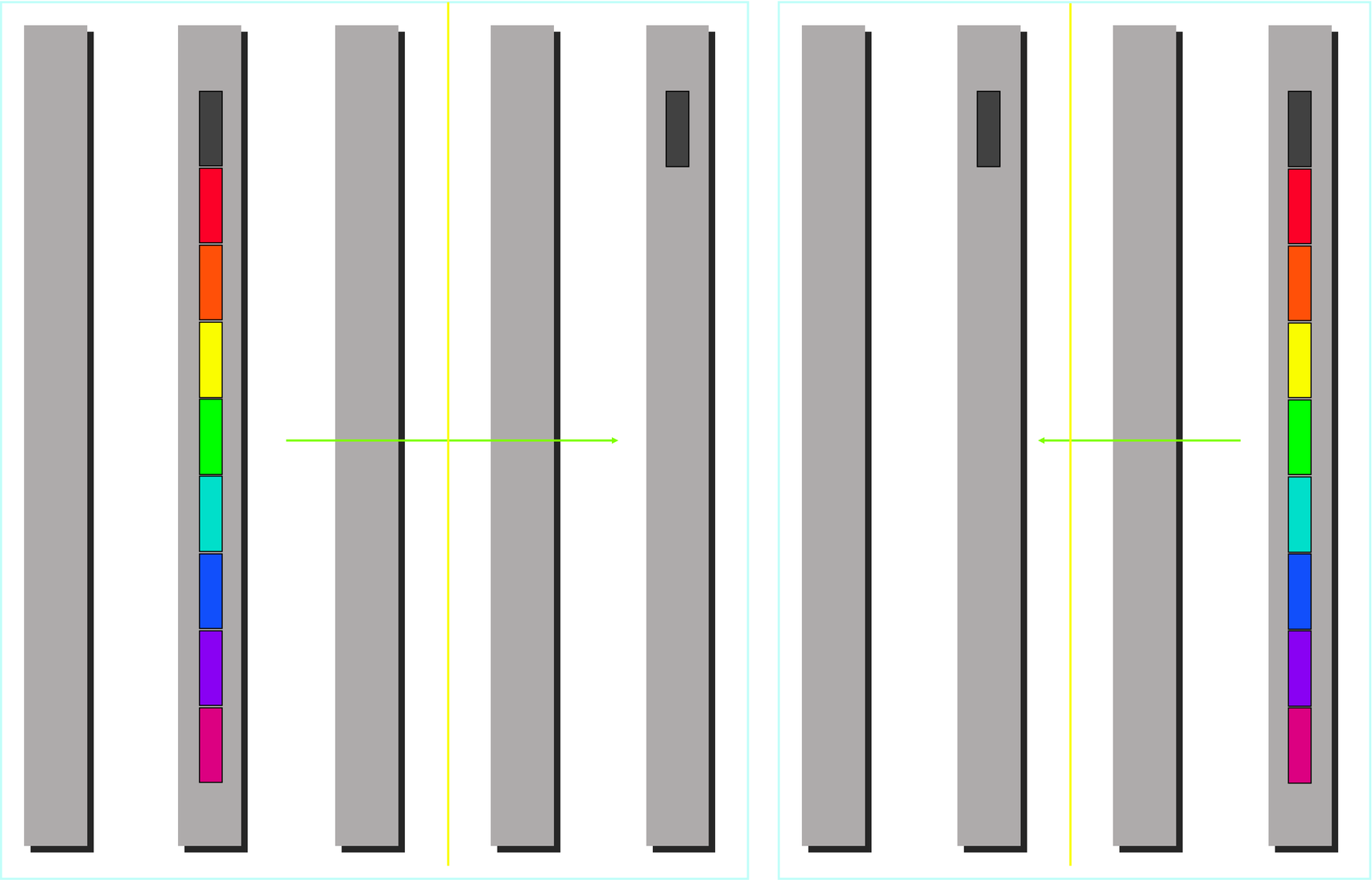


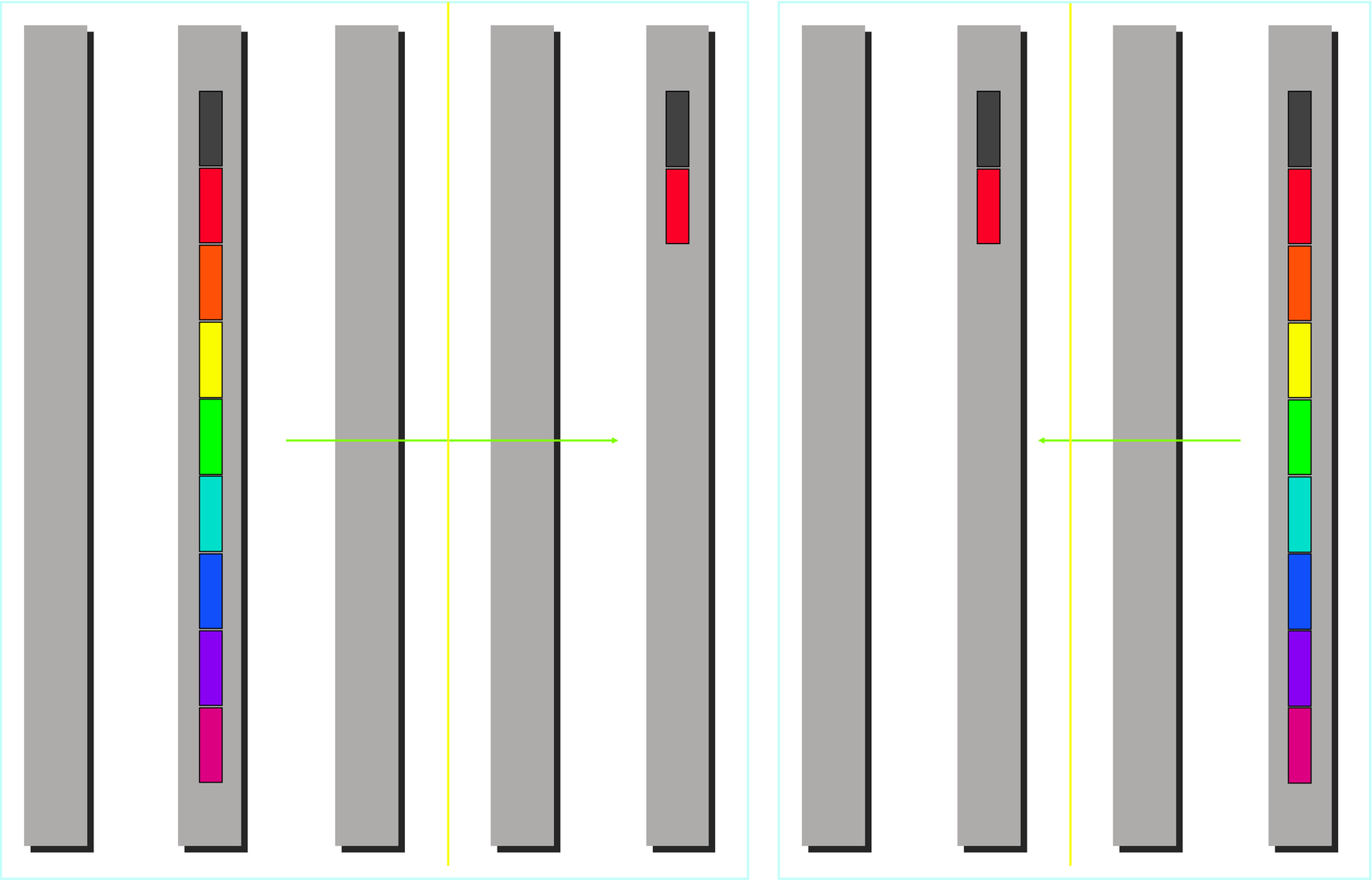


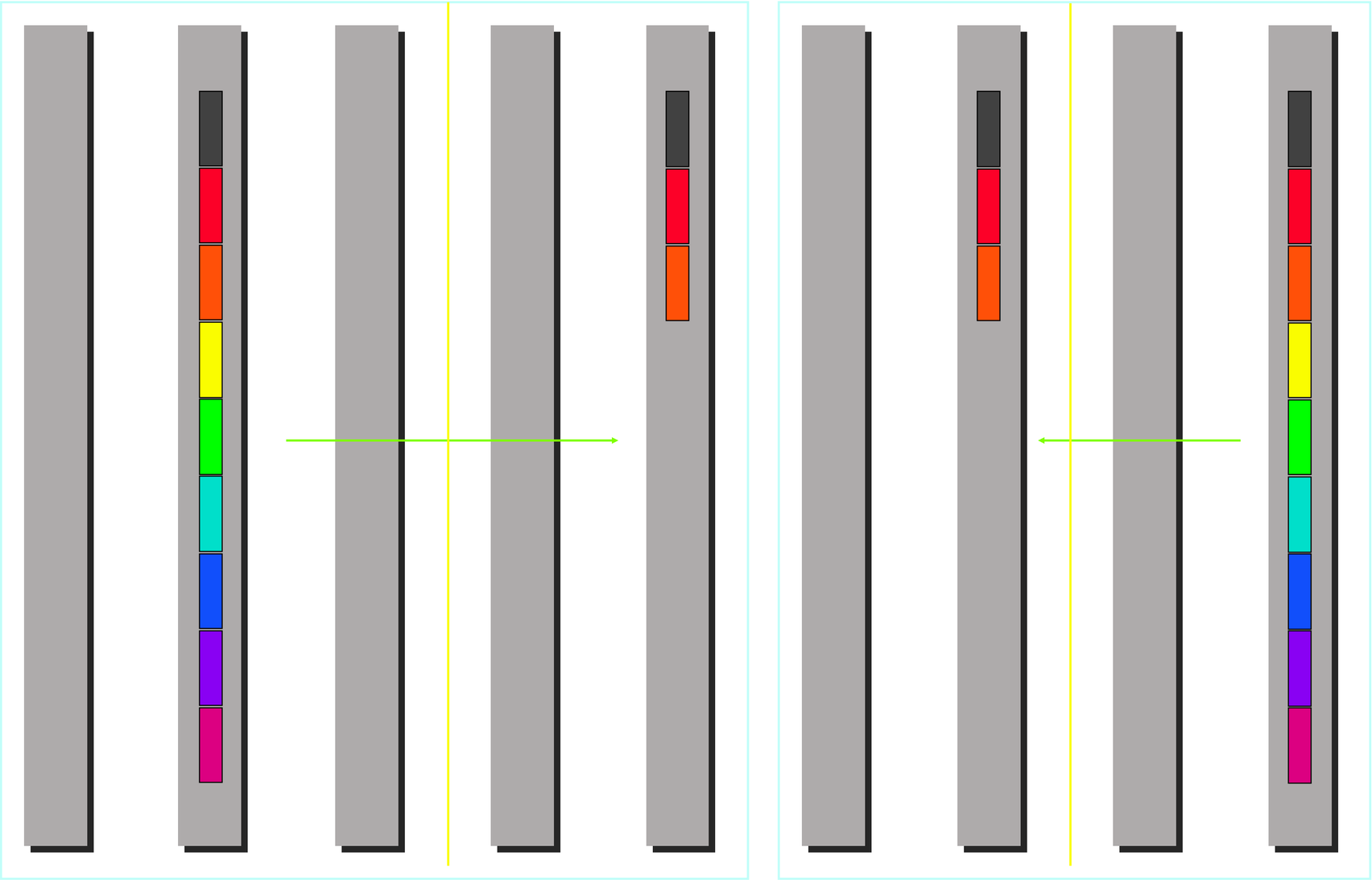


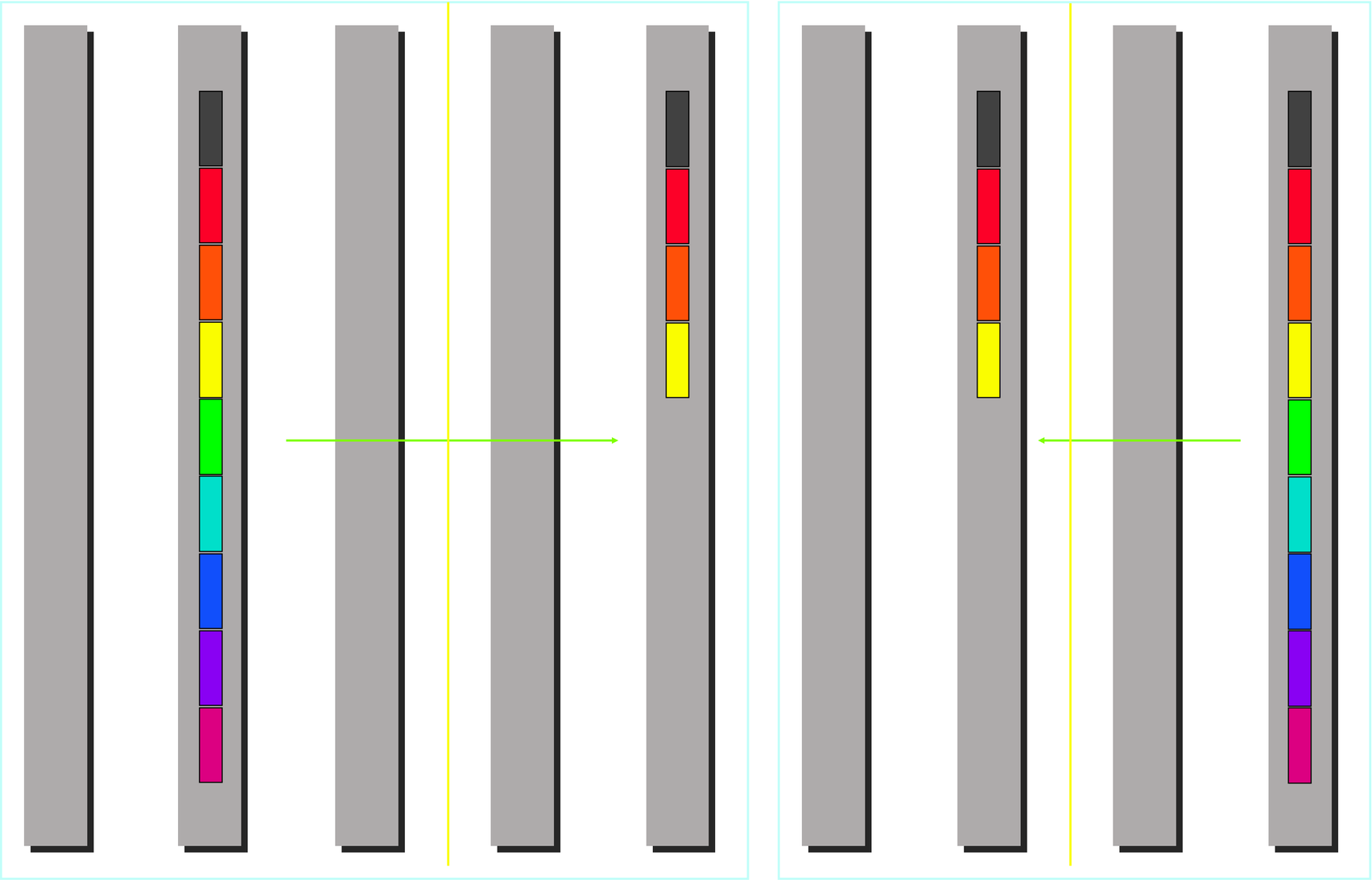


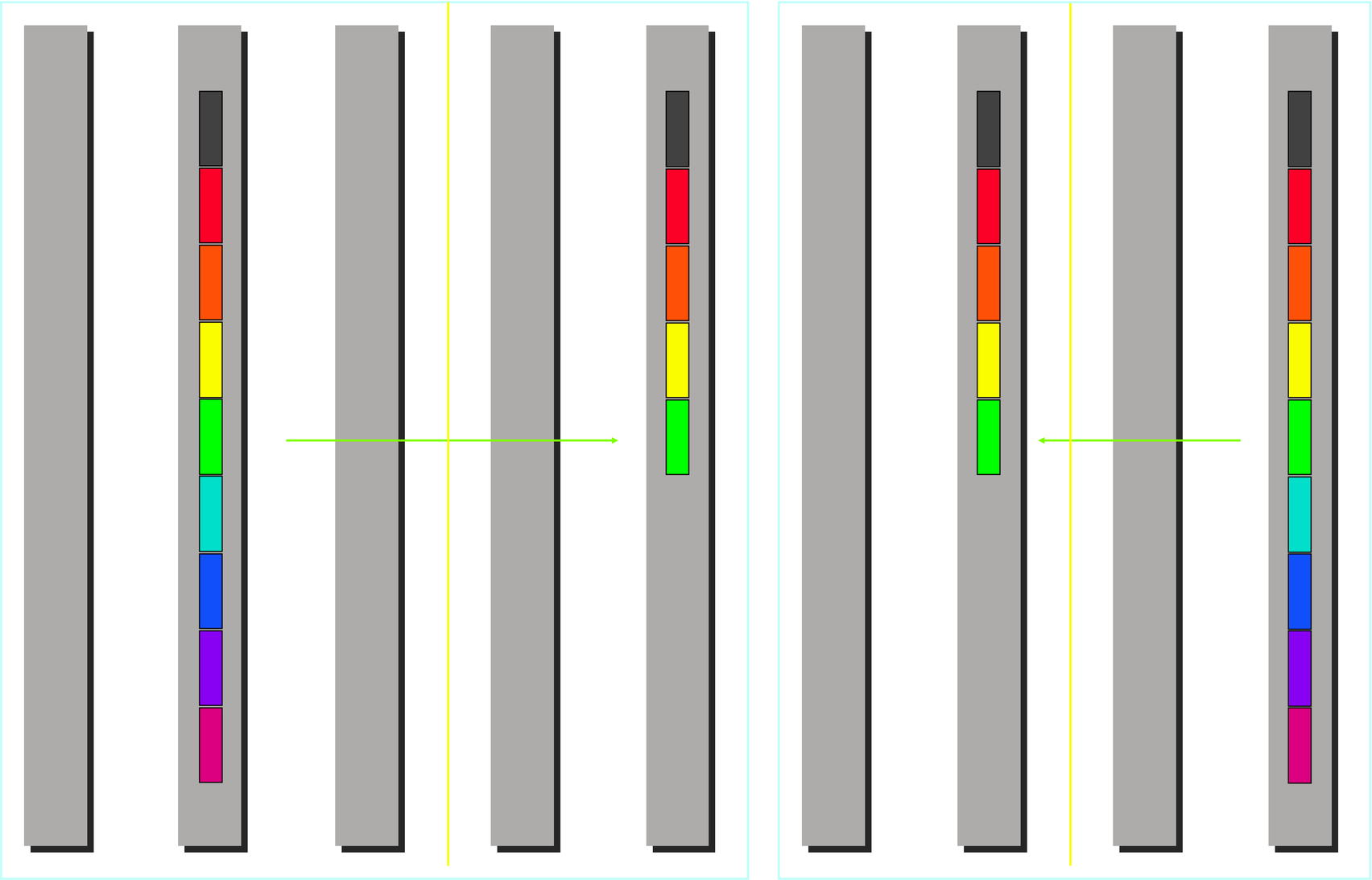


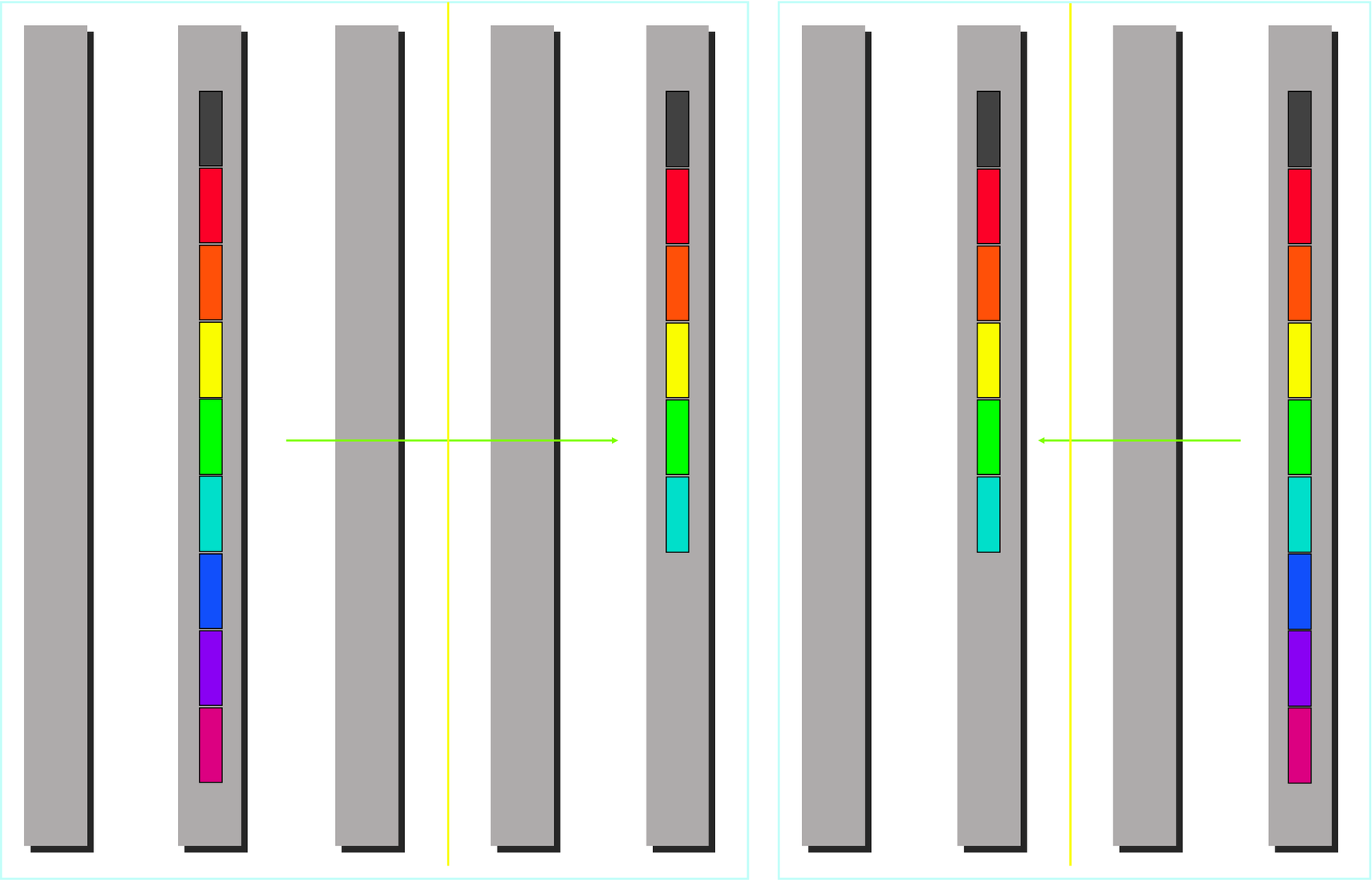


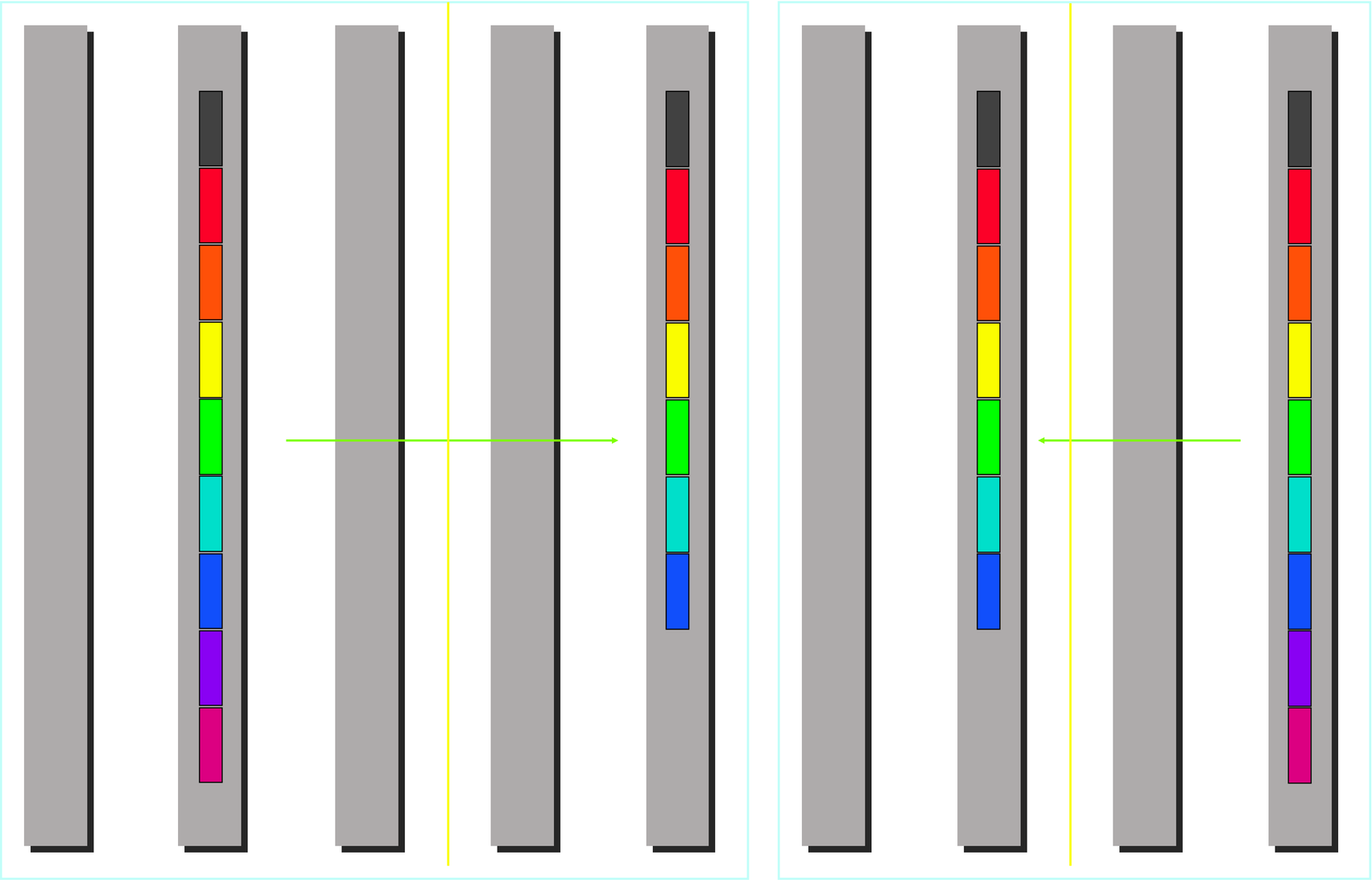


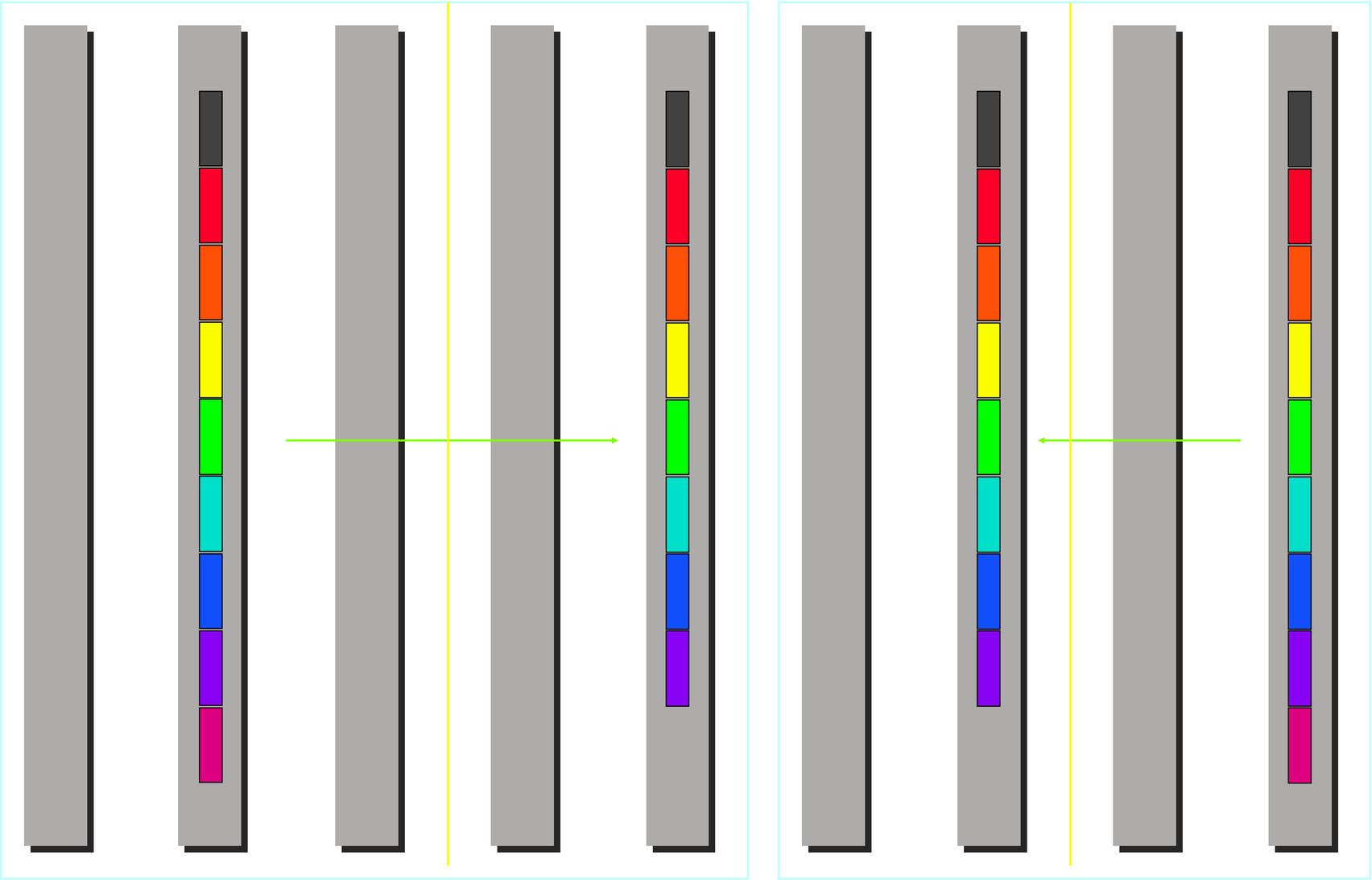


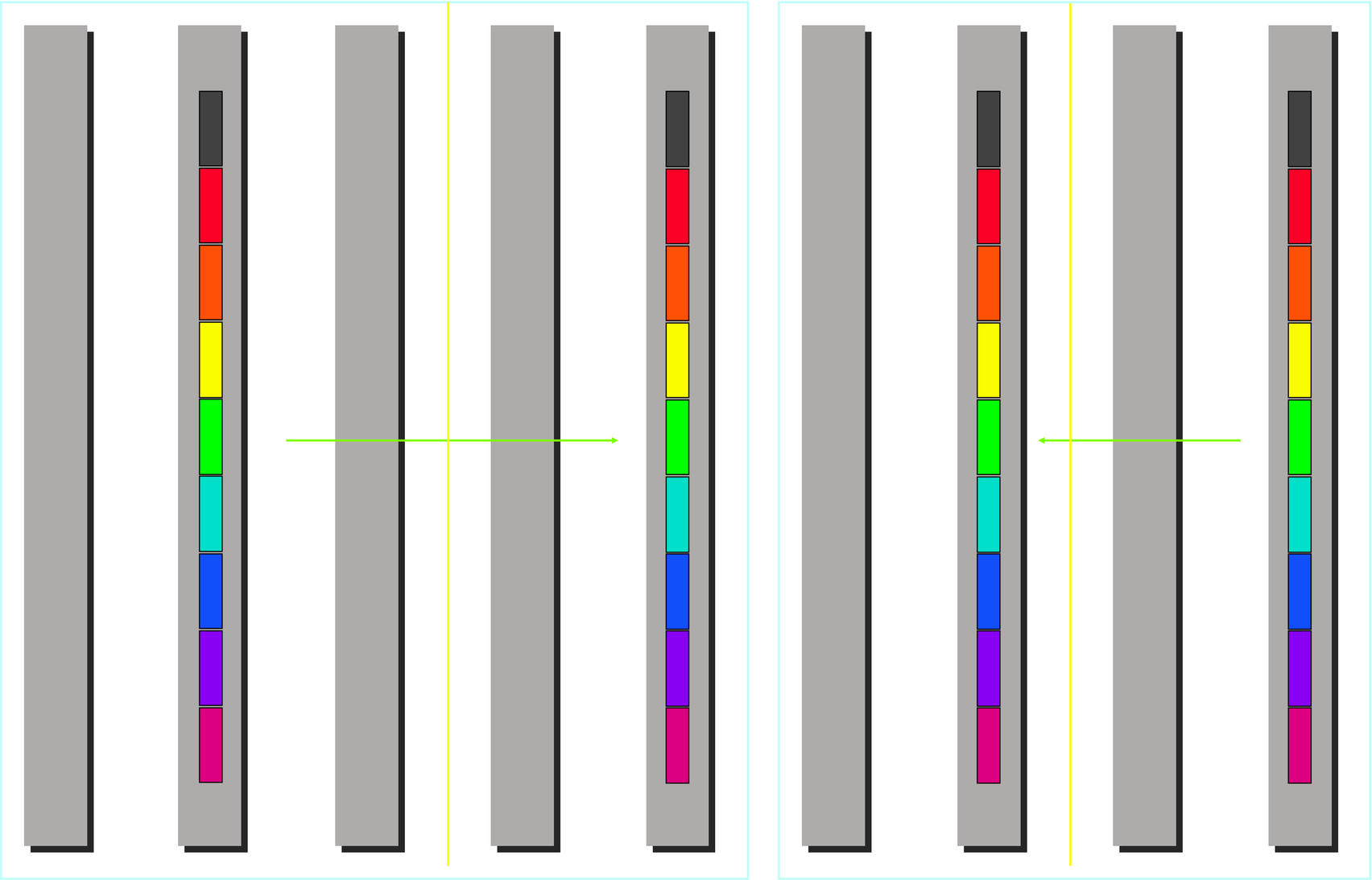


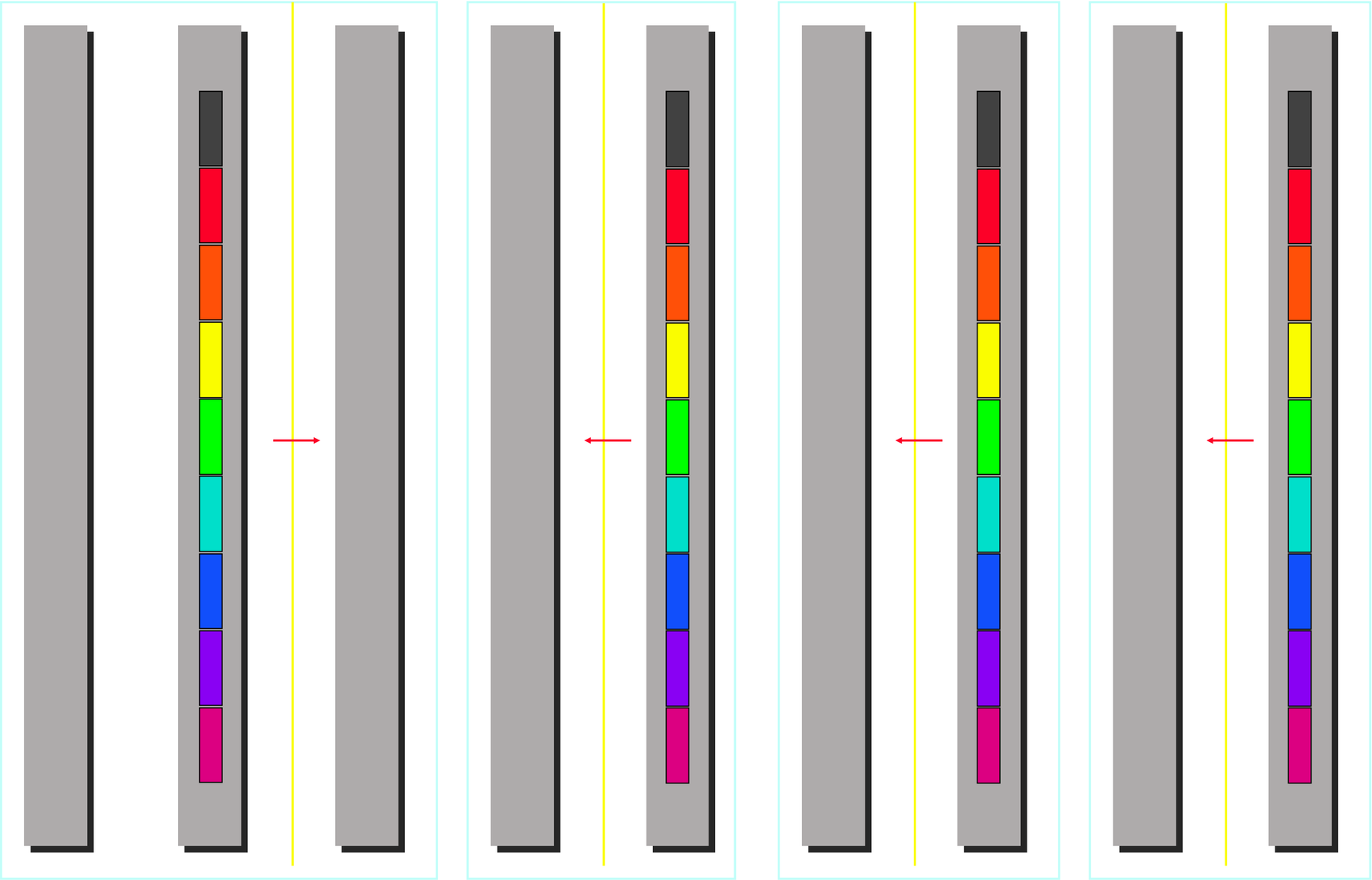


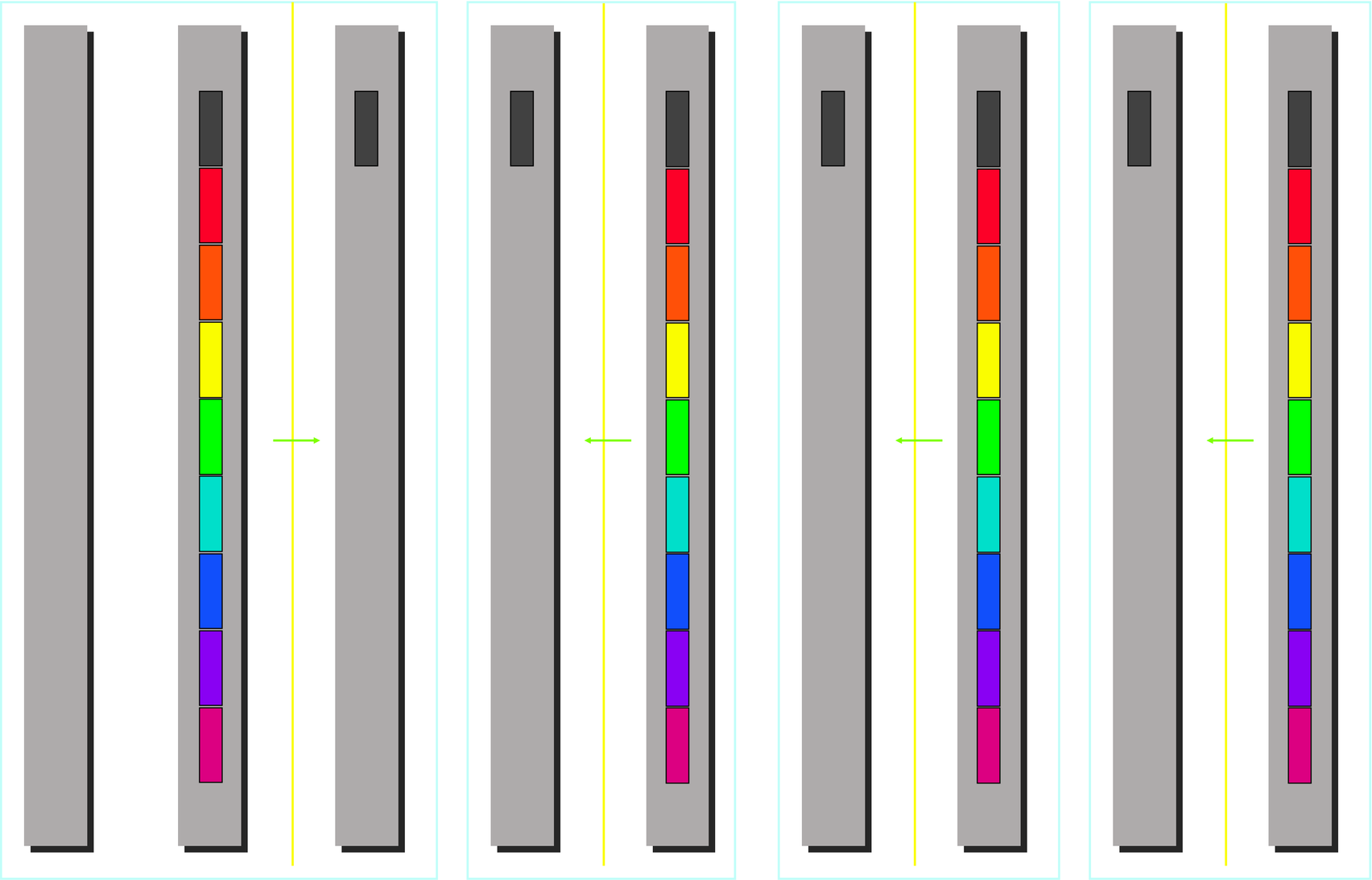


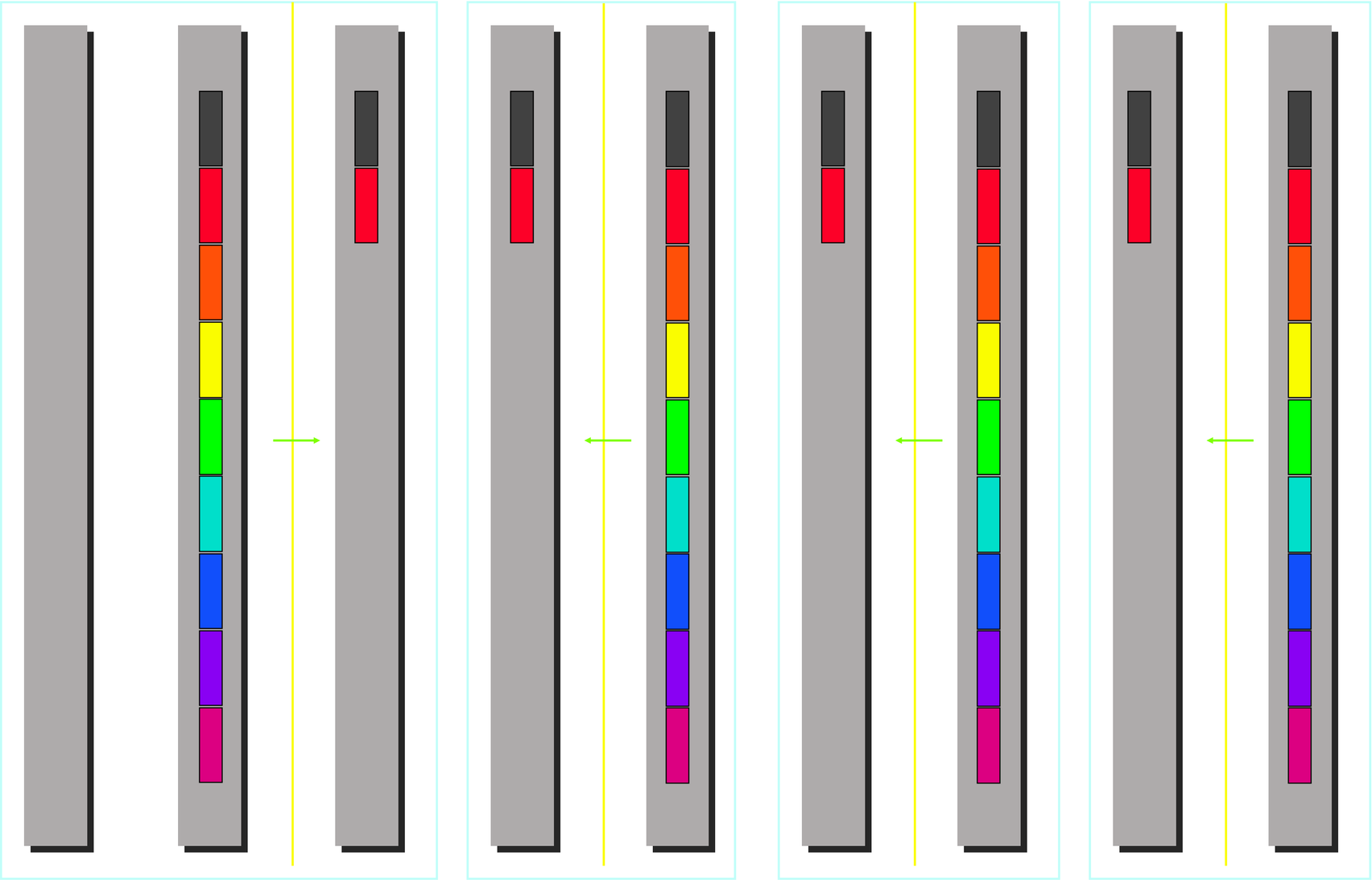


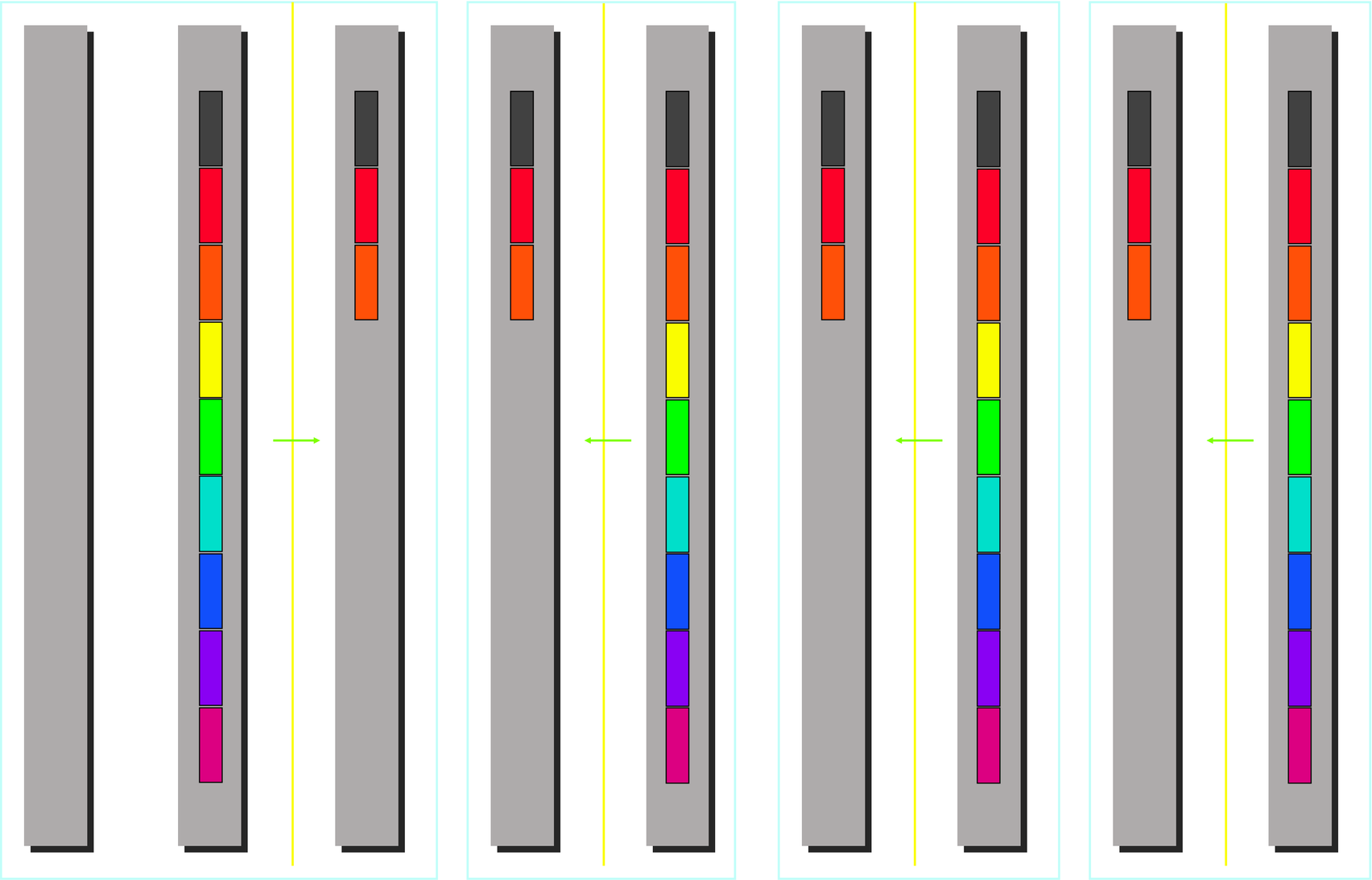


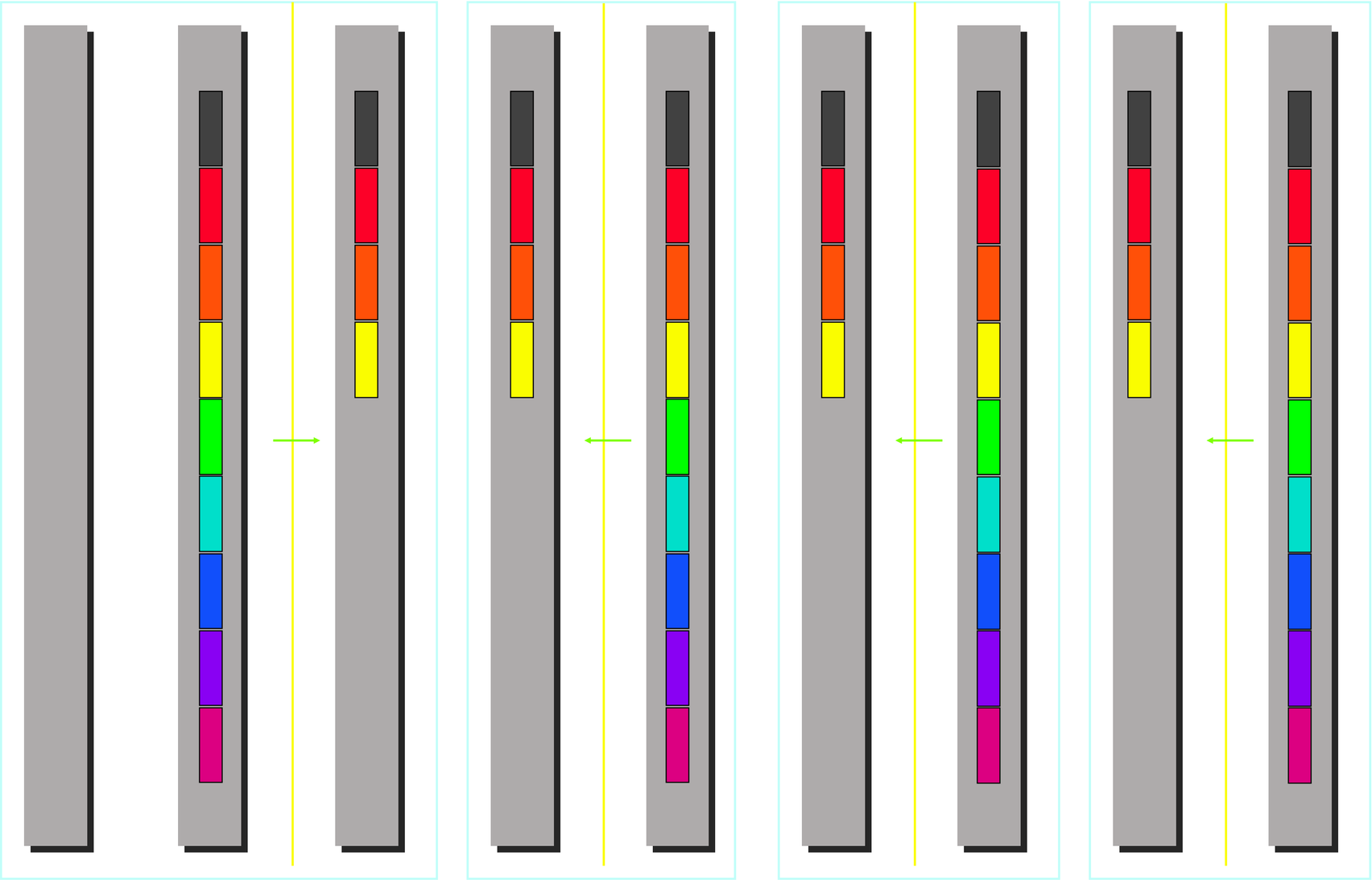


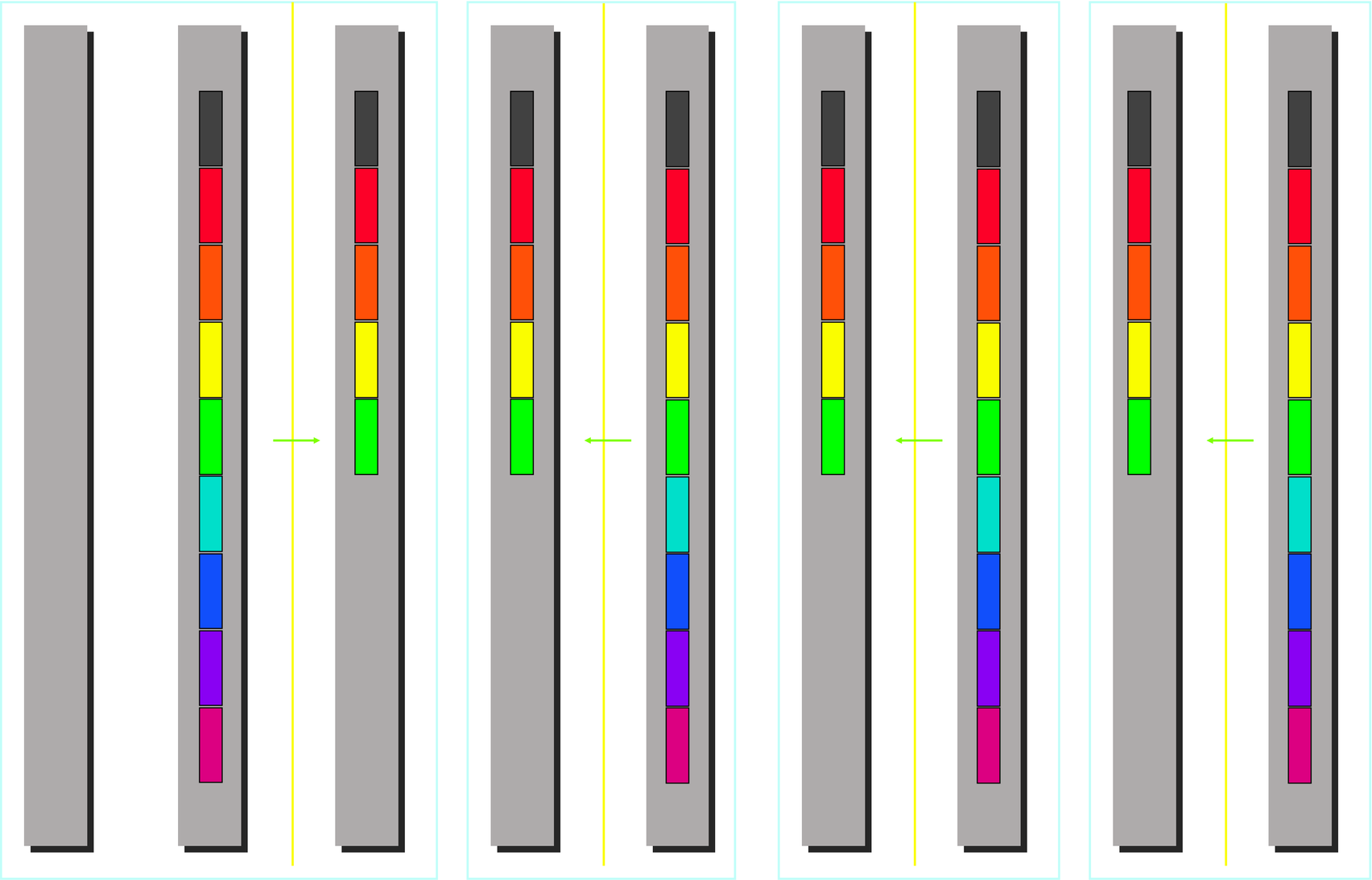


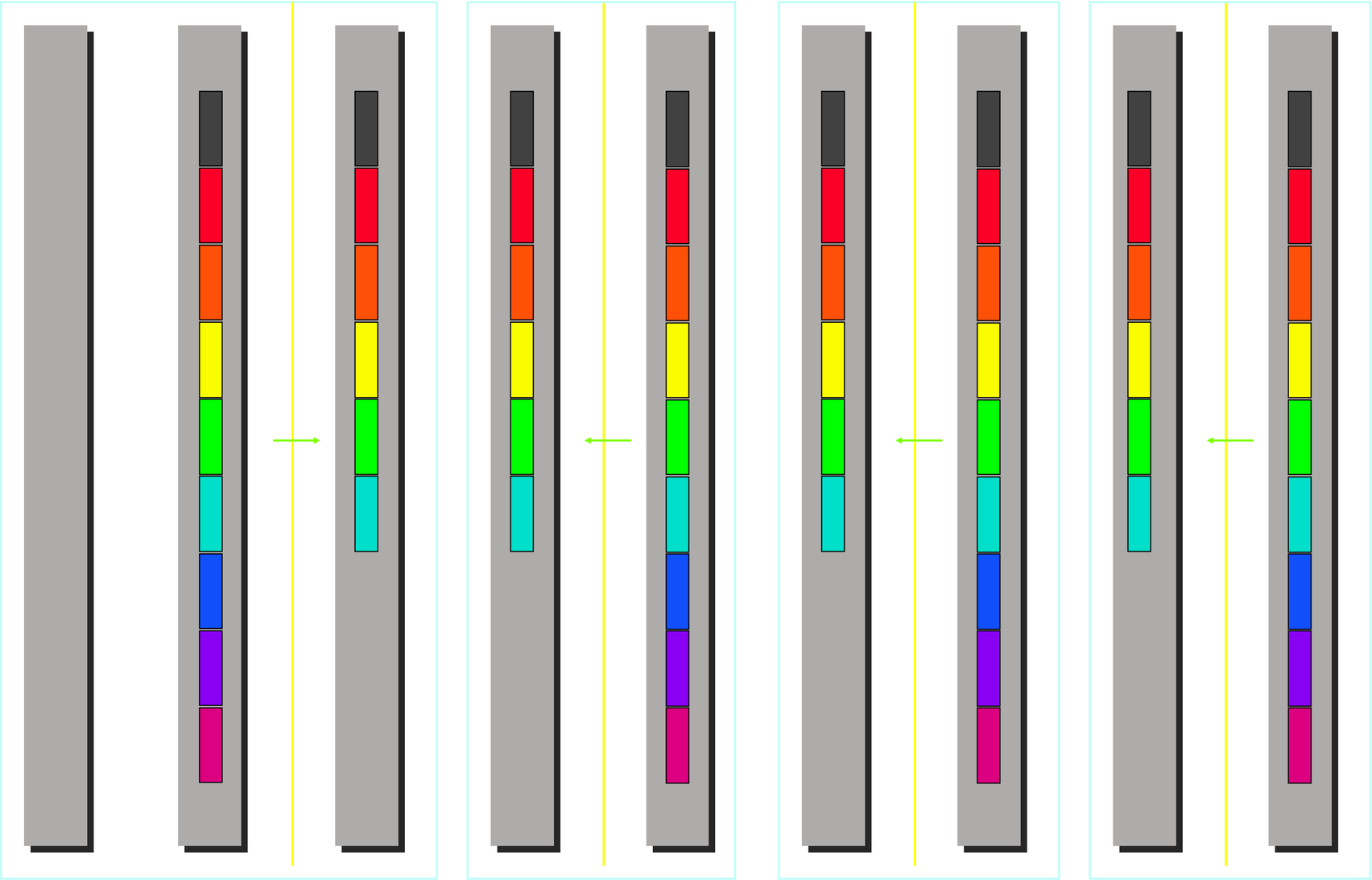


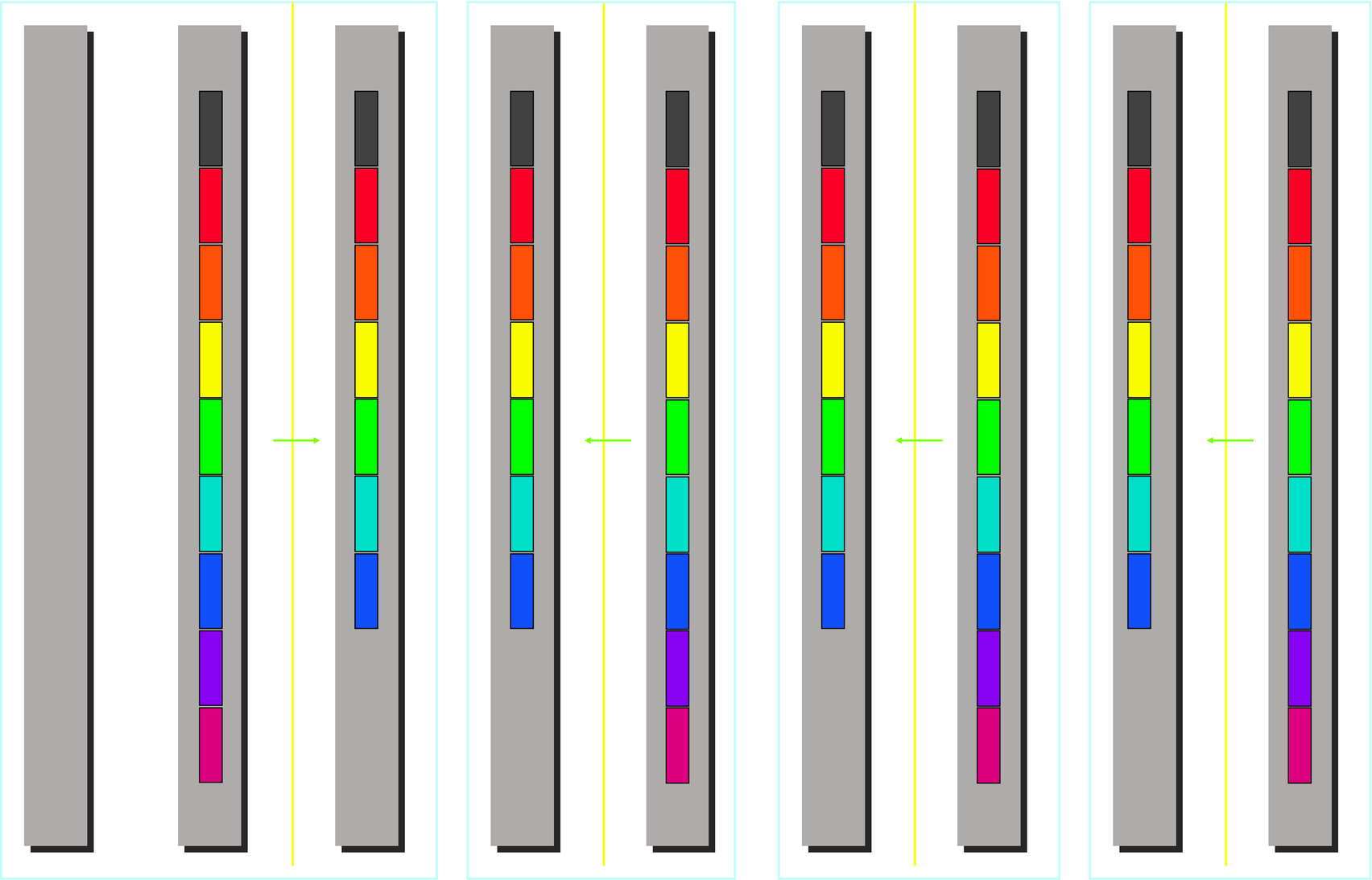


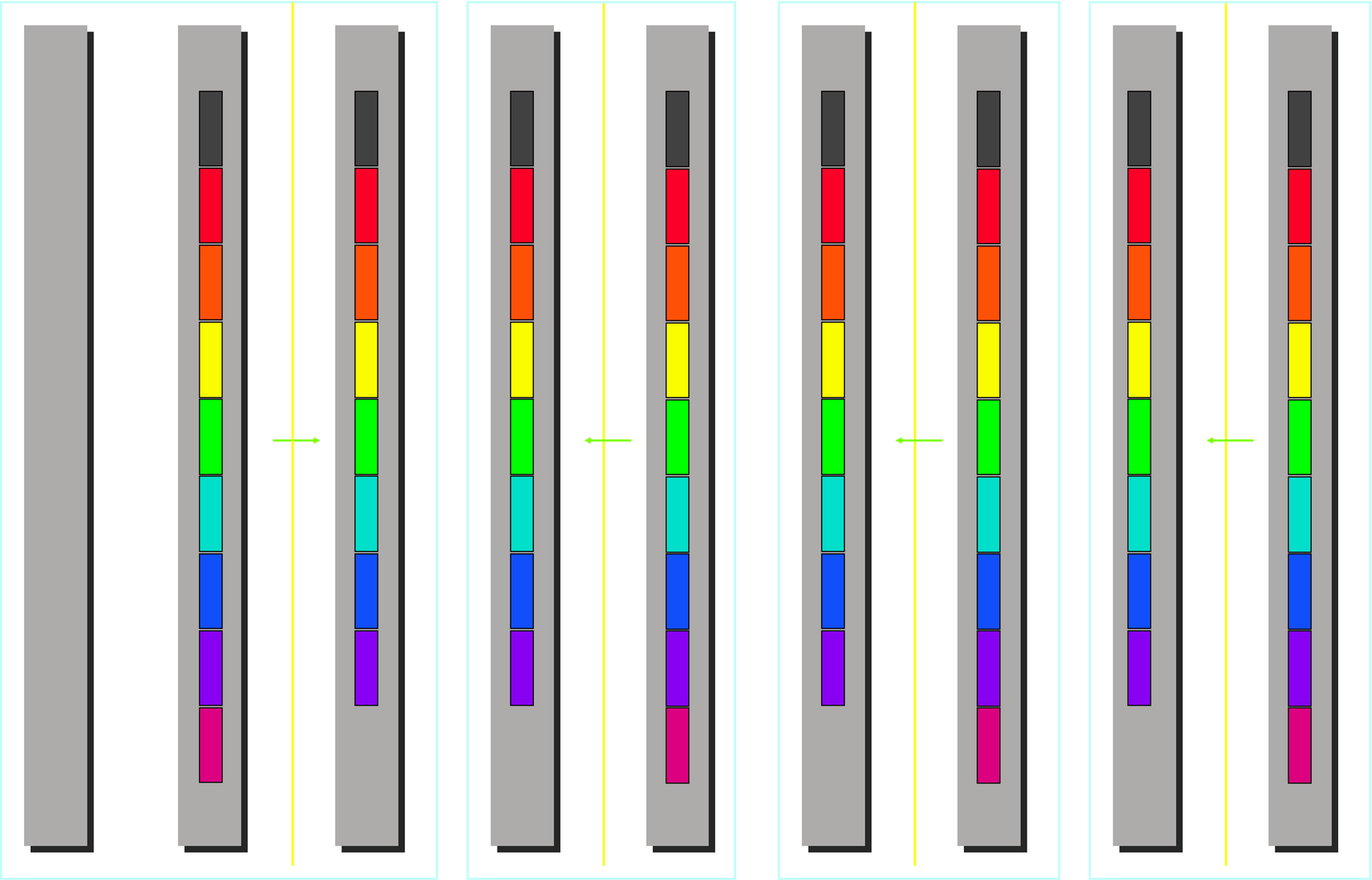


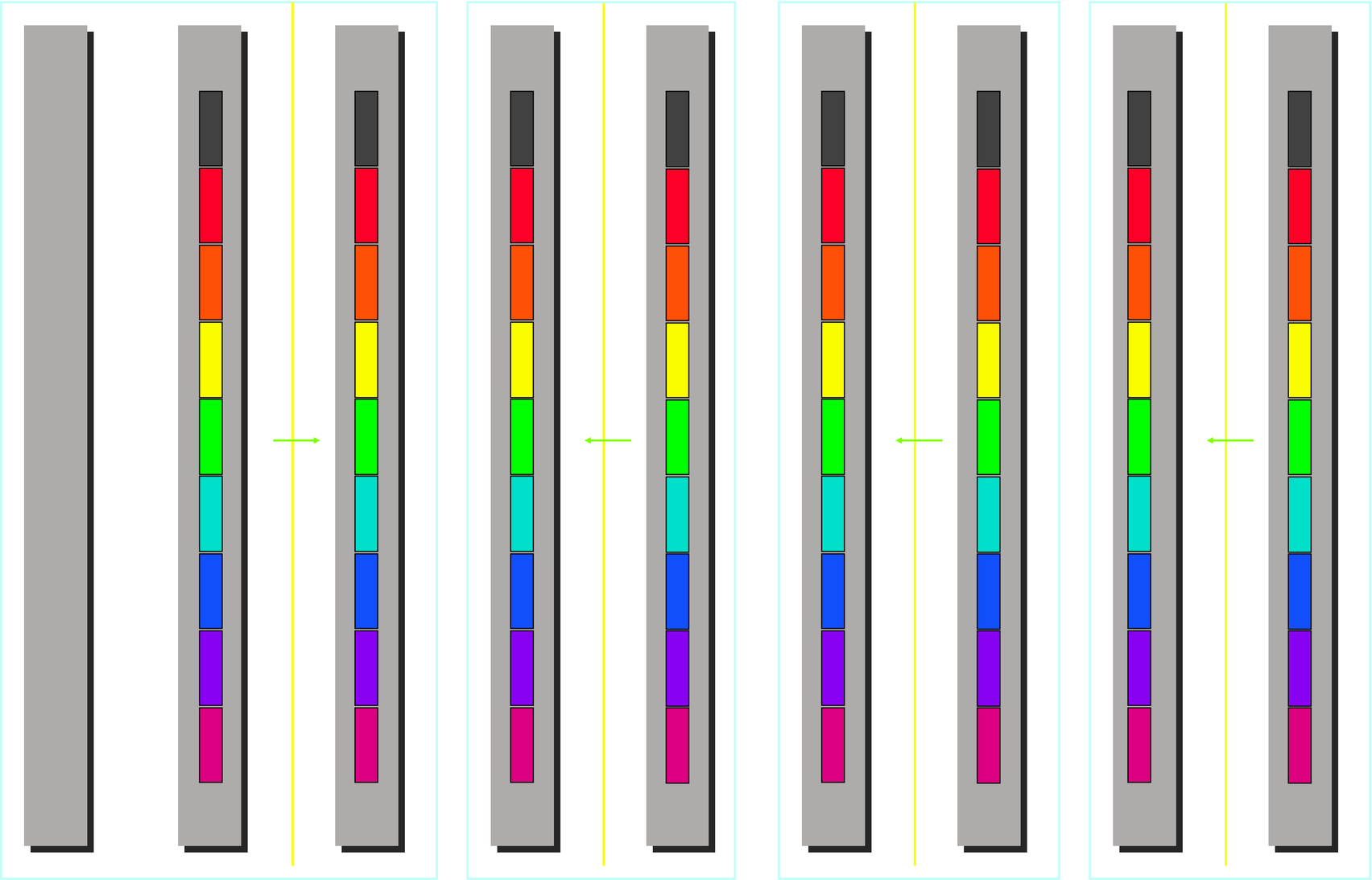


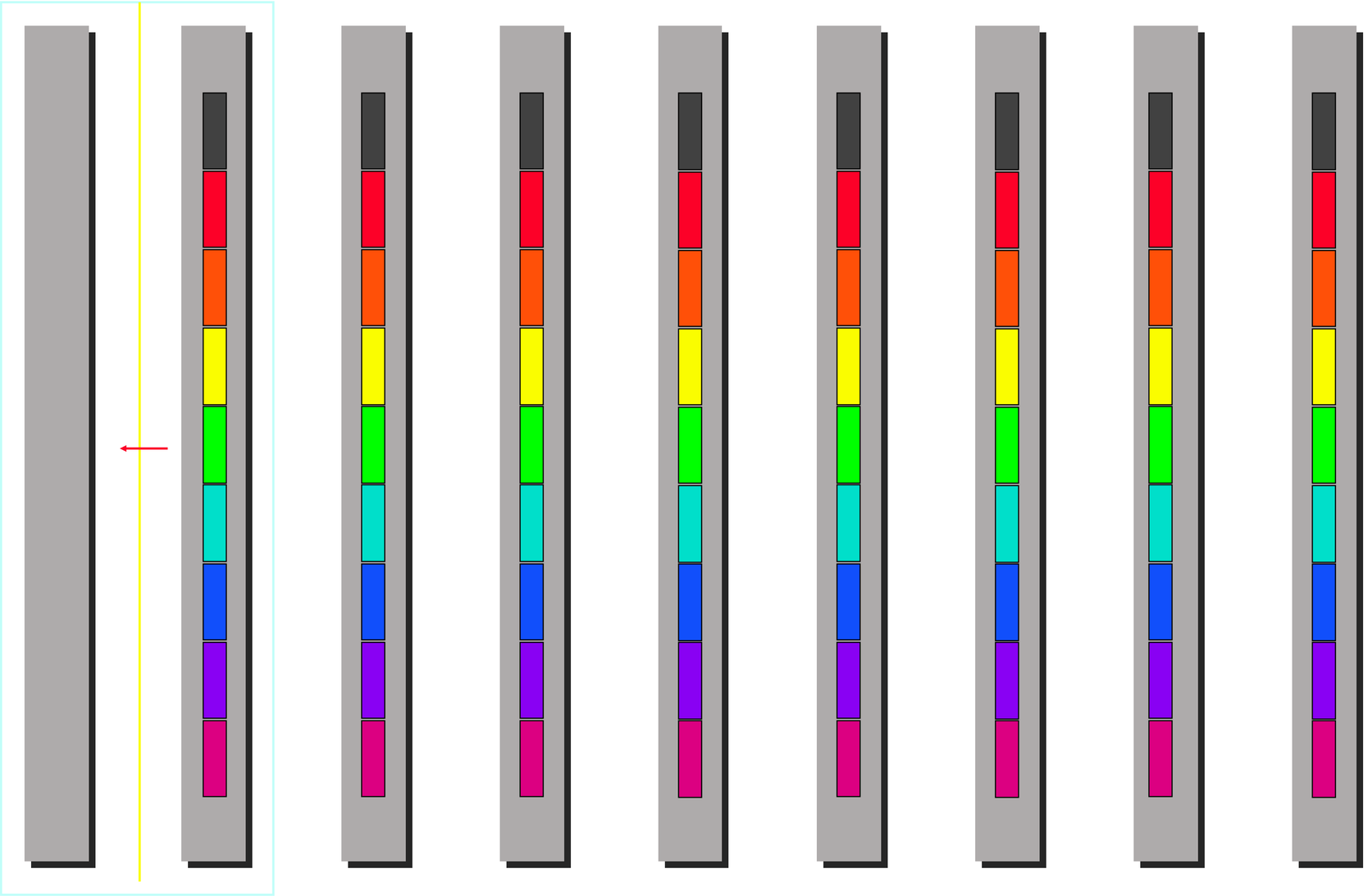


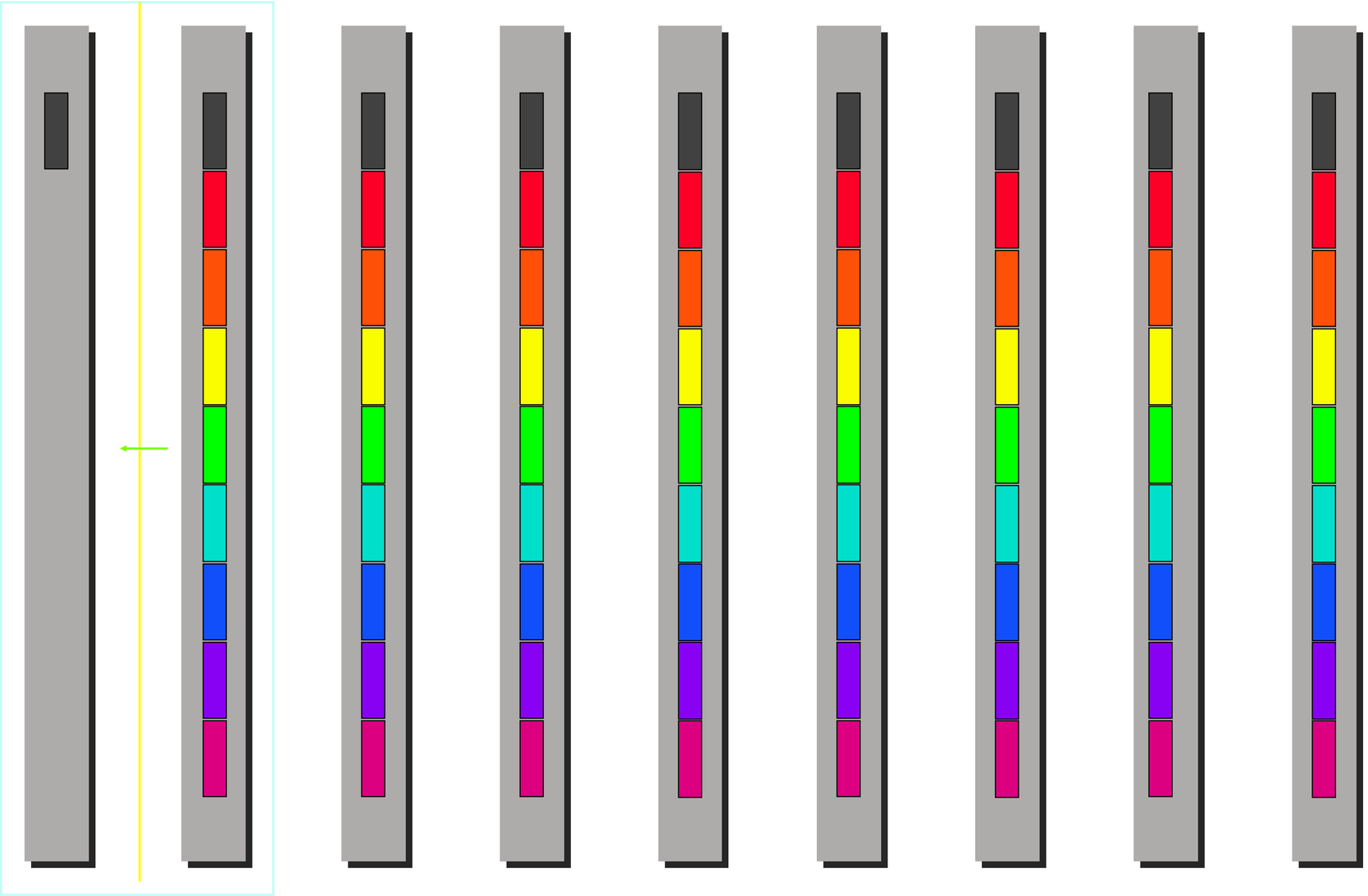


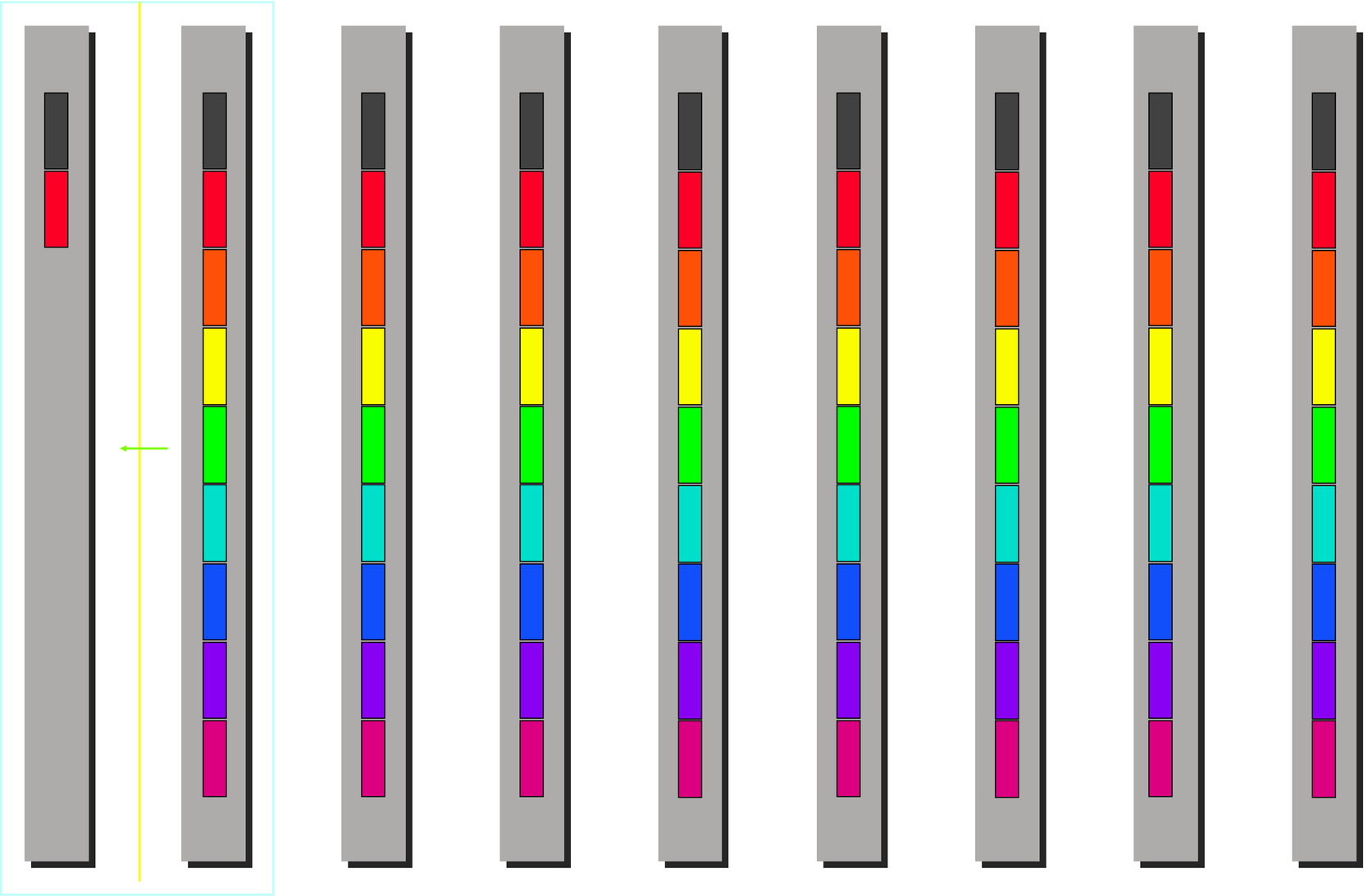


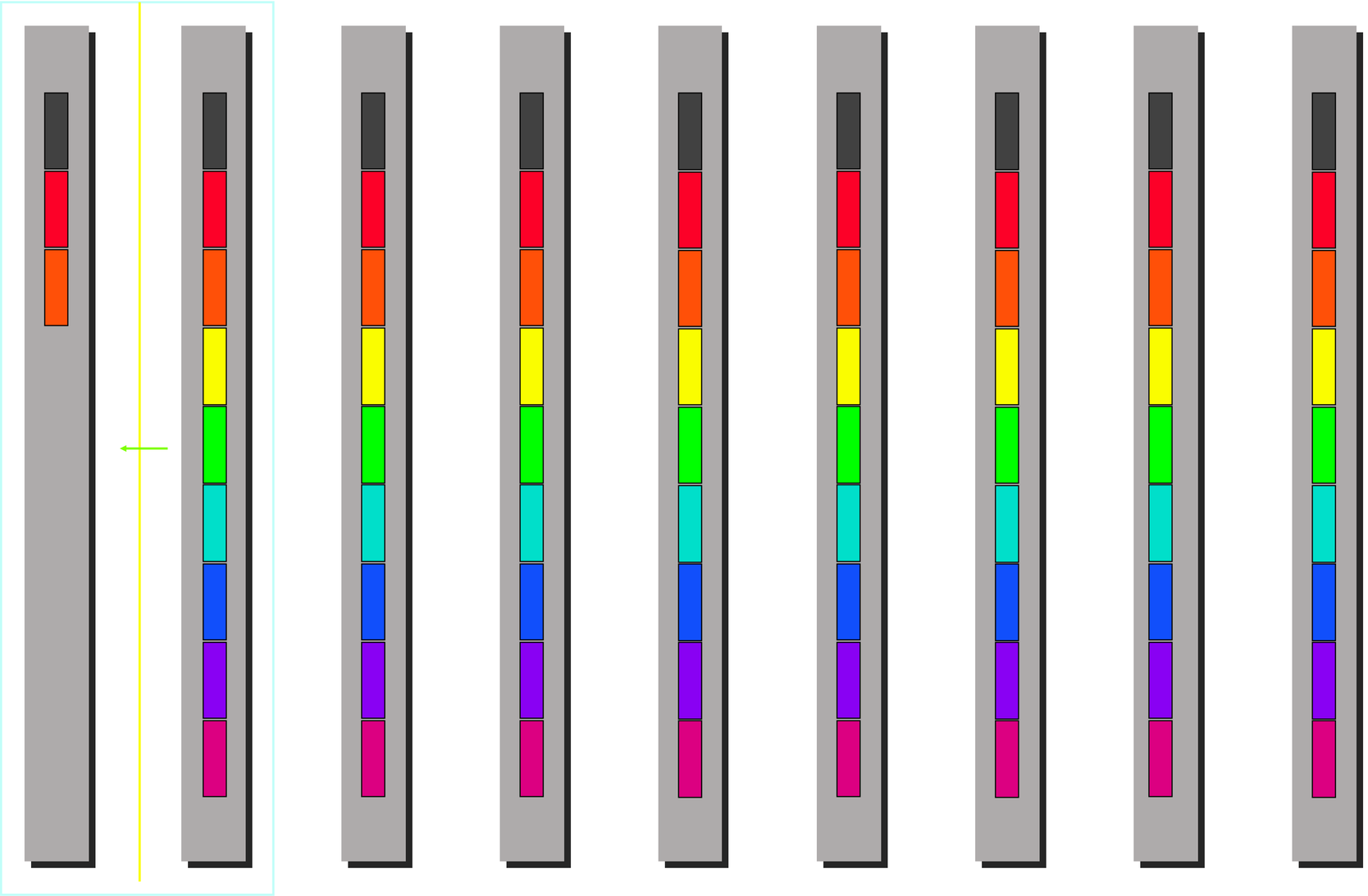


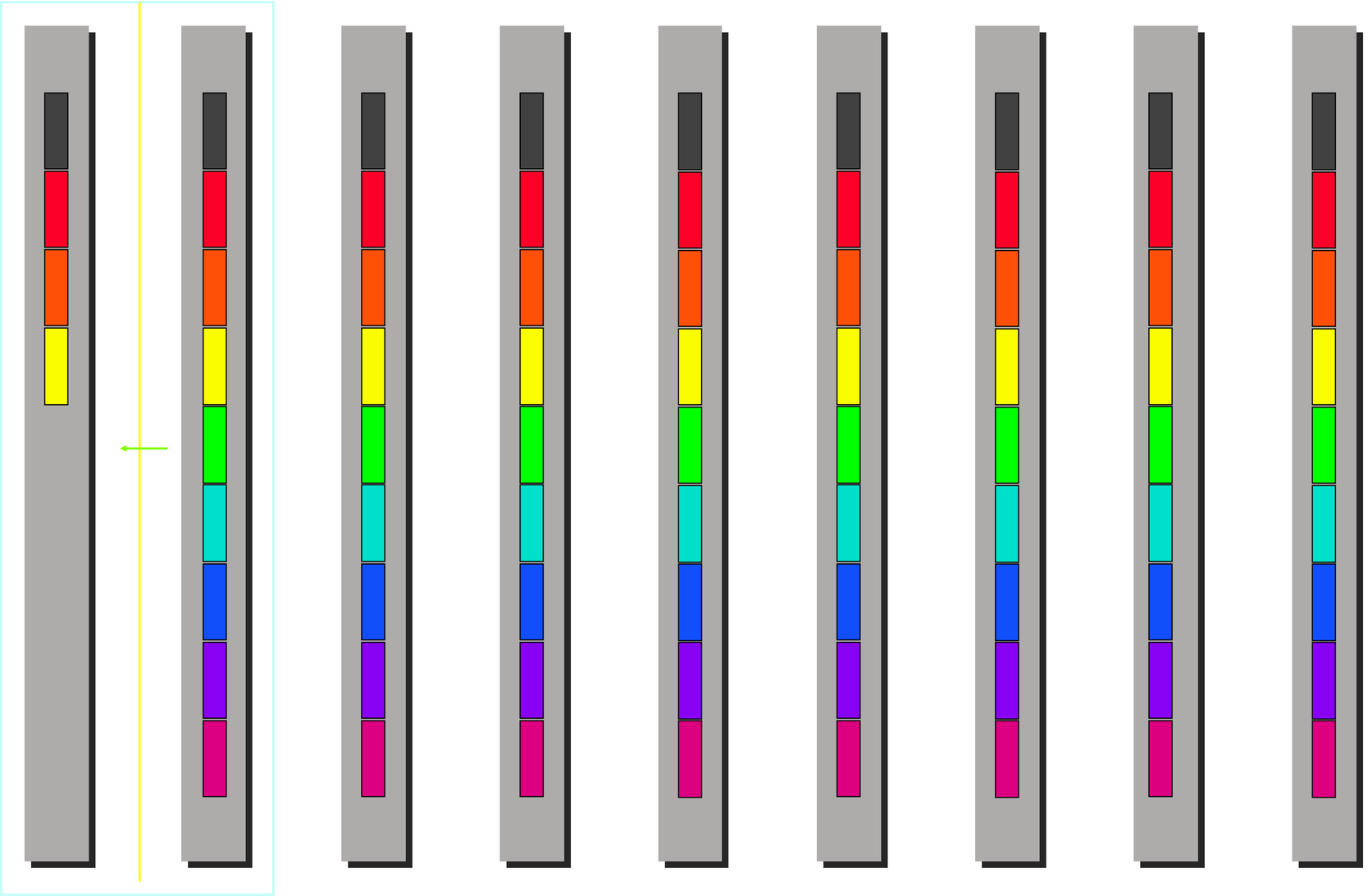


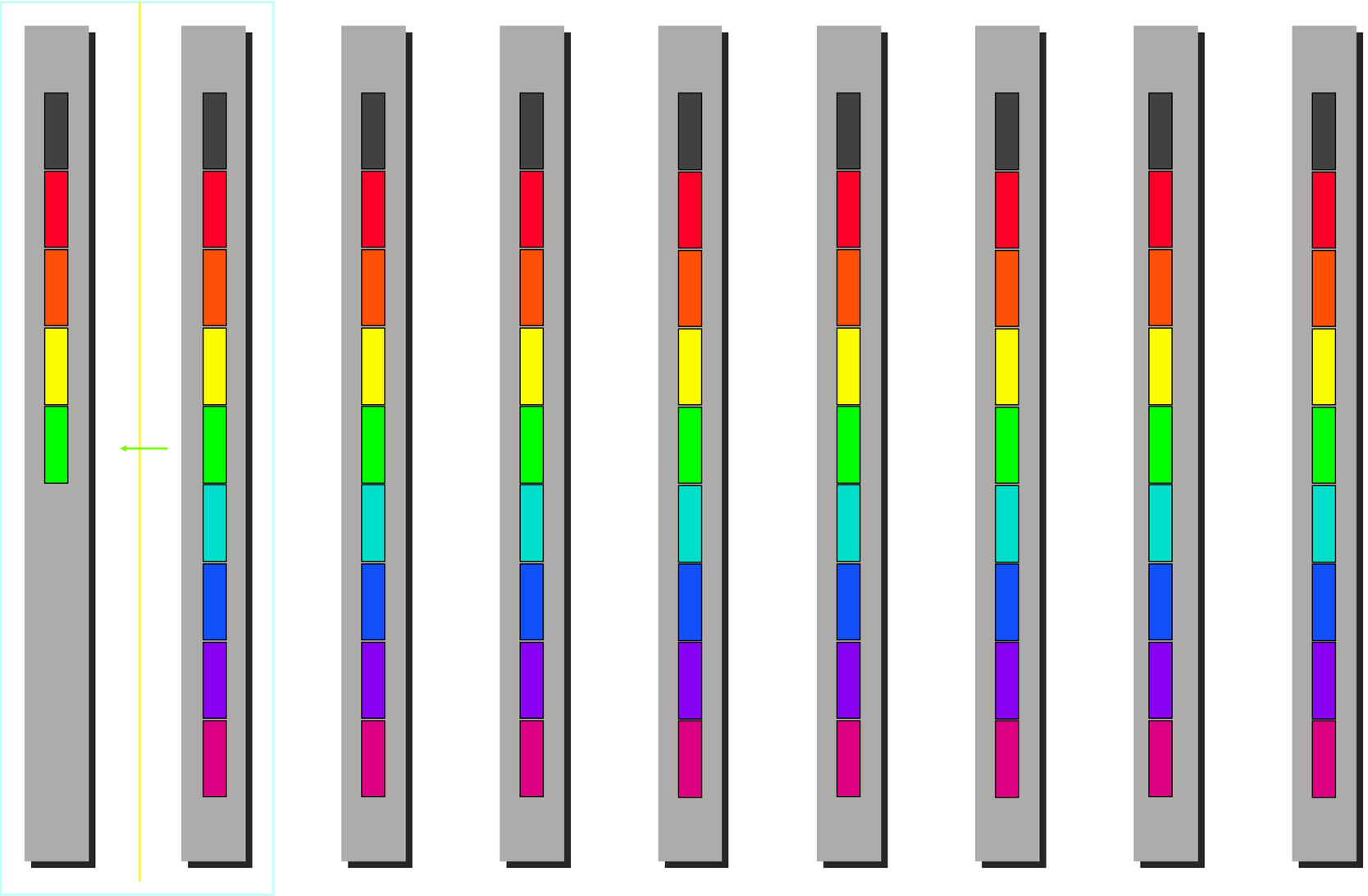


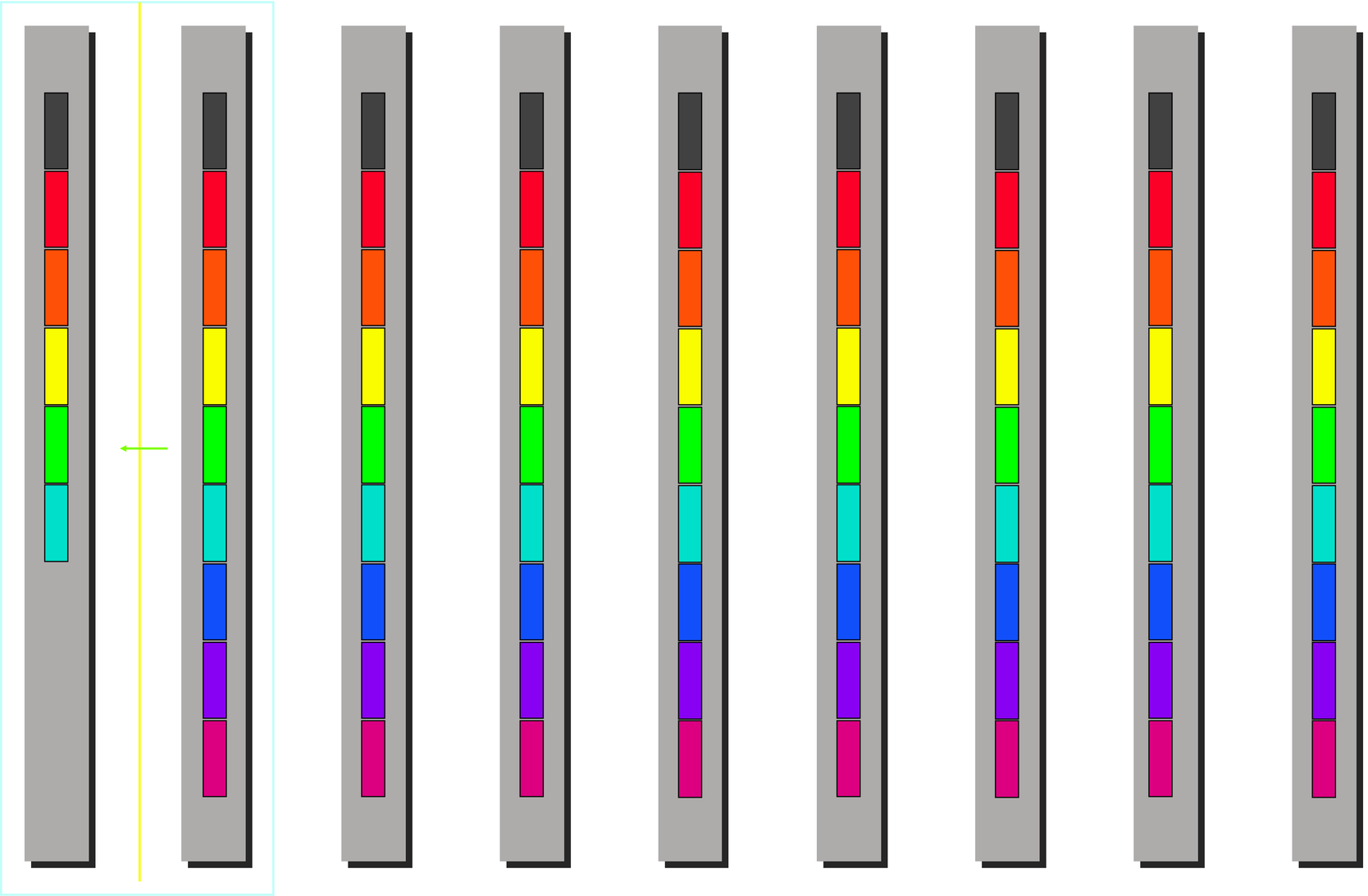


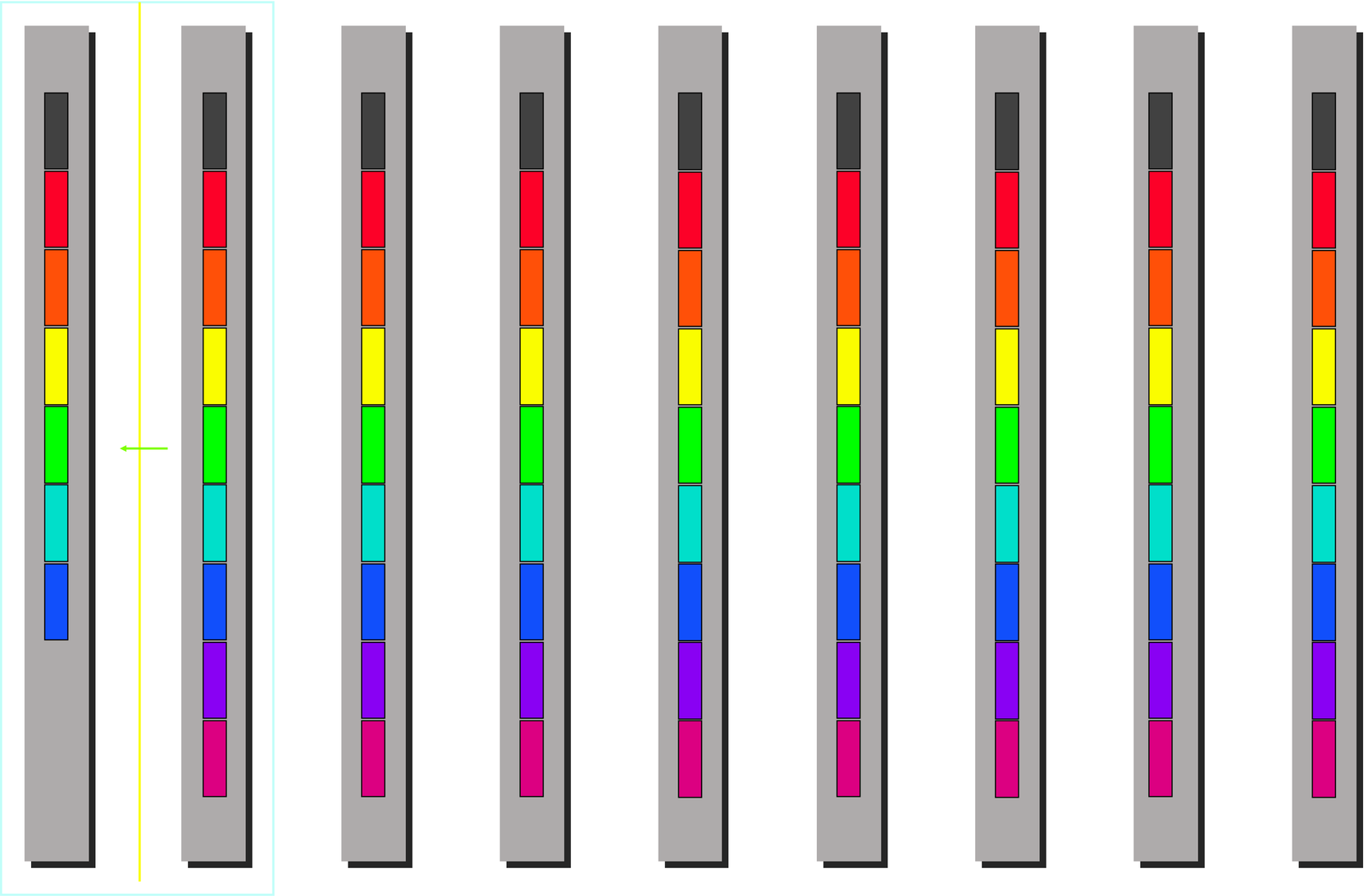


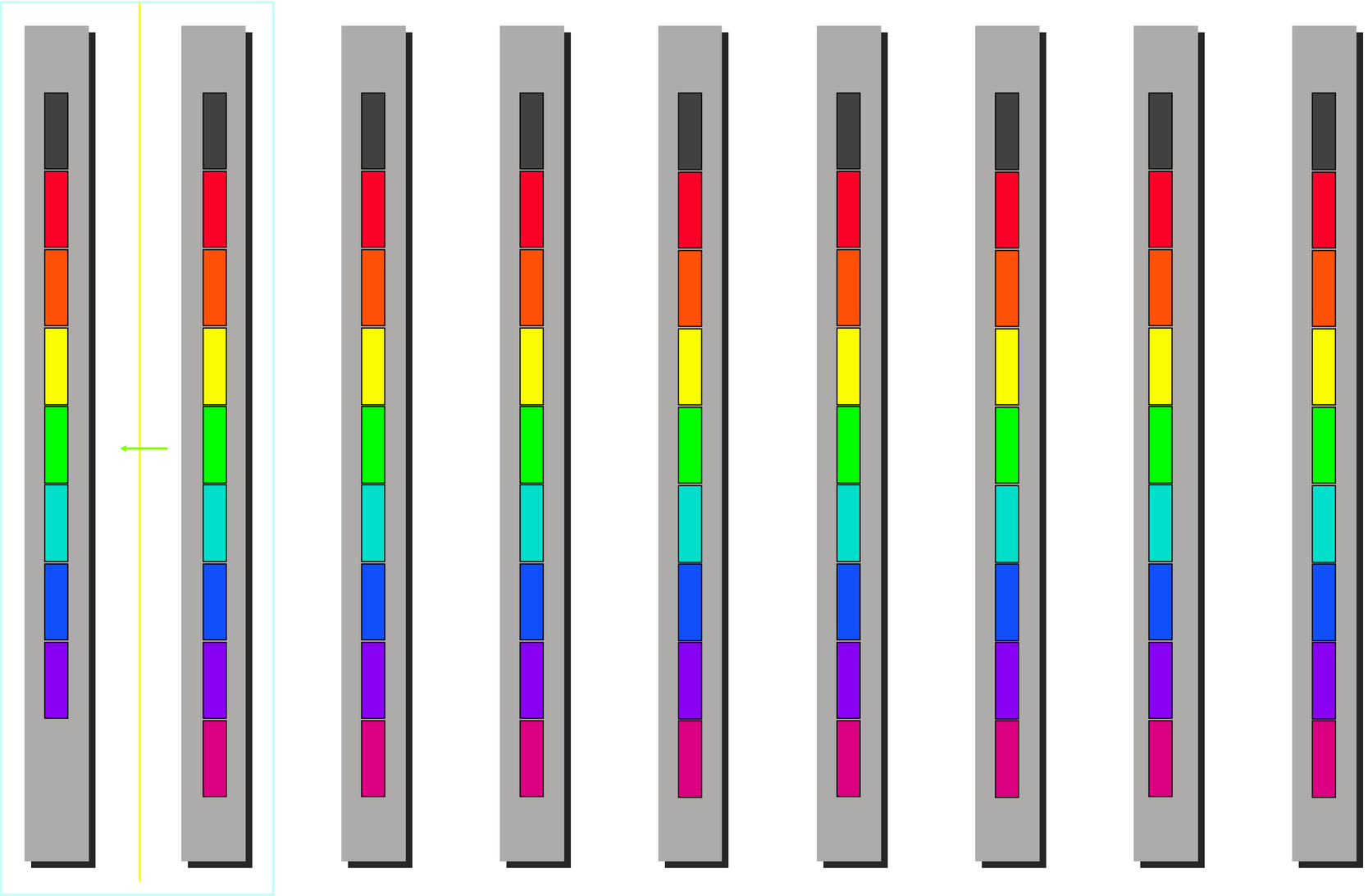


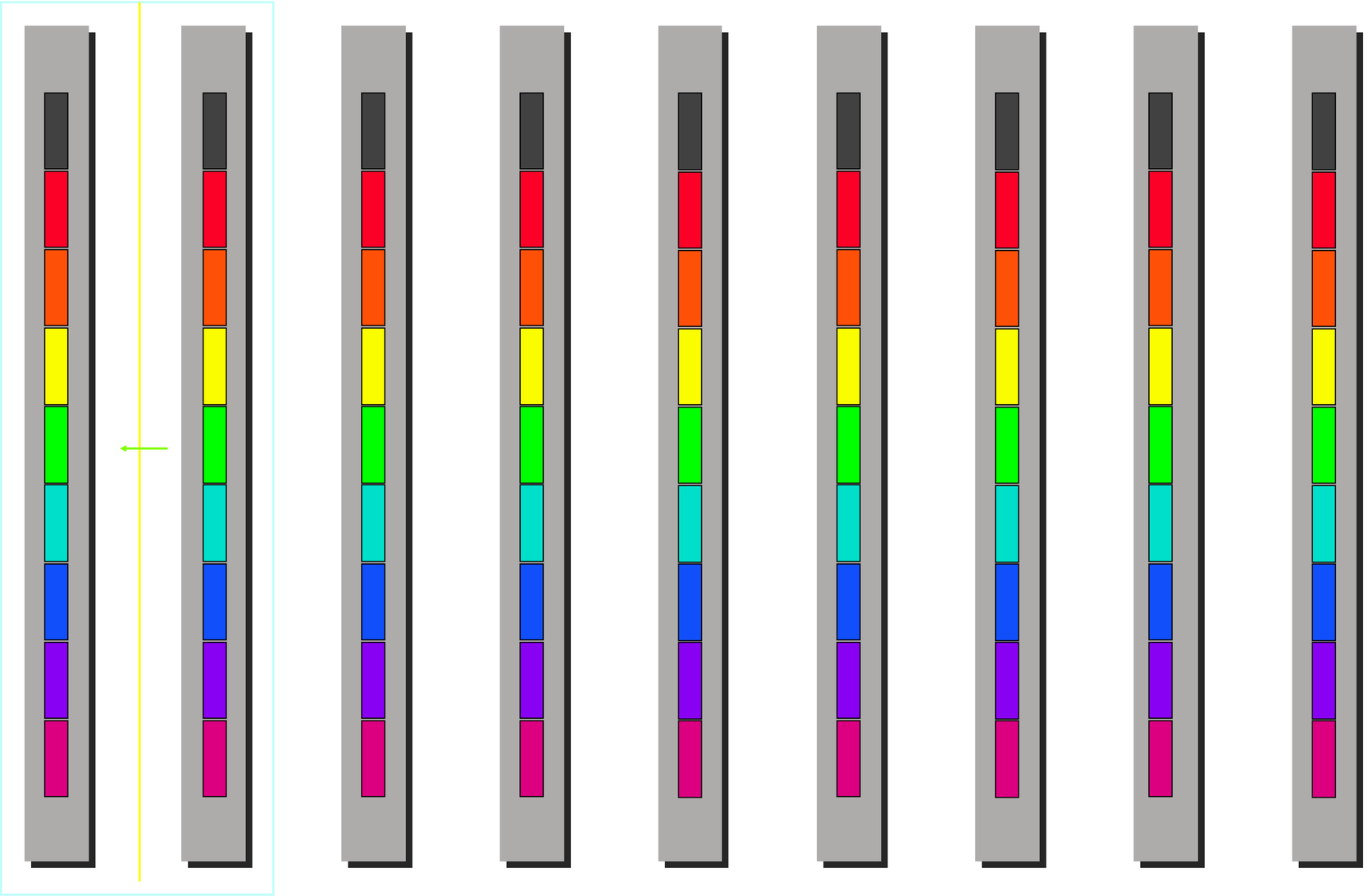


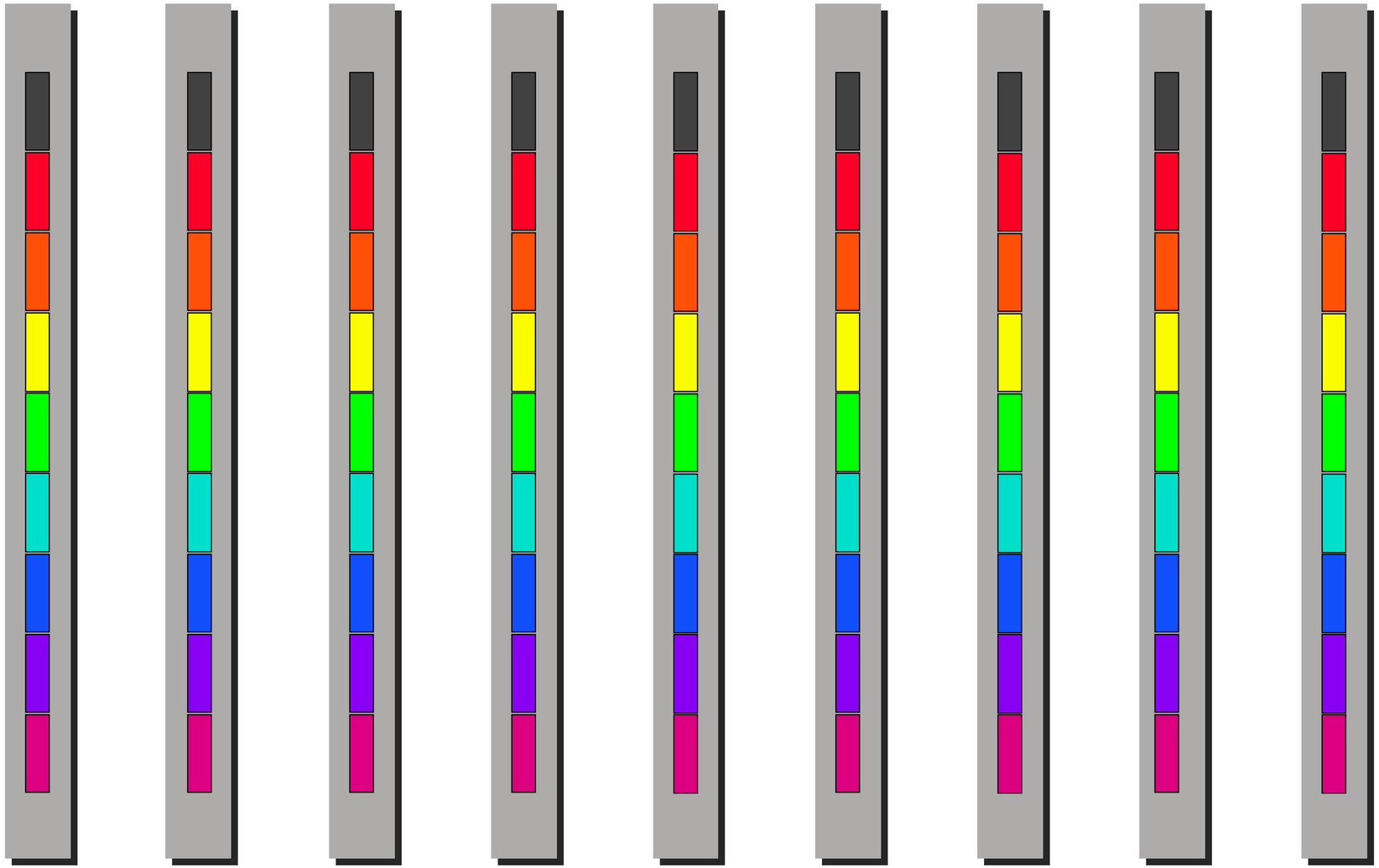












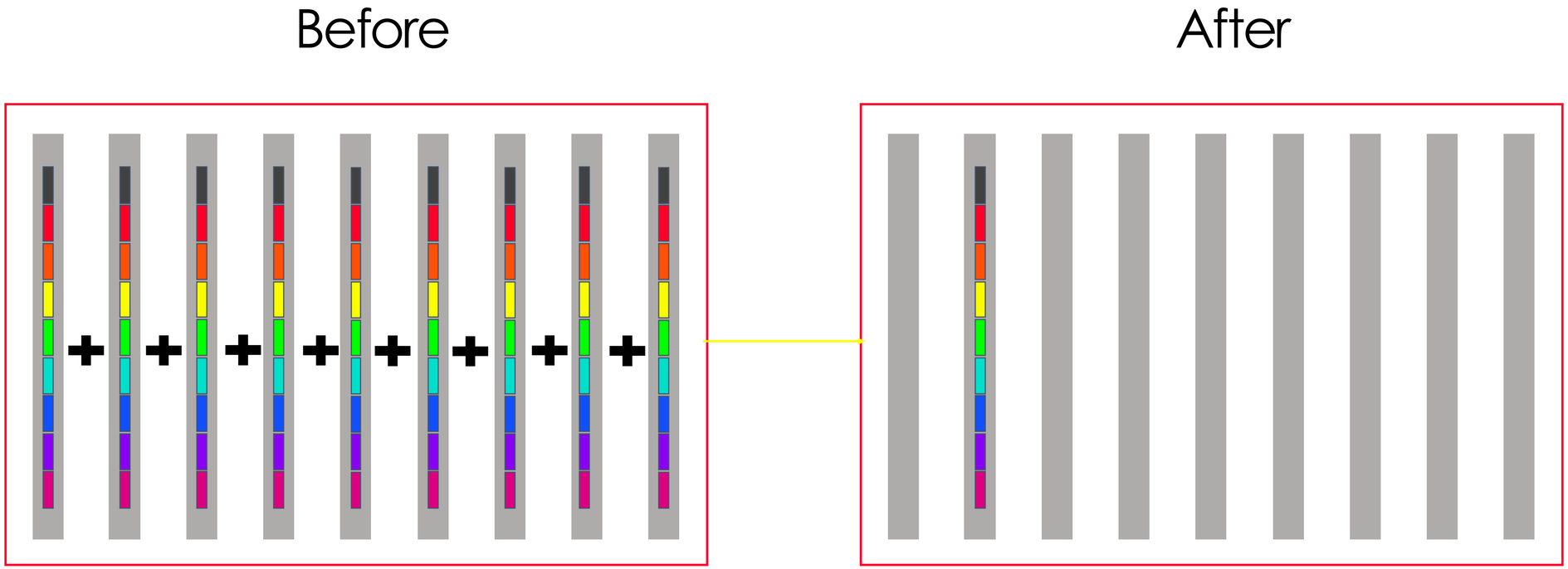
Cost of minimum spanning tree broadcast

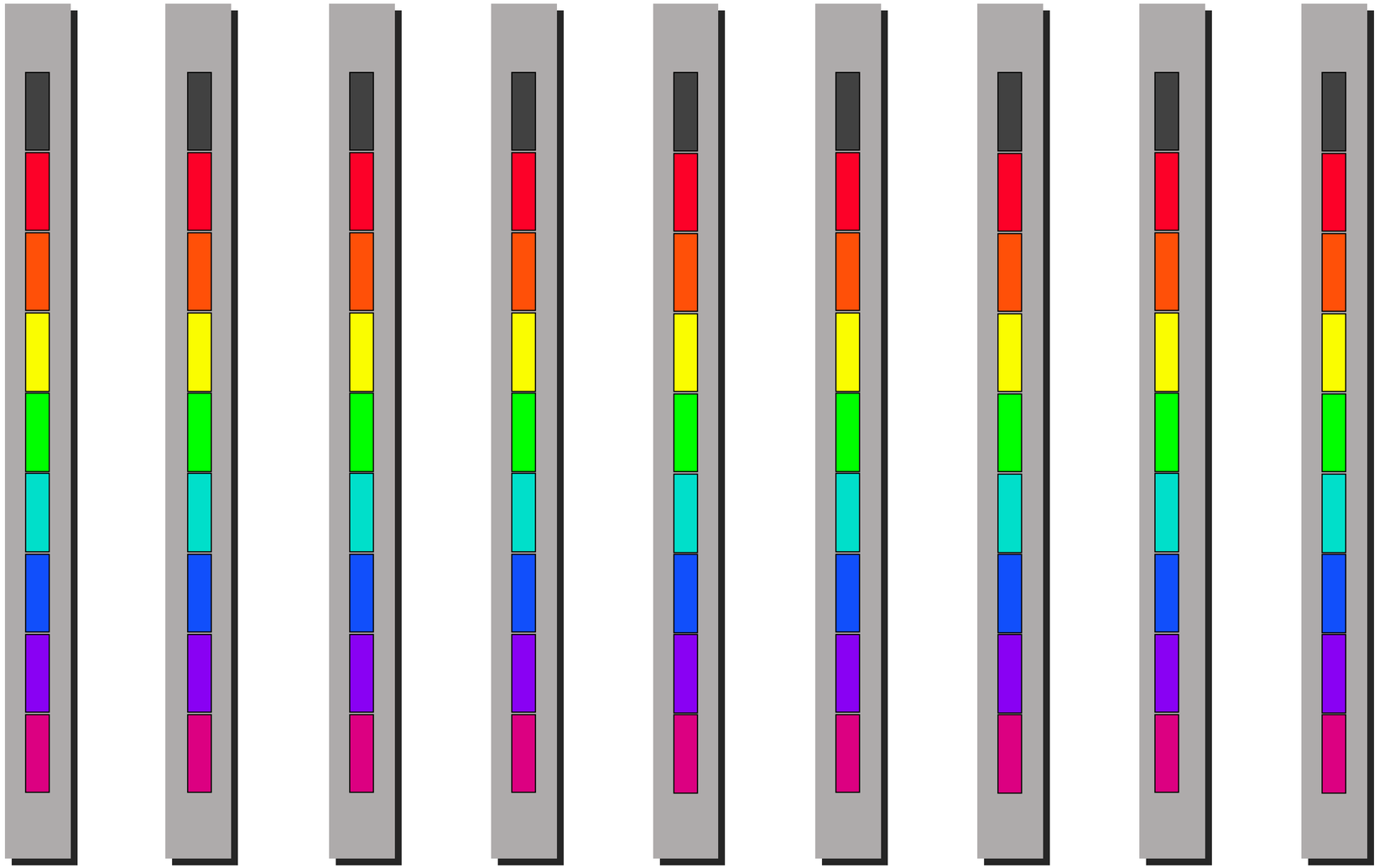
$$\lceil log(p) \rceil \, (\alpha + n\beta)$$

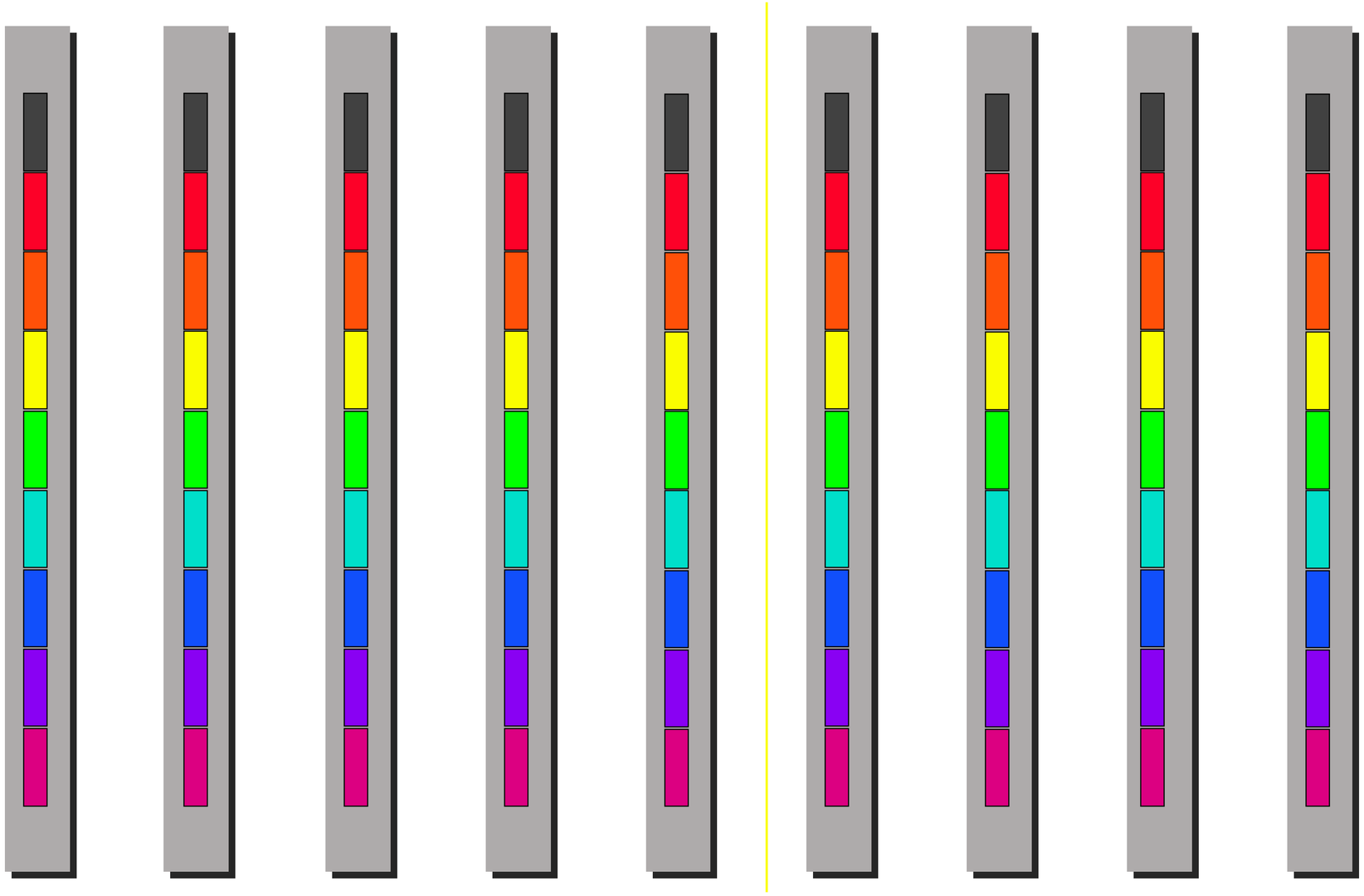
number of steps

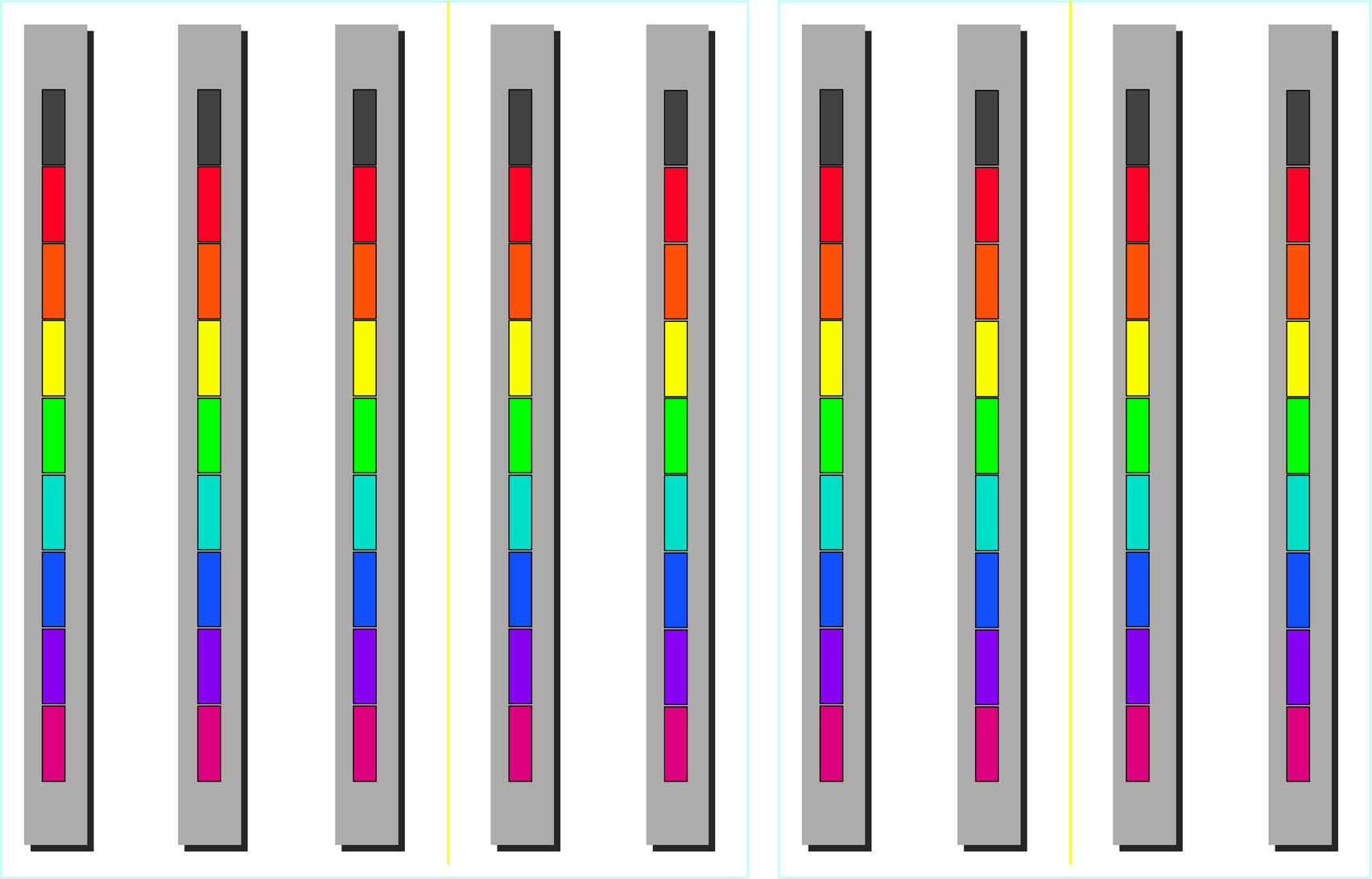
cost per steps

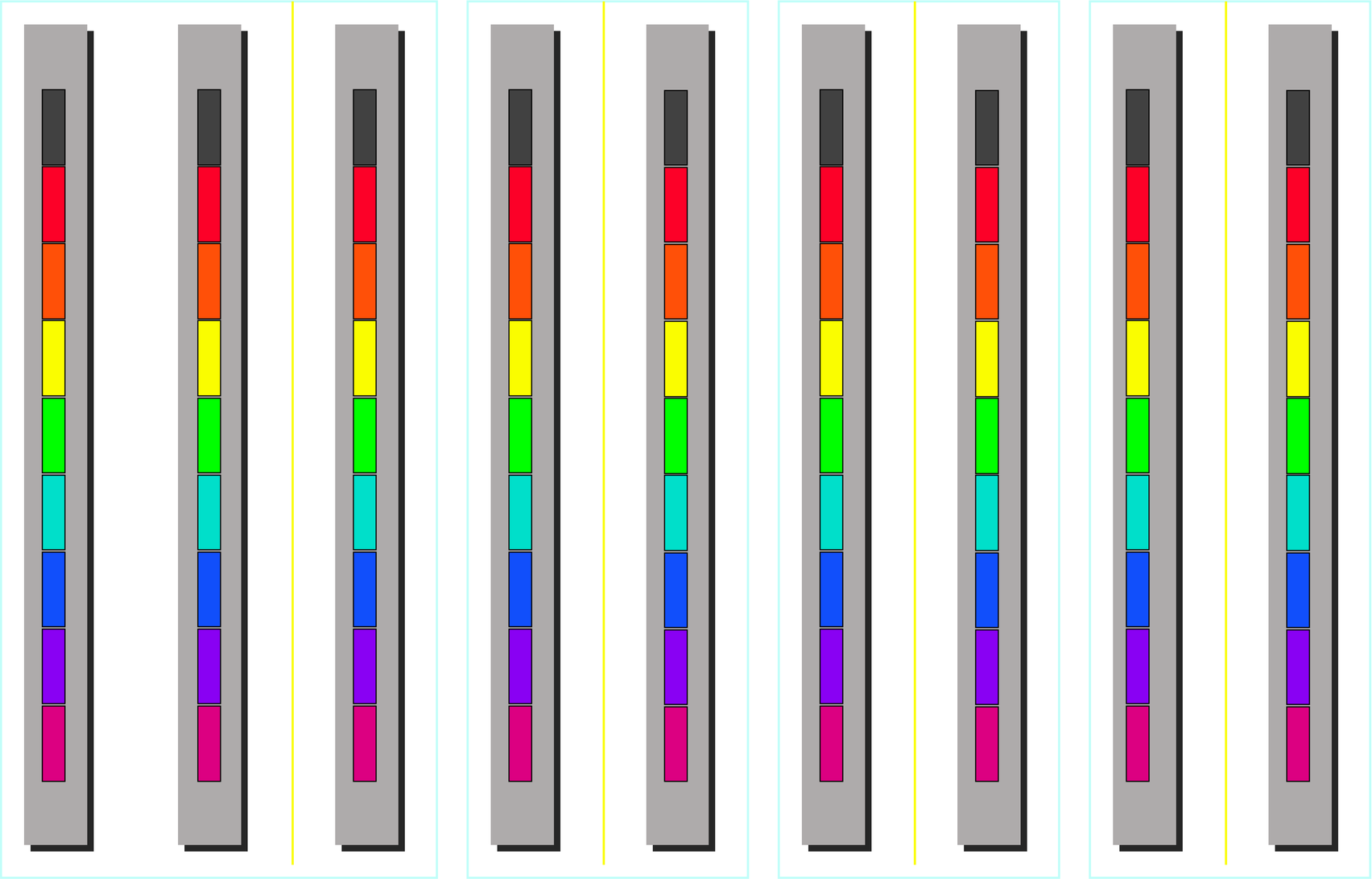
Reduce(-to-one)

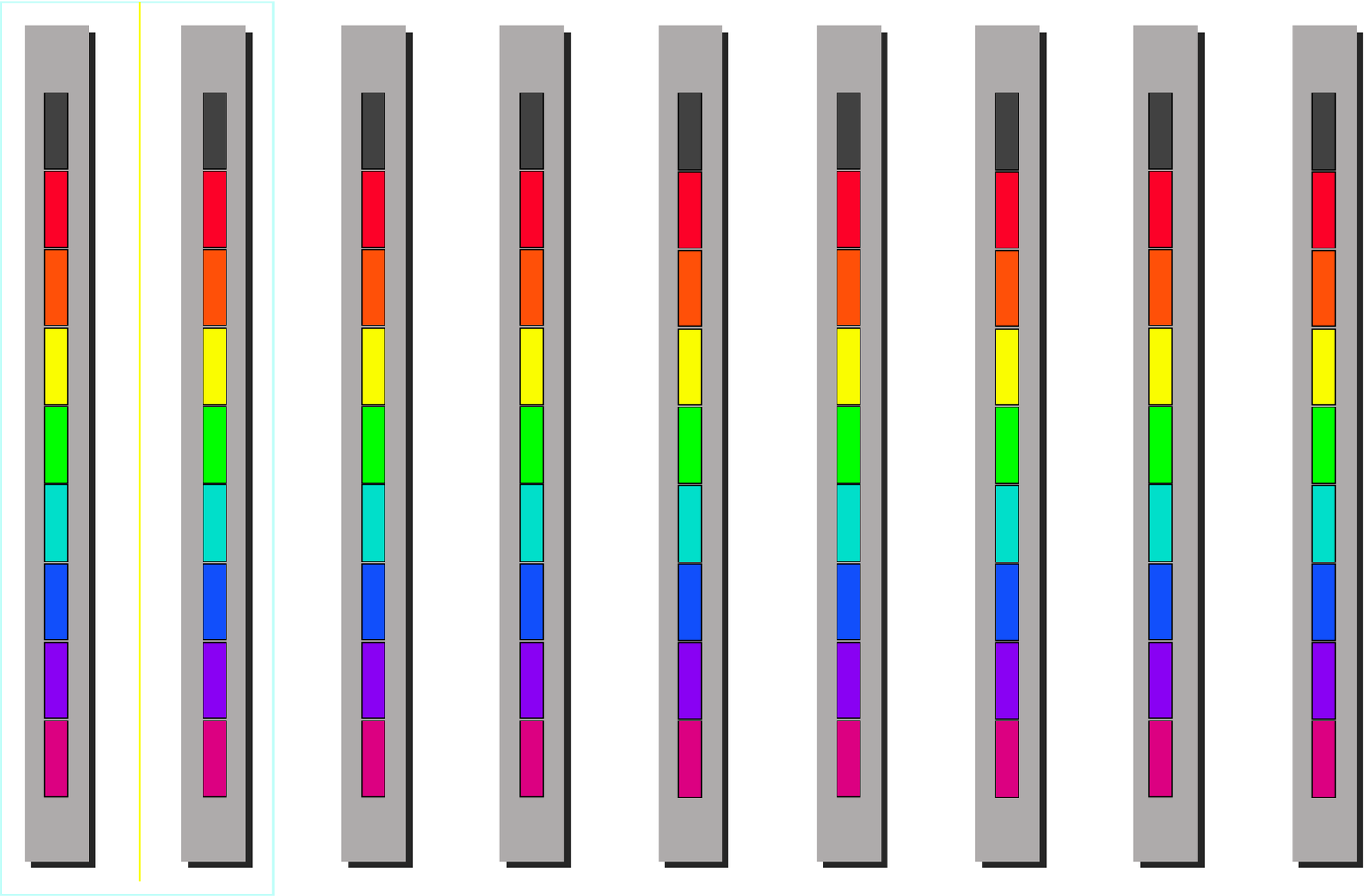


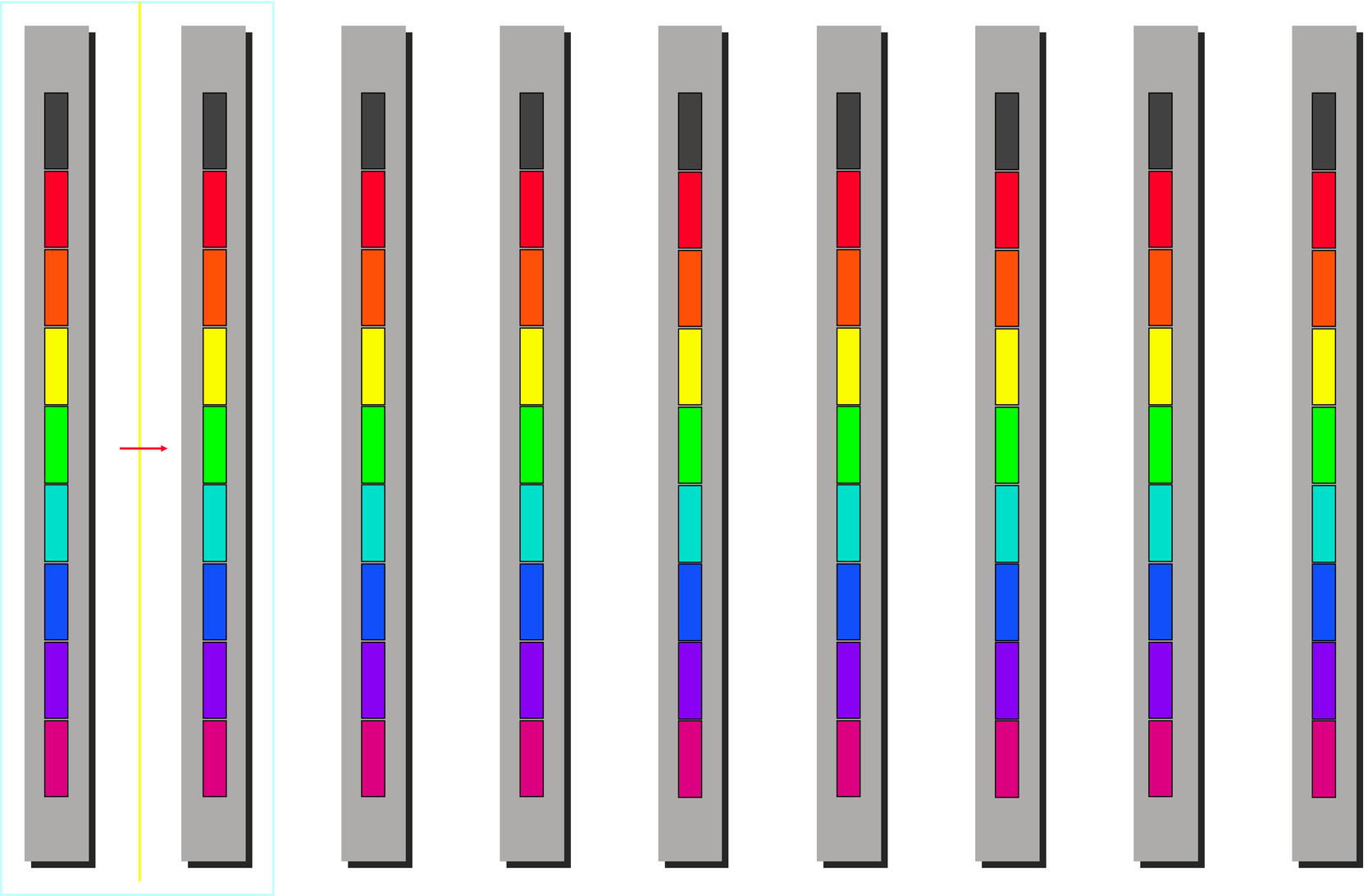


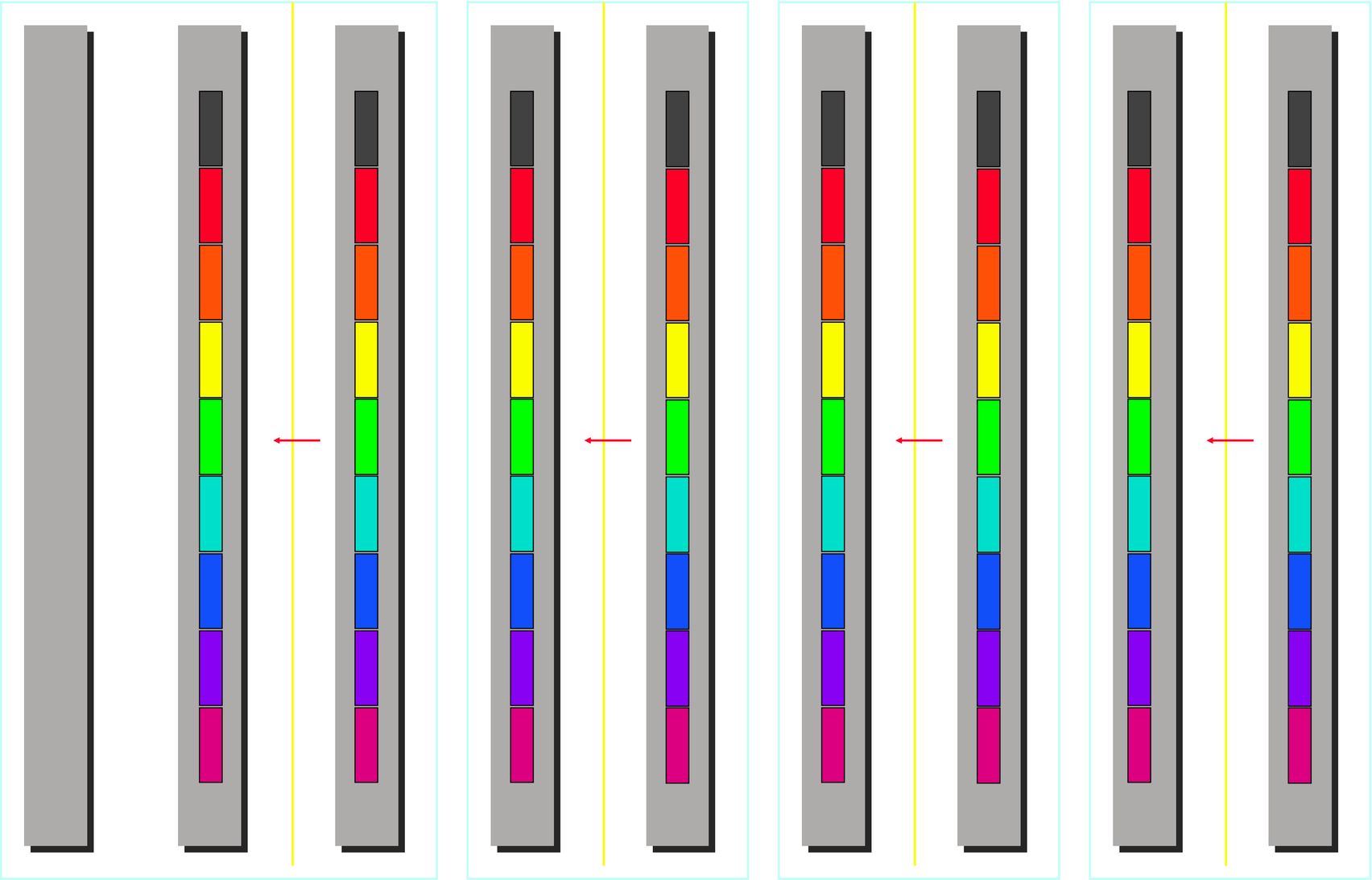


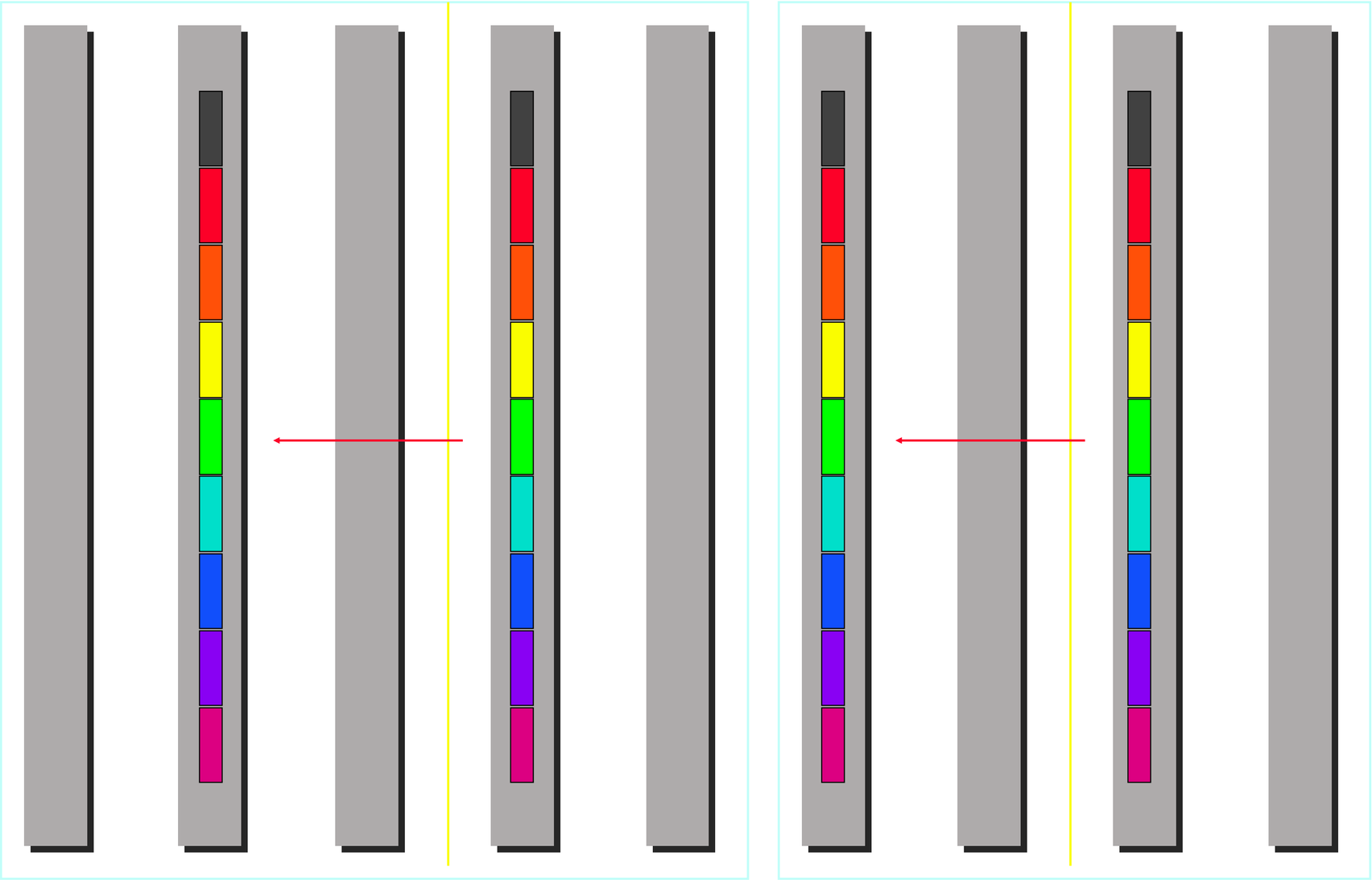


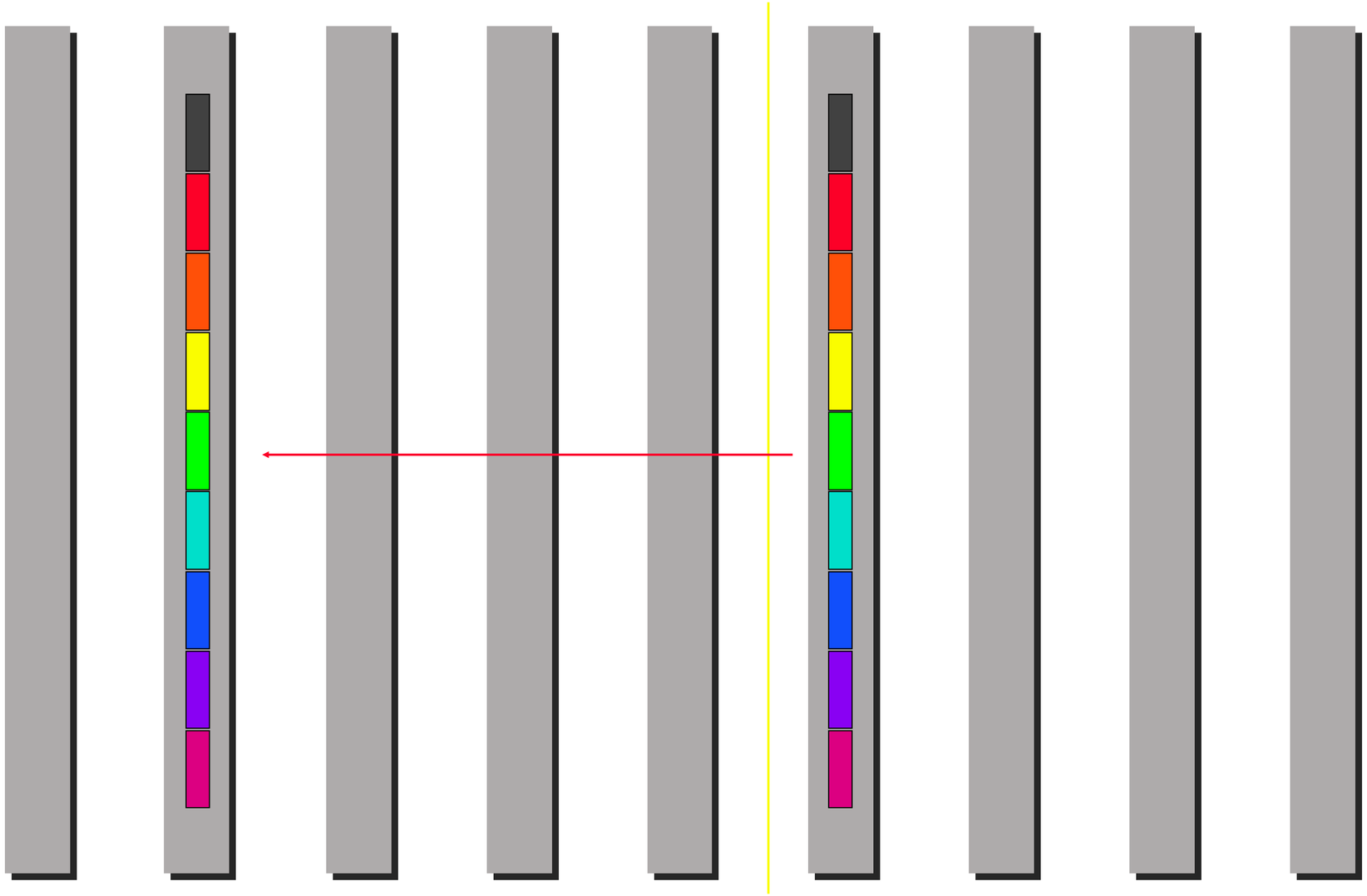








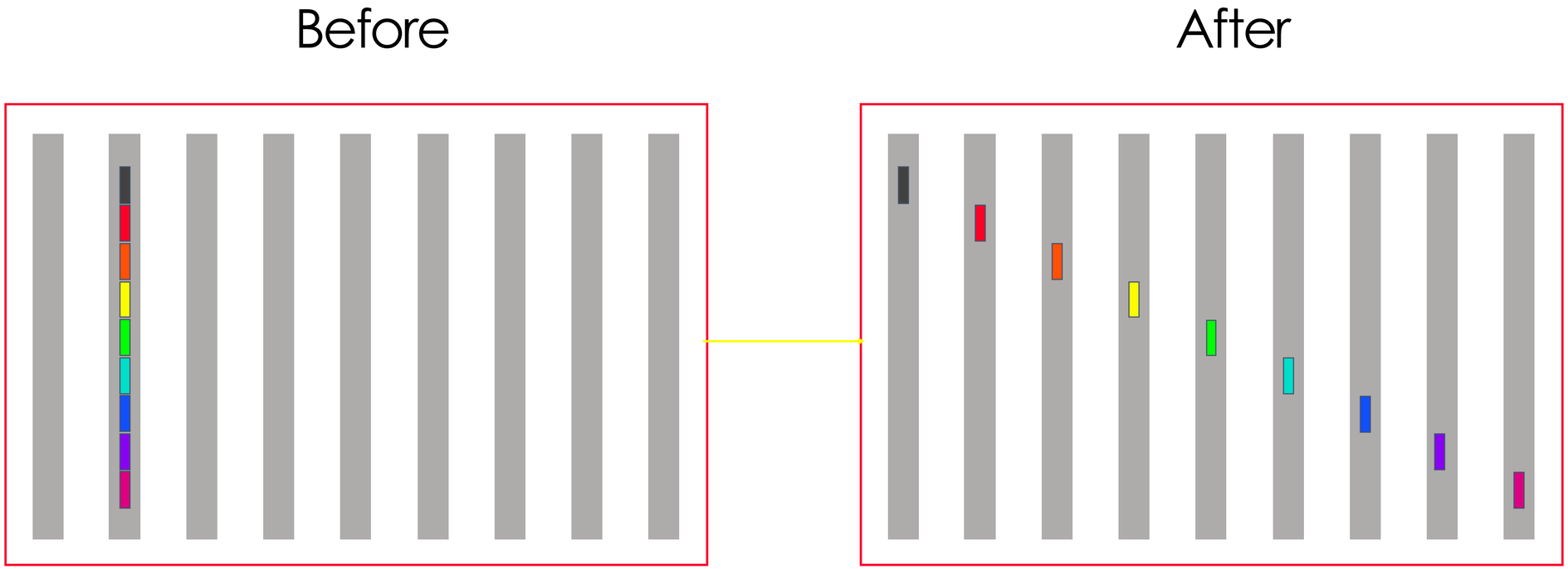


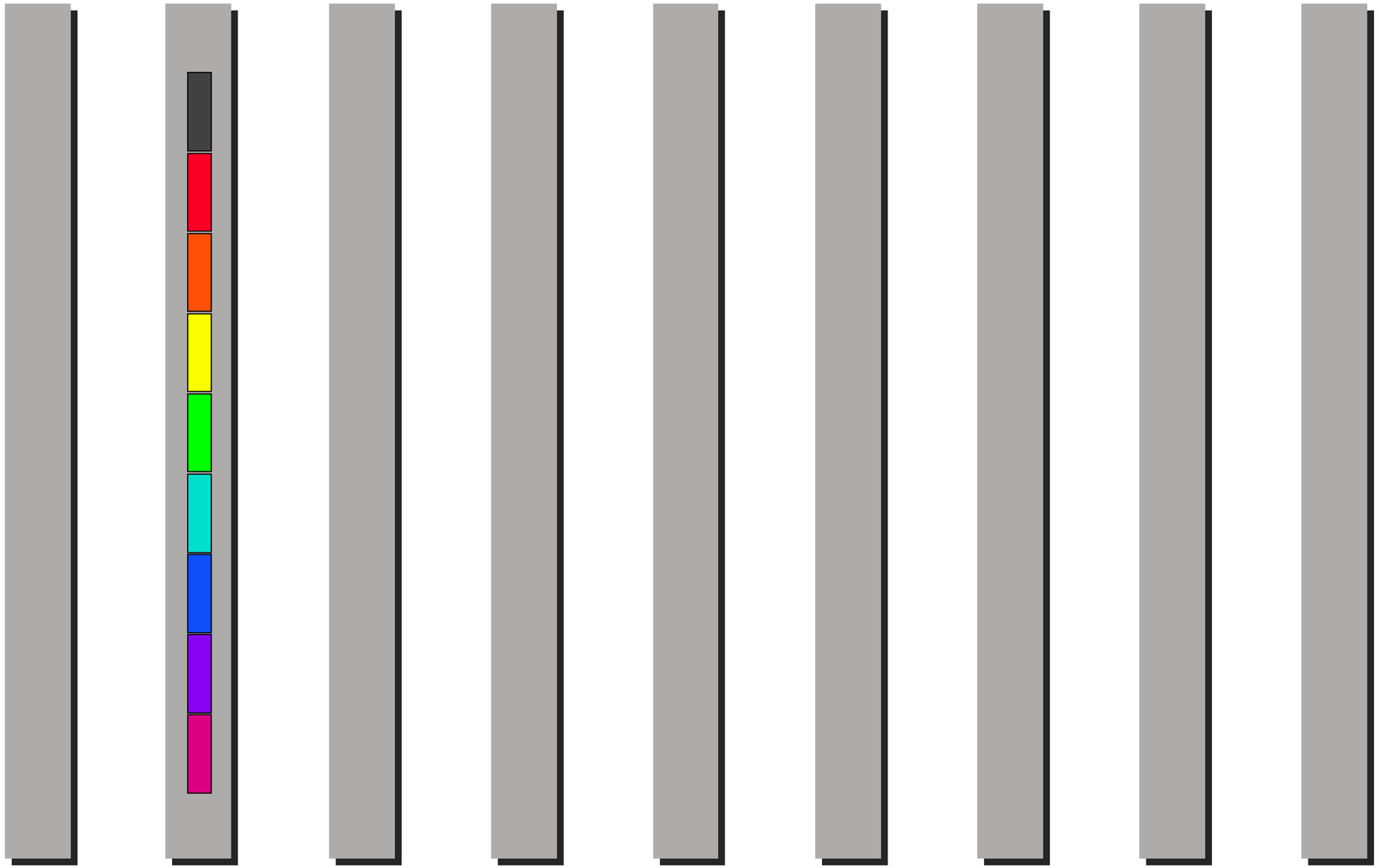


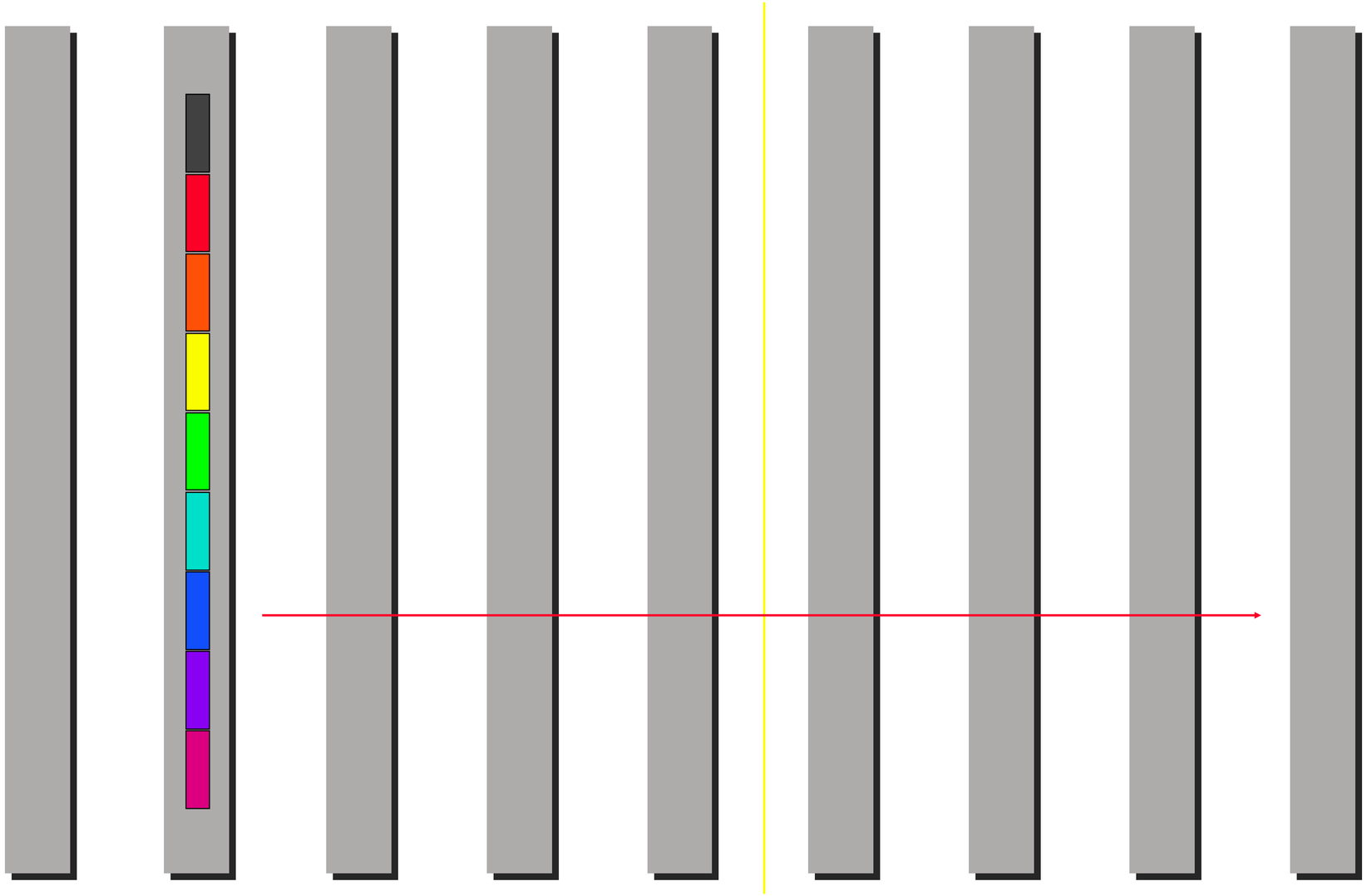
Cost of minimum spanning tree reduce(-to-one)

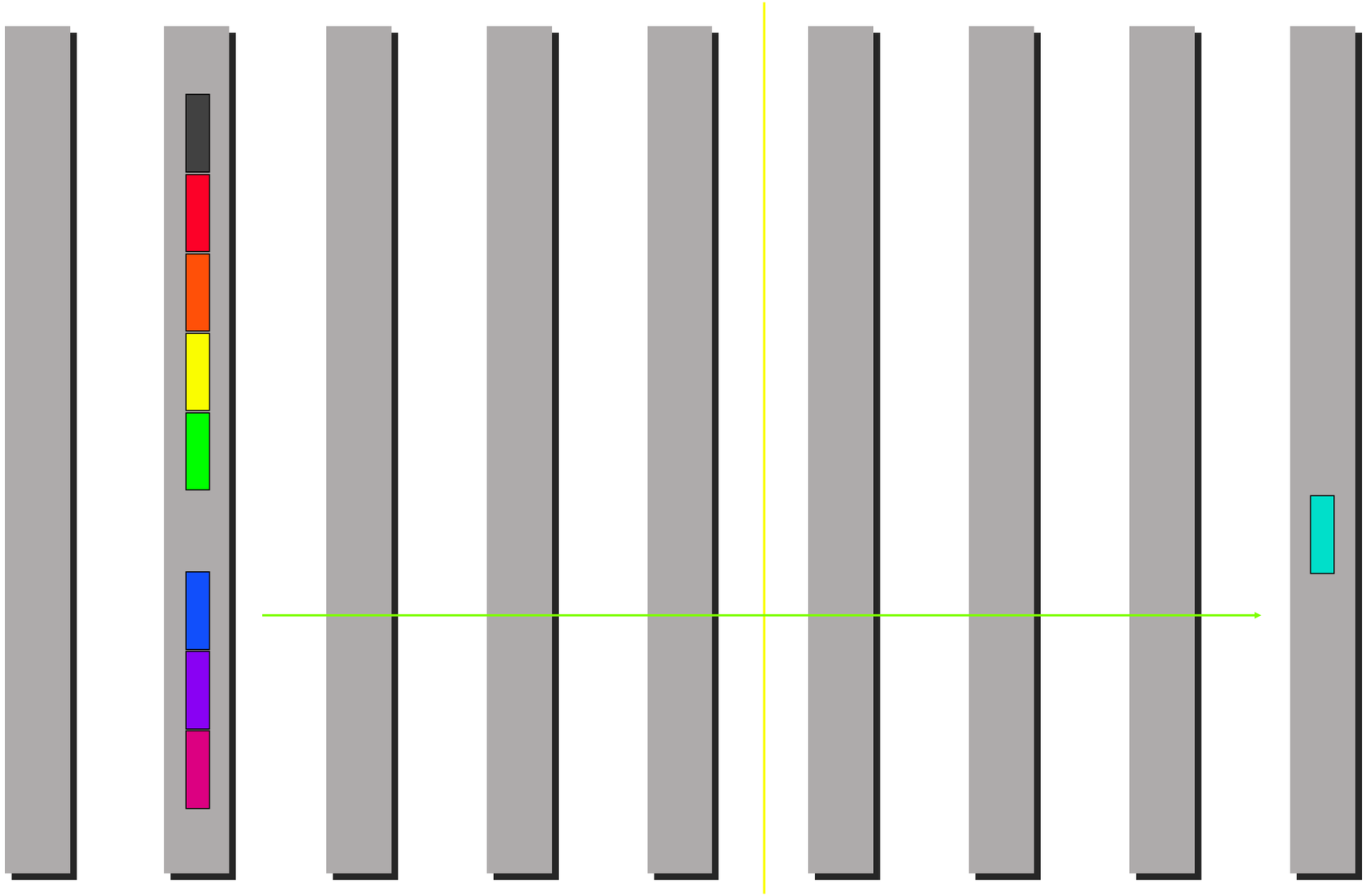
$$\lceil log(p) \rceil \ (\alpha + n\beta + n\gamma)$$

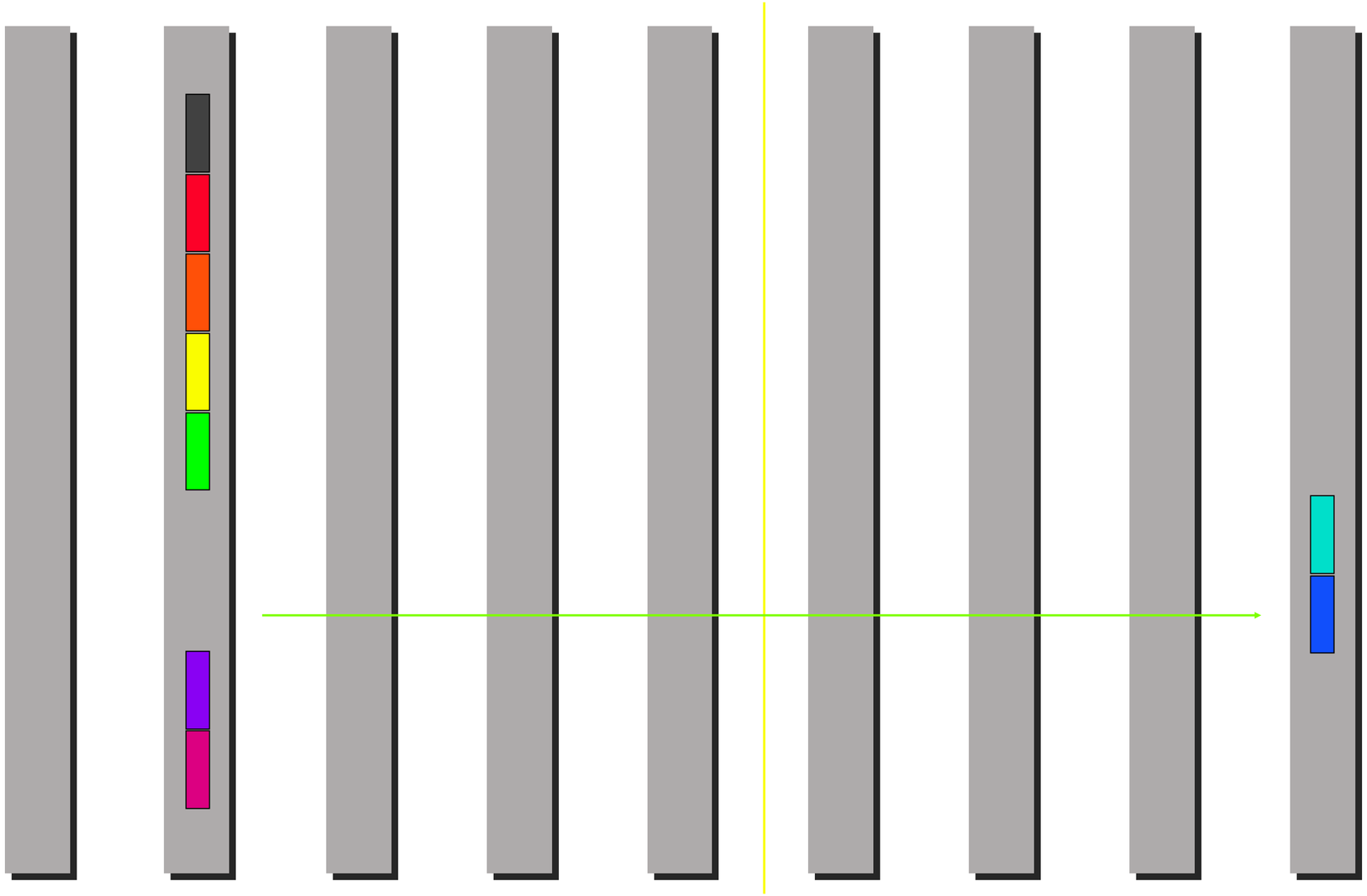
Scatter

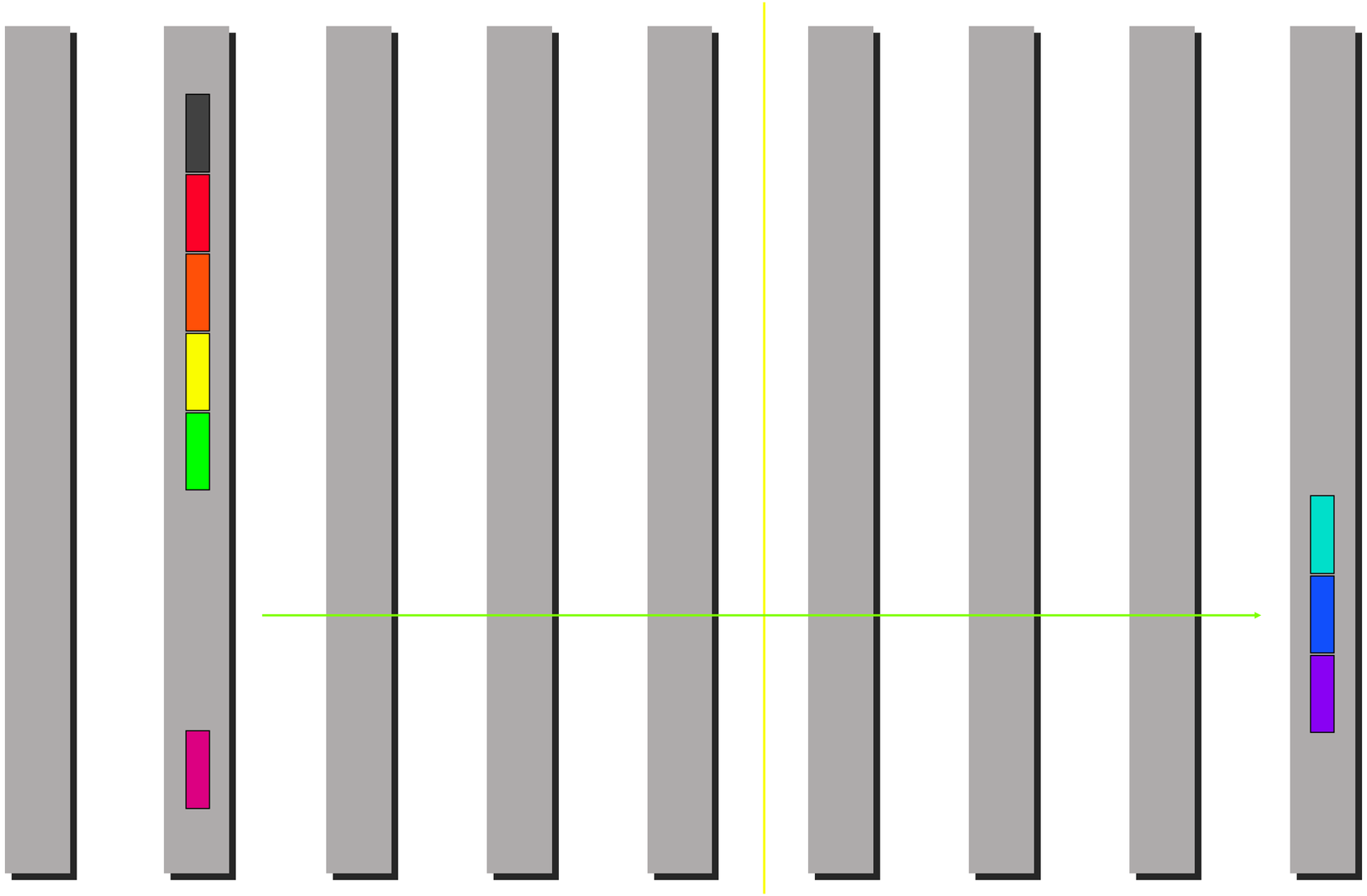


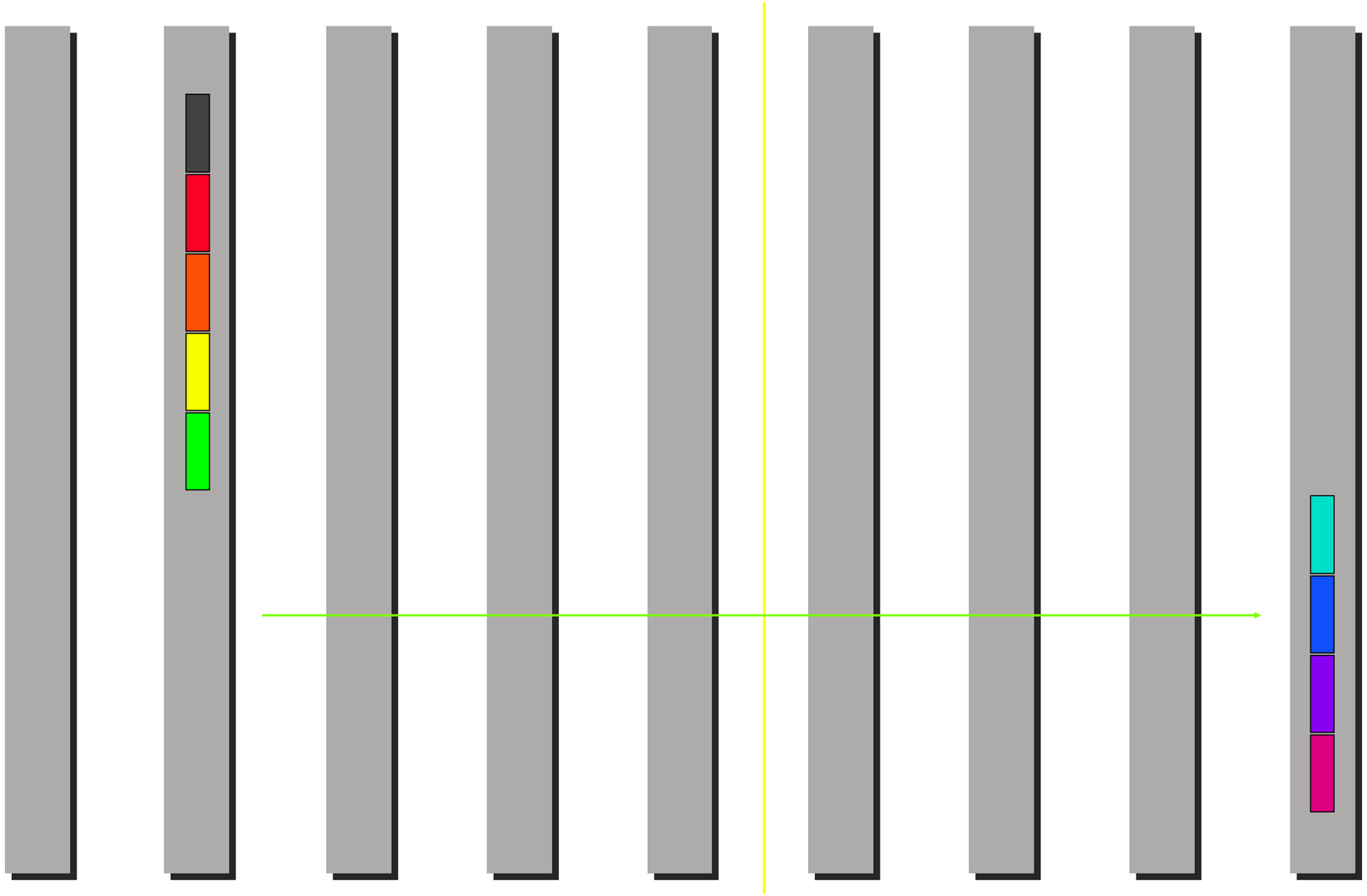


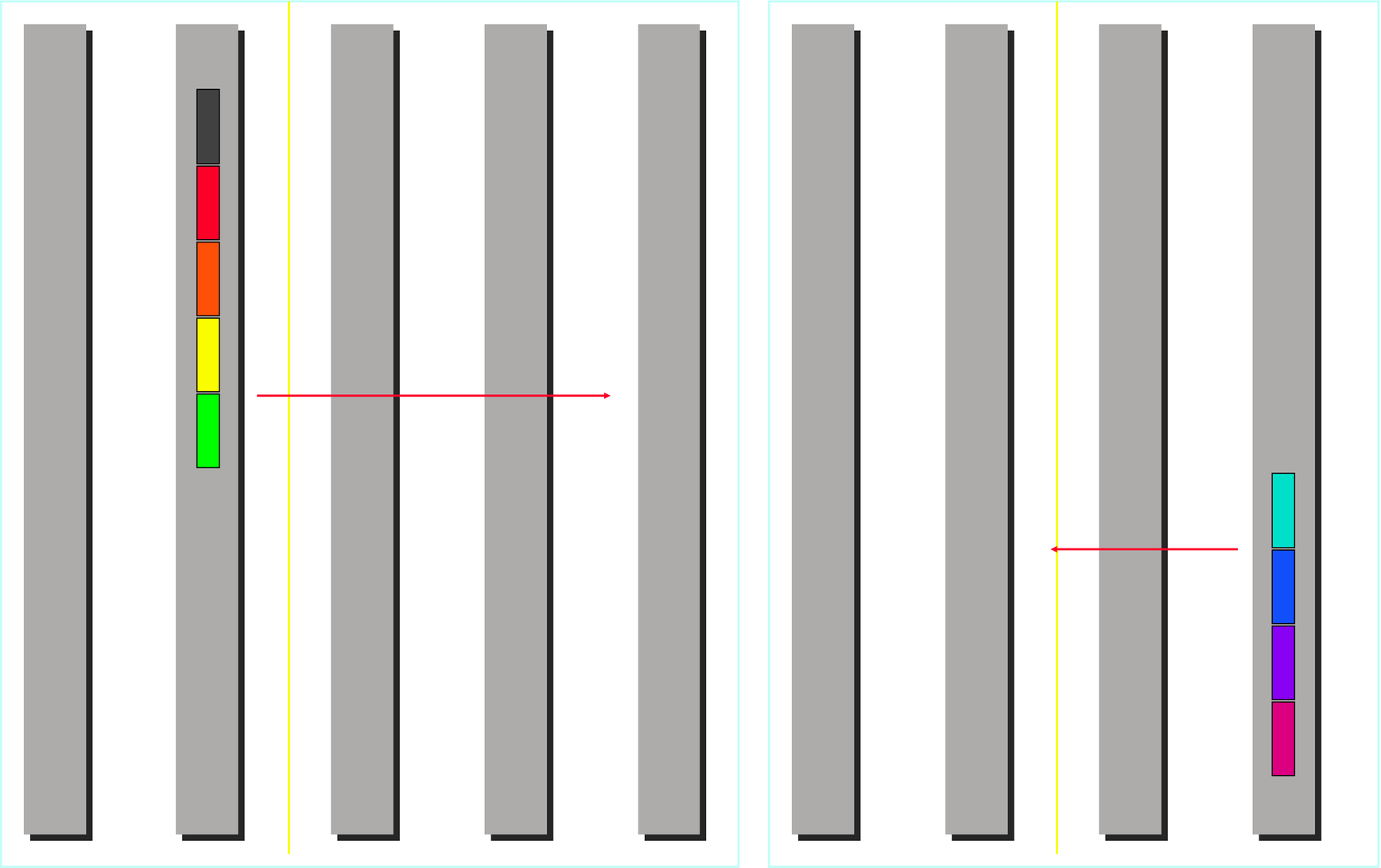


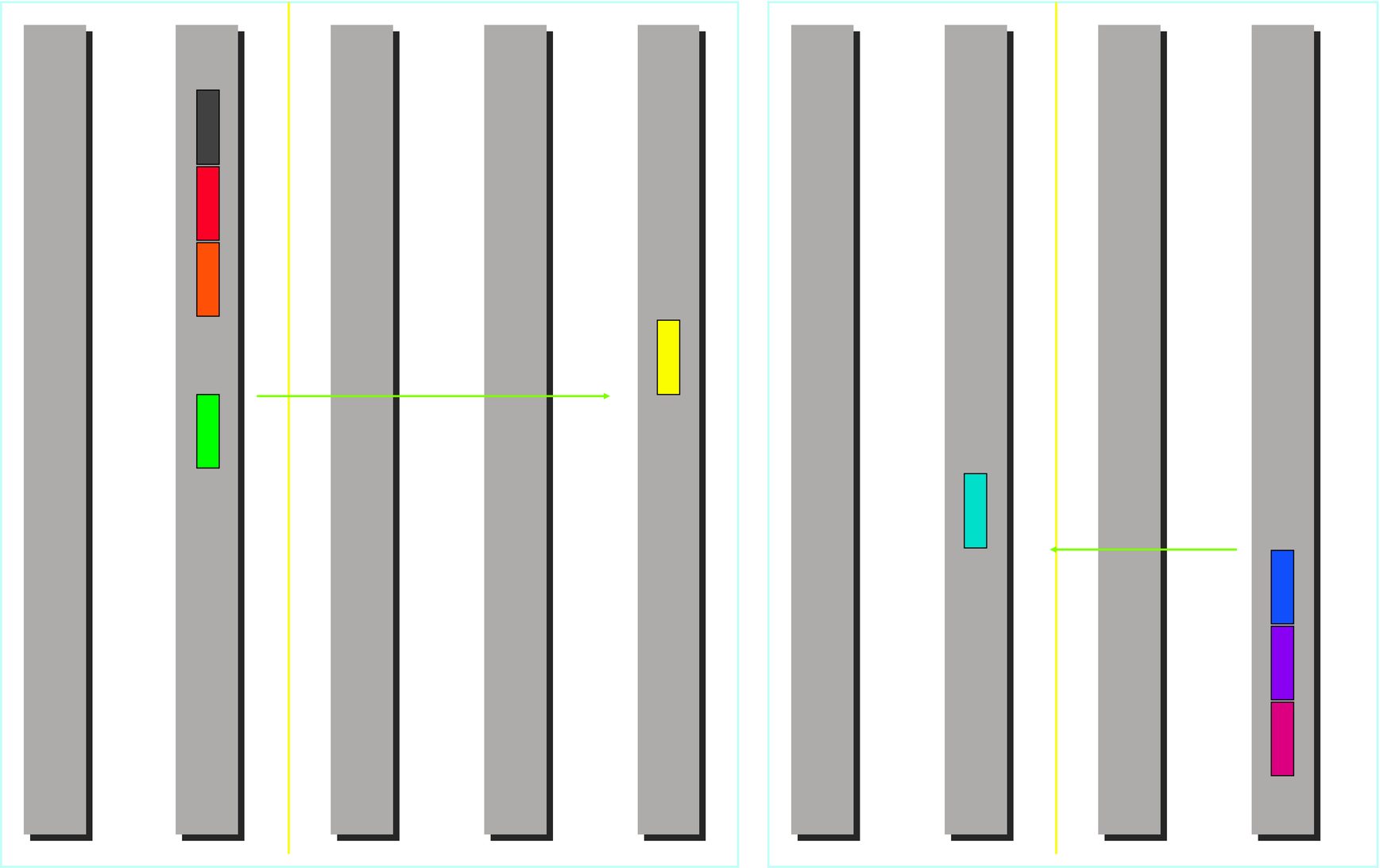


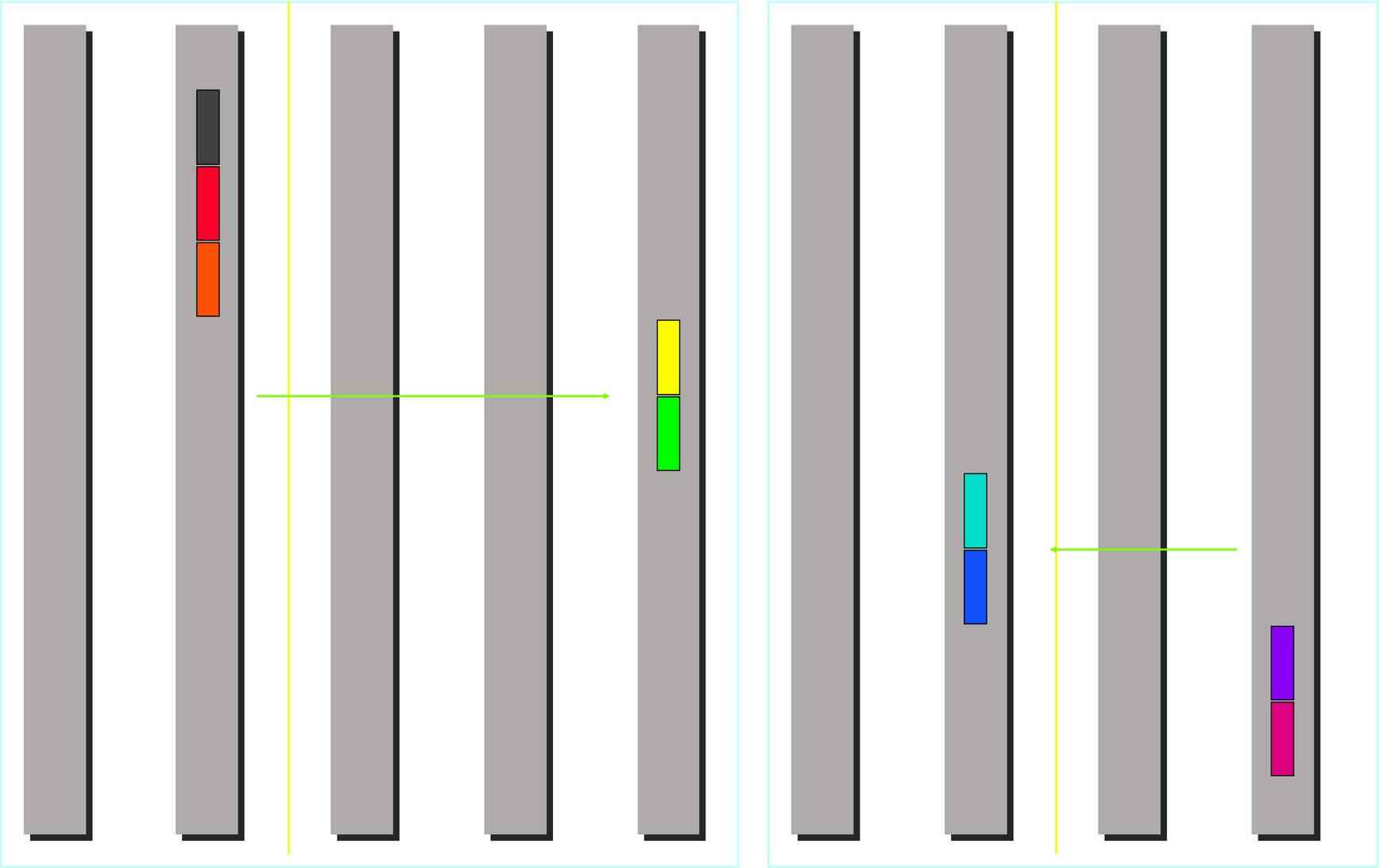


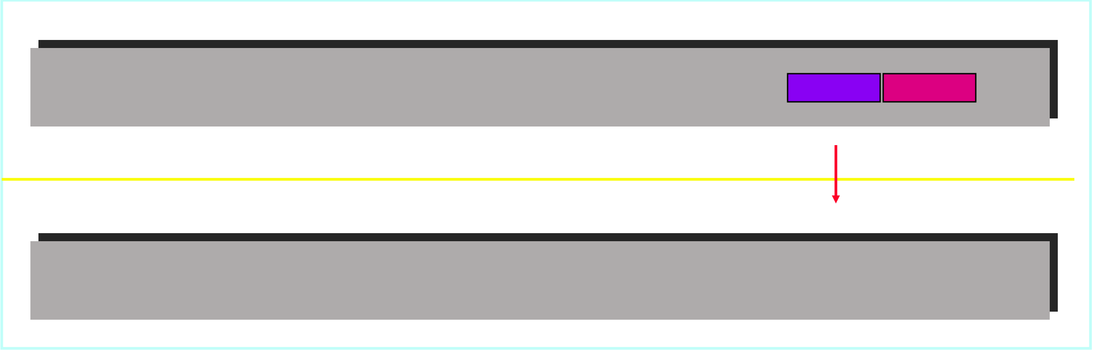


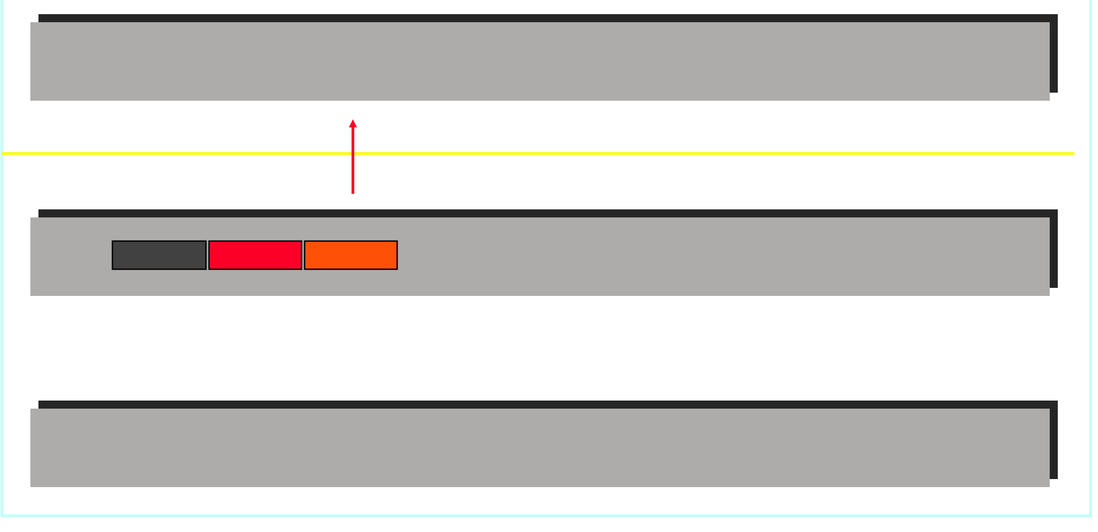


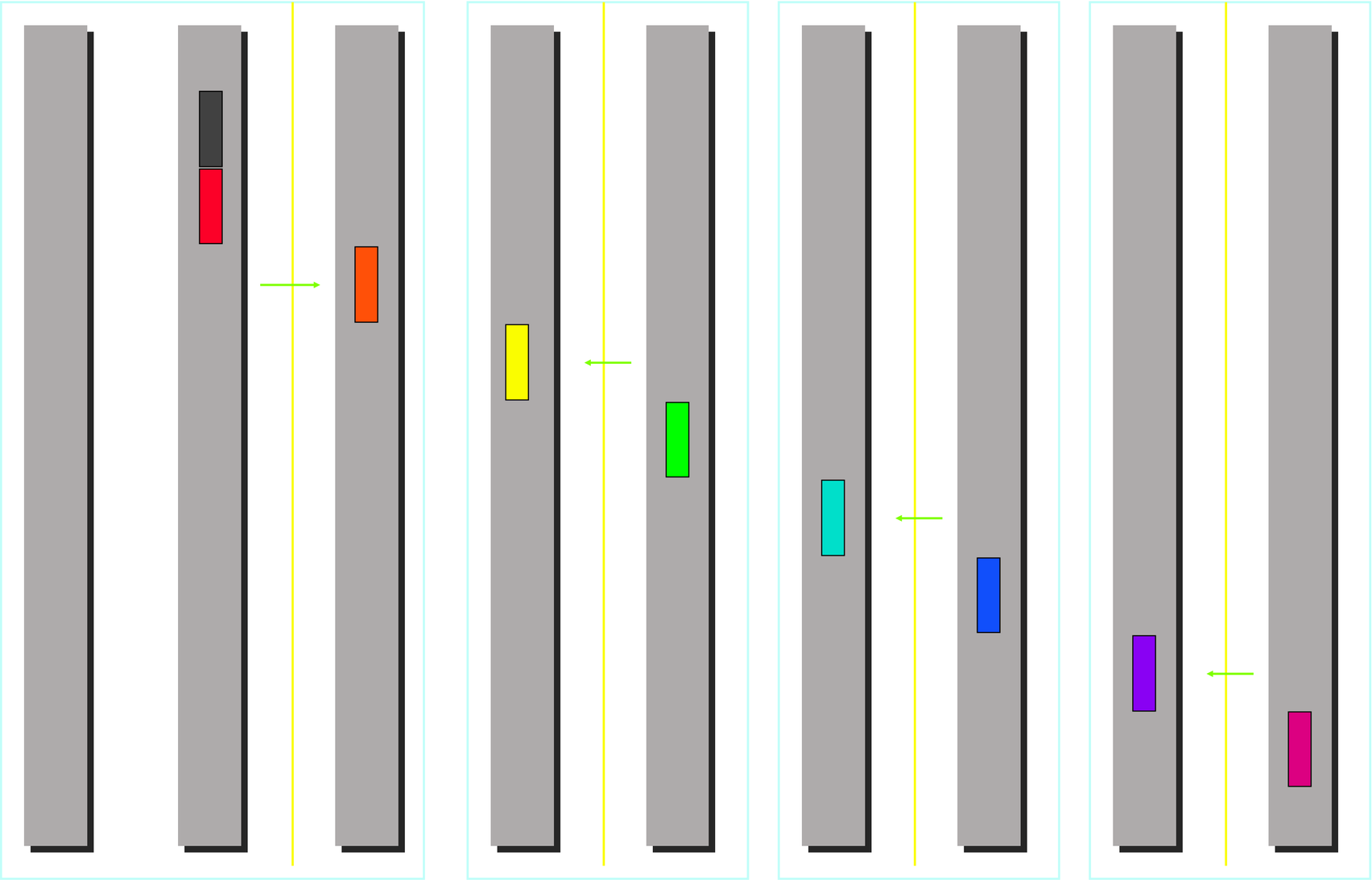


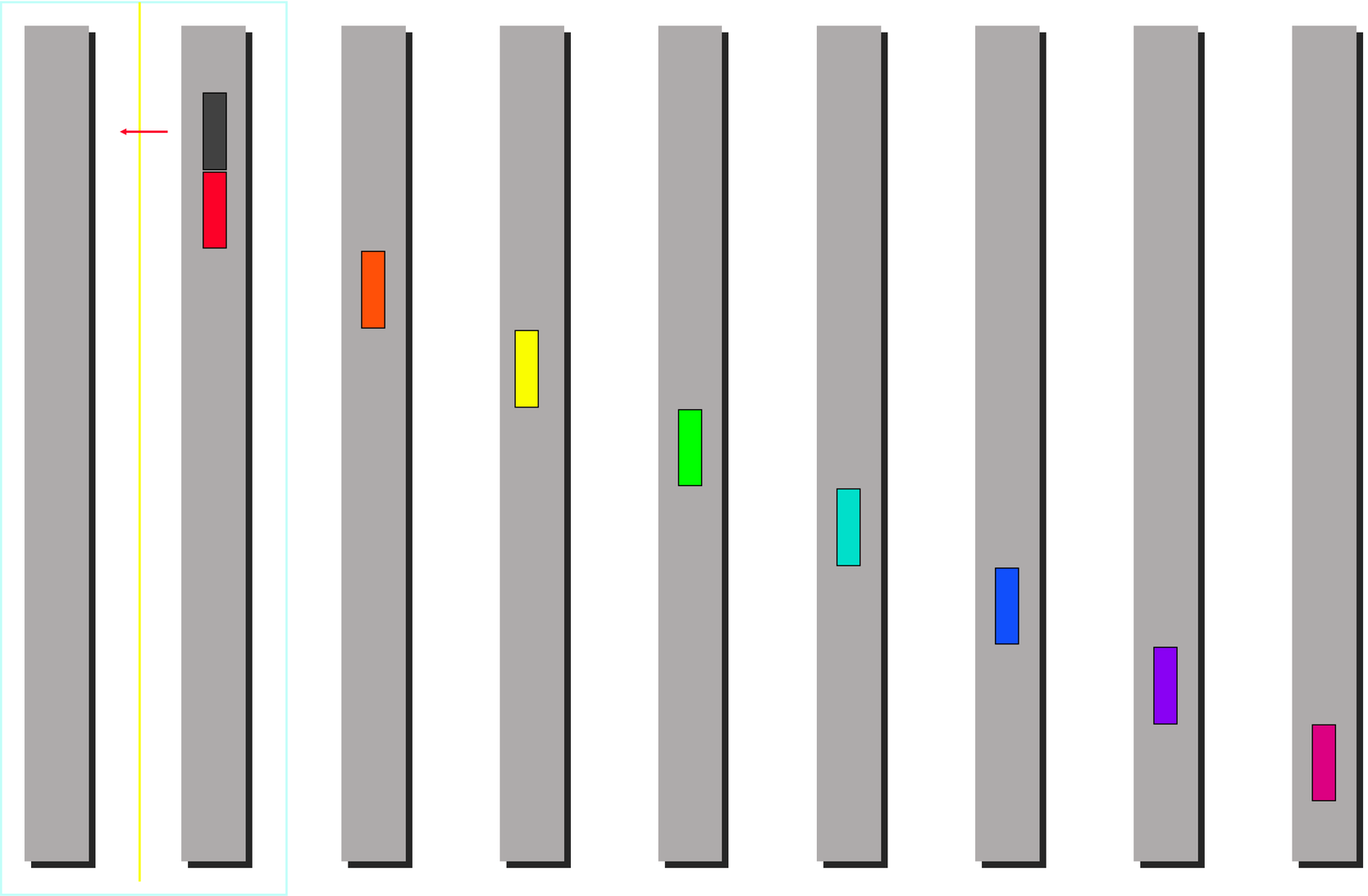


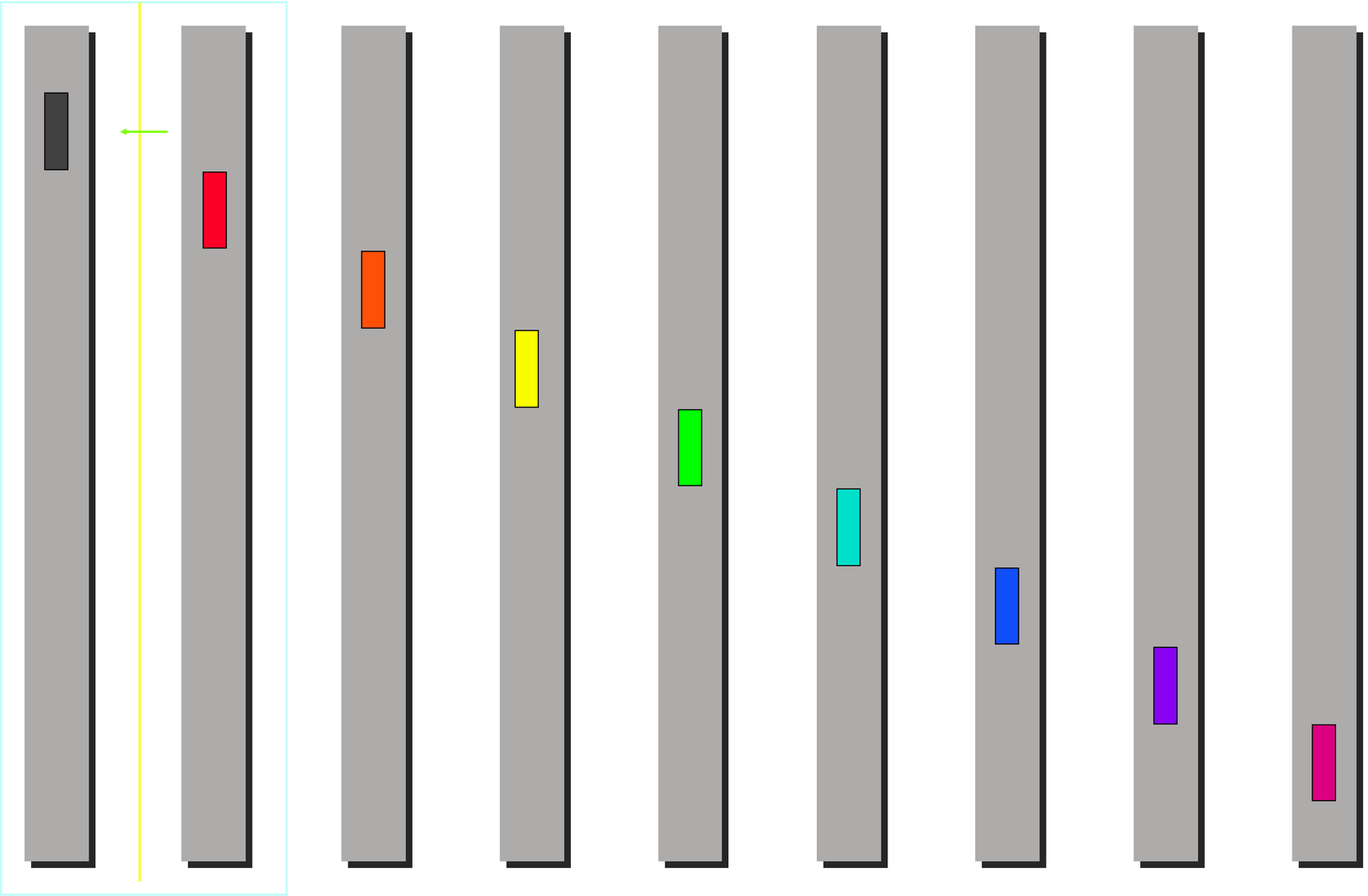


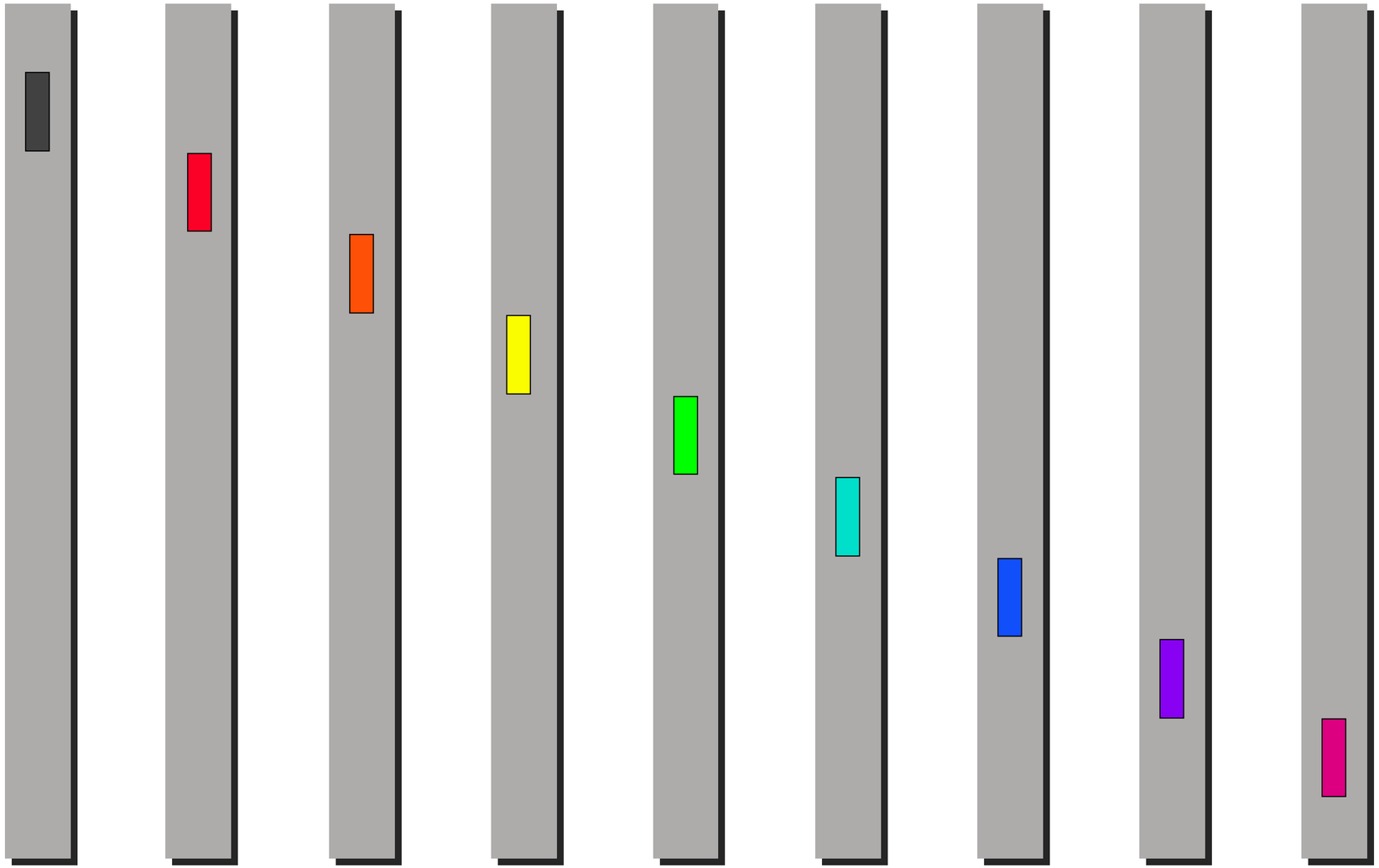












Cost of minimum spanning tree scatter

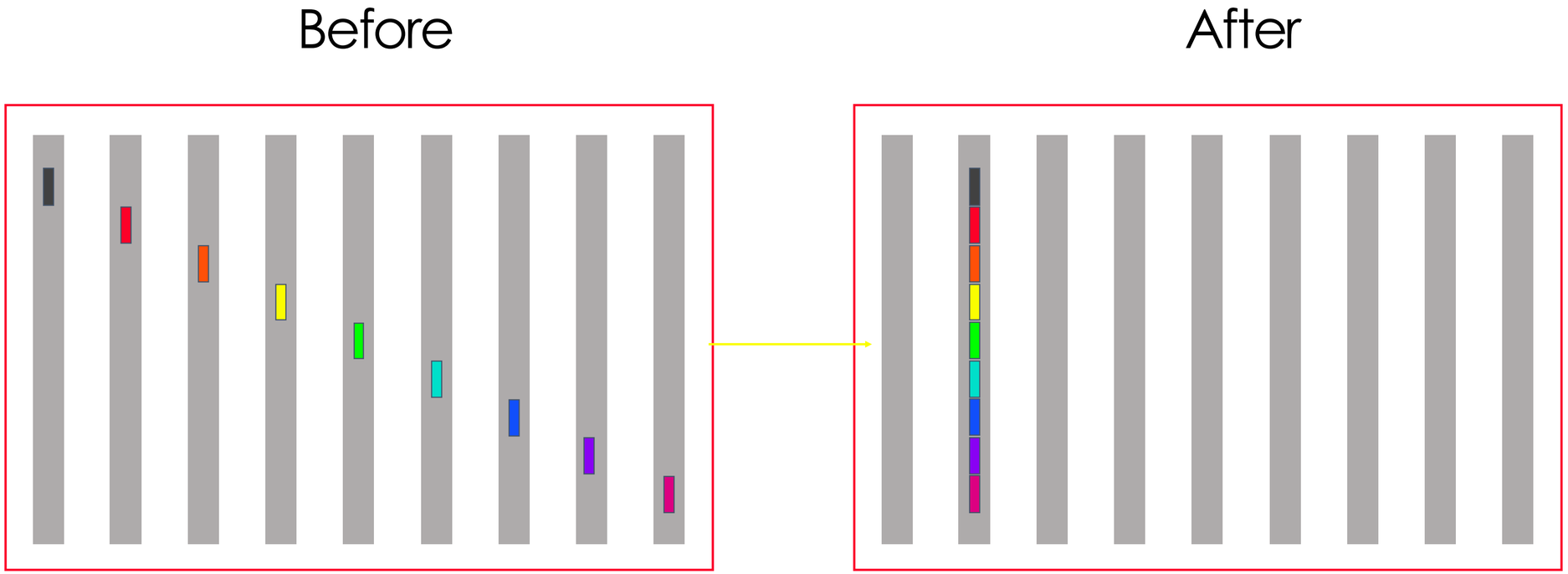
- Assumption: power of two number of nodes

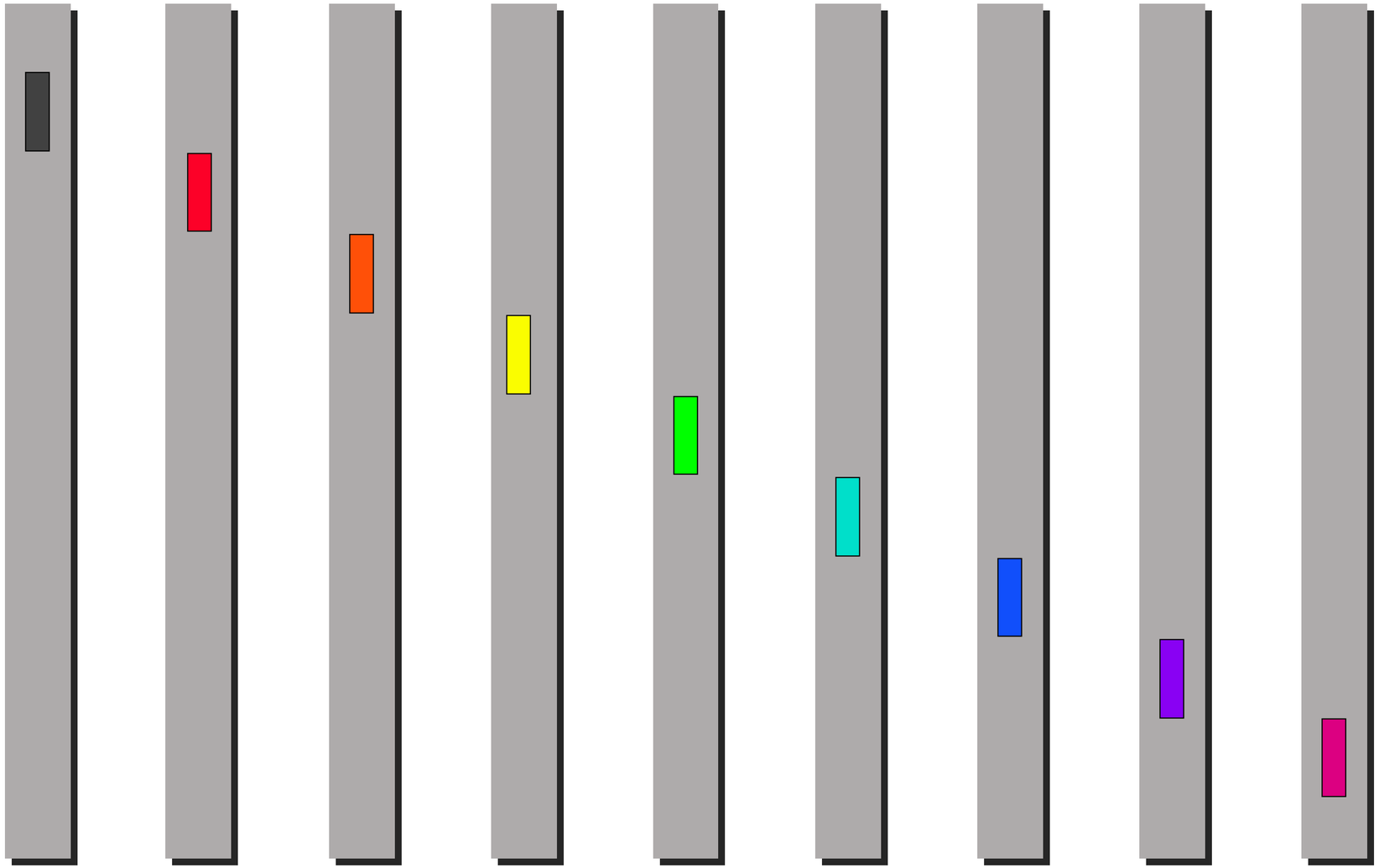
$$\sum_{k=1}^{log(p)} \left(\alpha + \frac{n}{2^k} \beta \right)$$

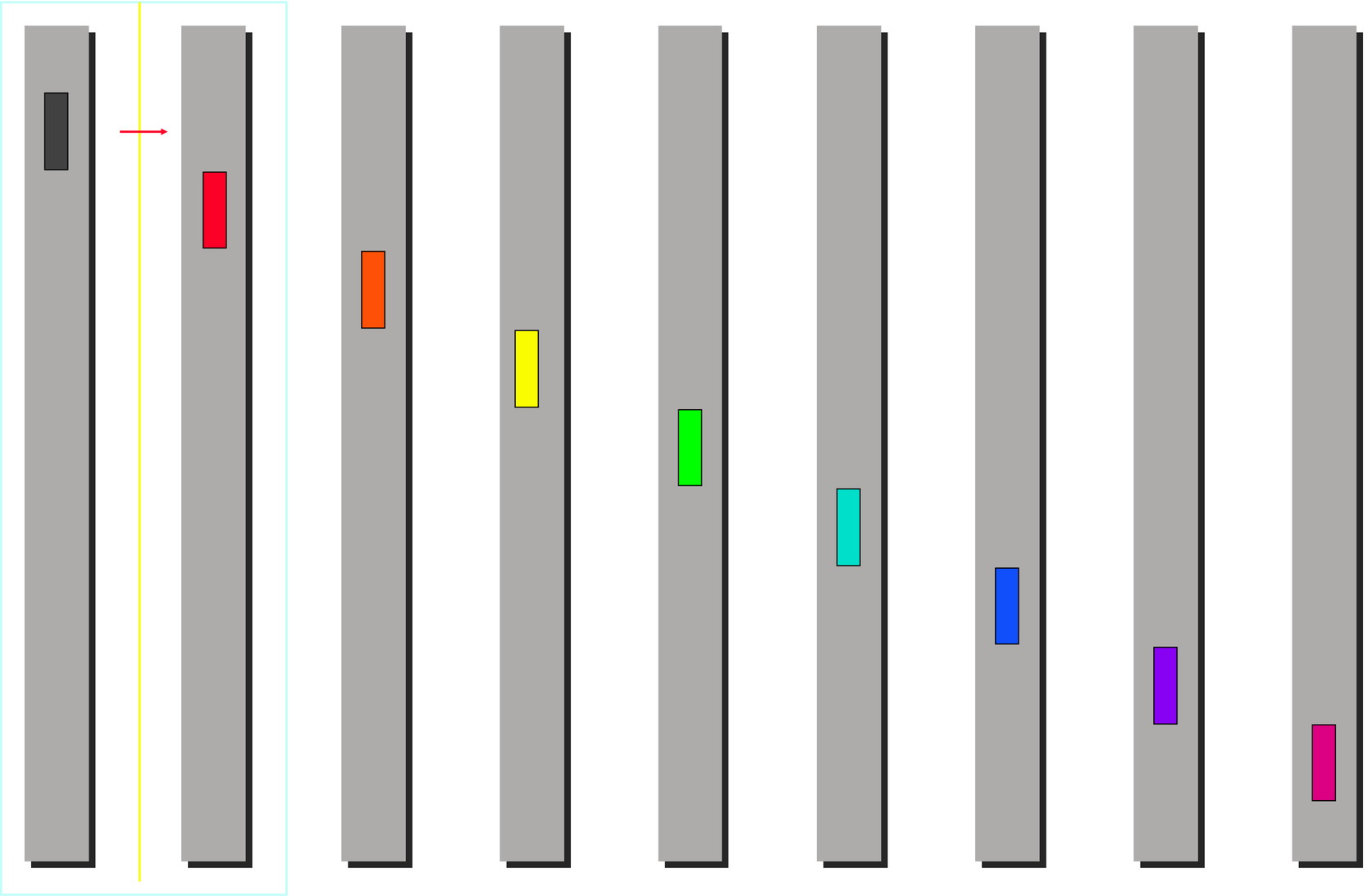
$$=$$

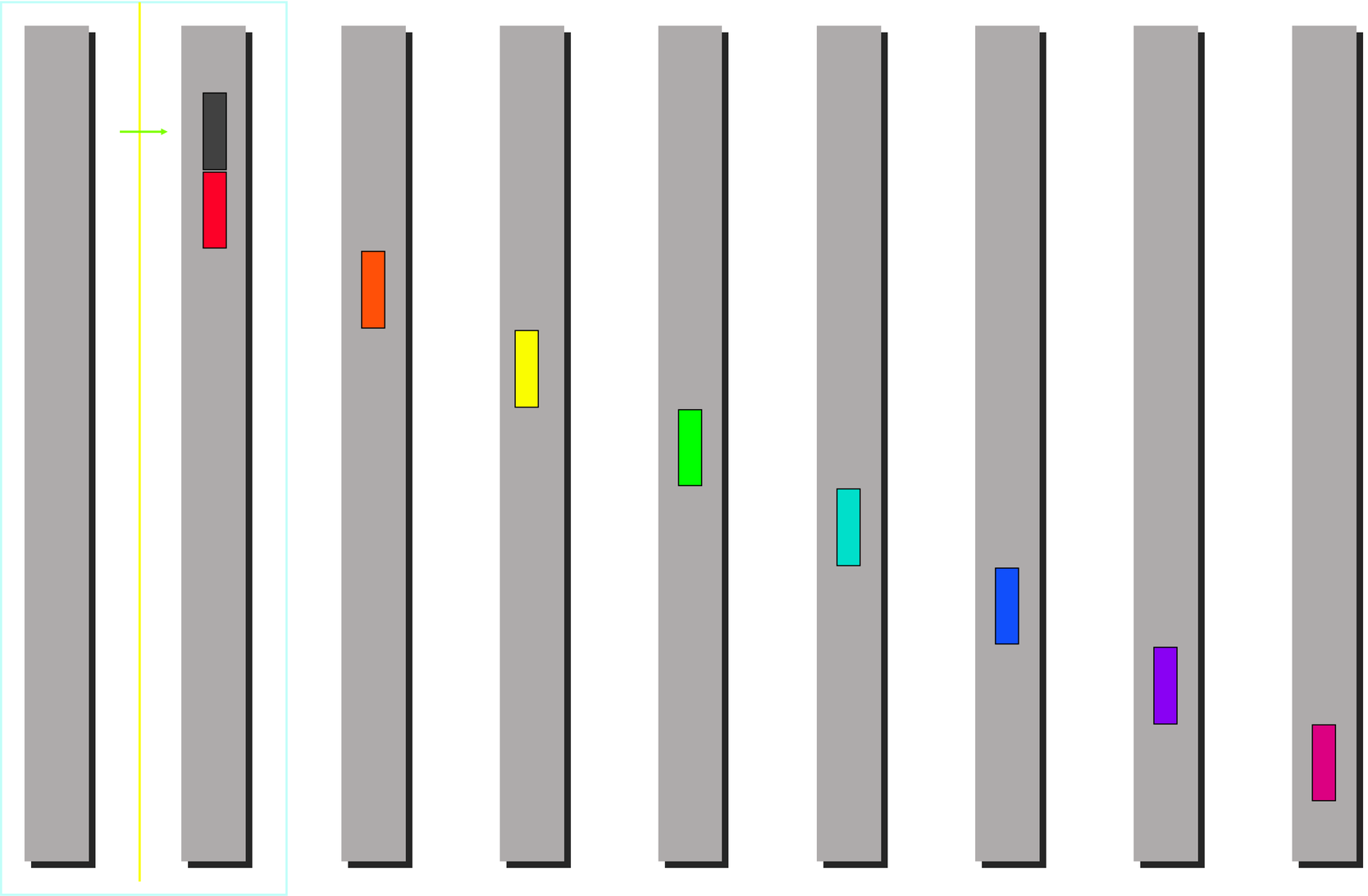
$$log(p)\ \alpha + \frac{p-1}{p} n\beta$$

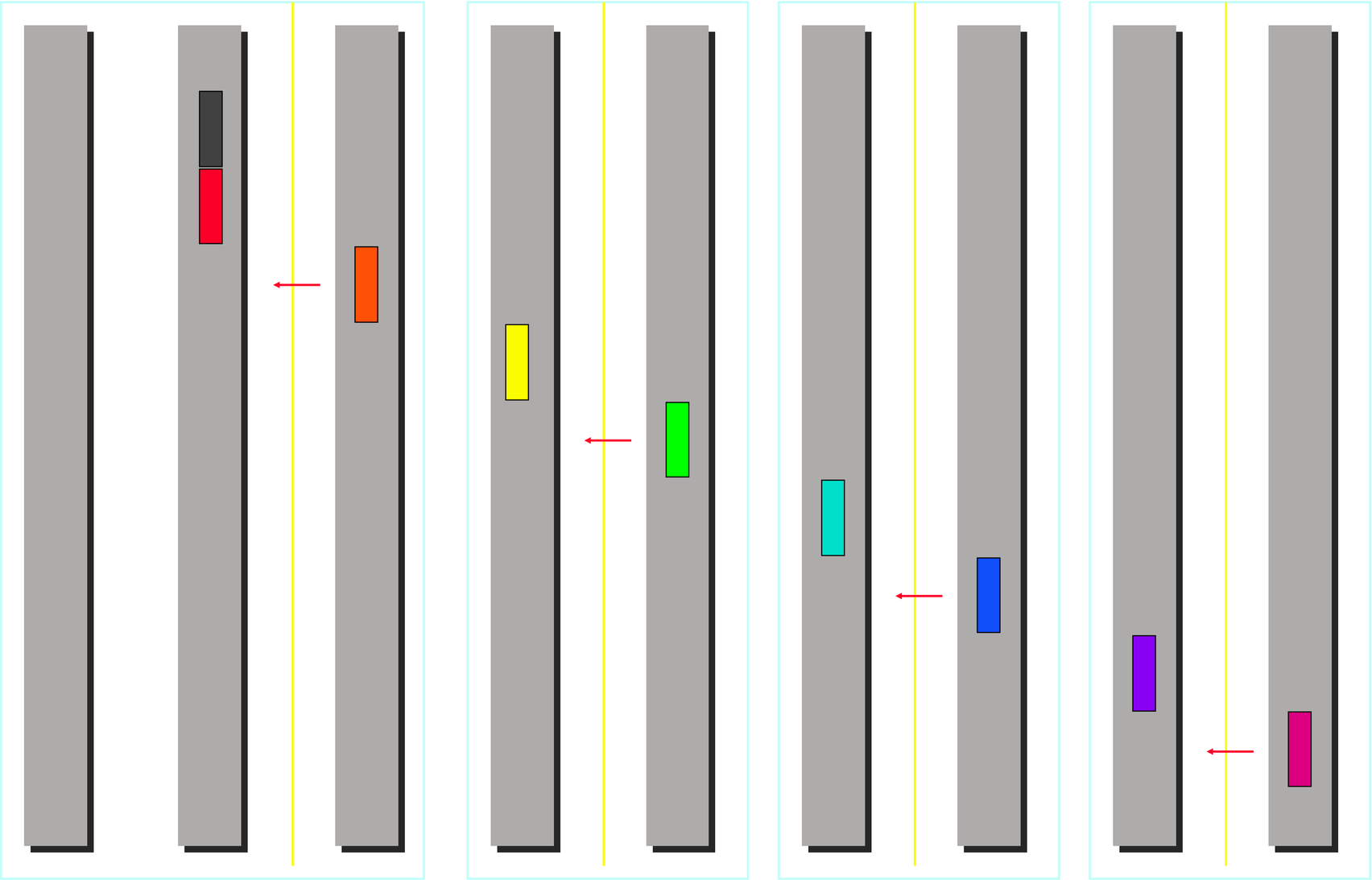
Gather

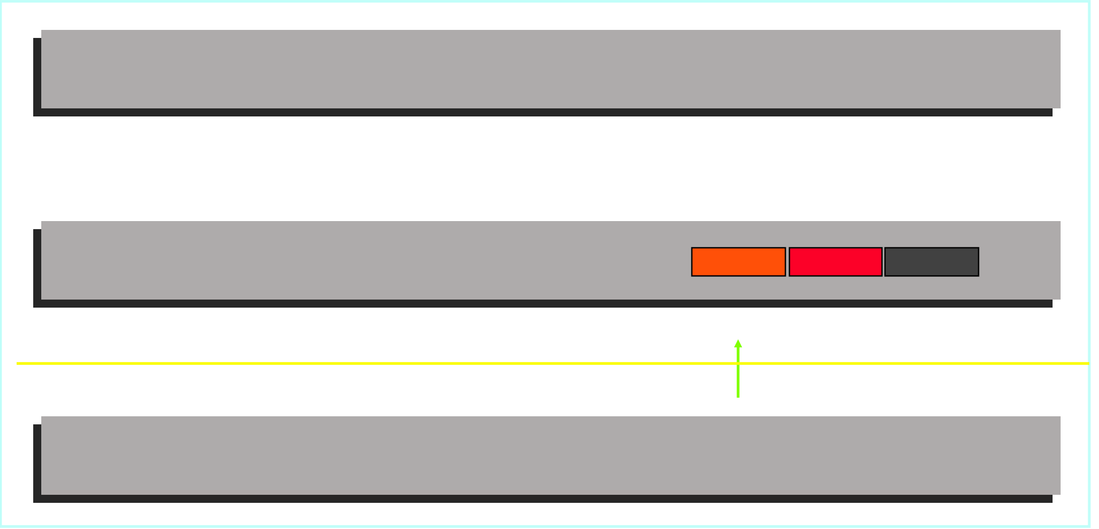


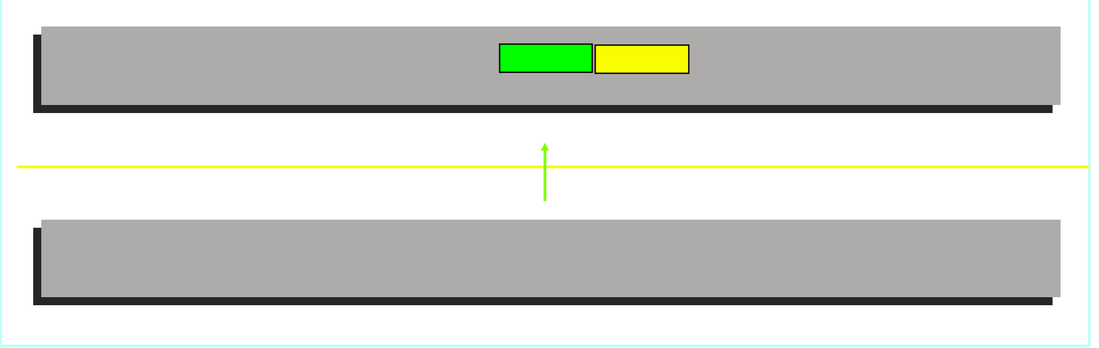


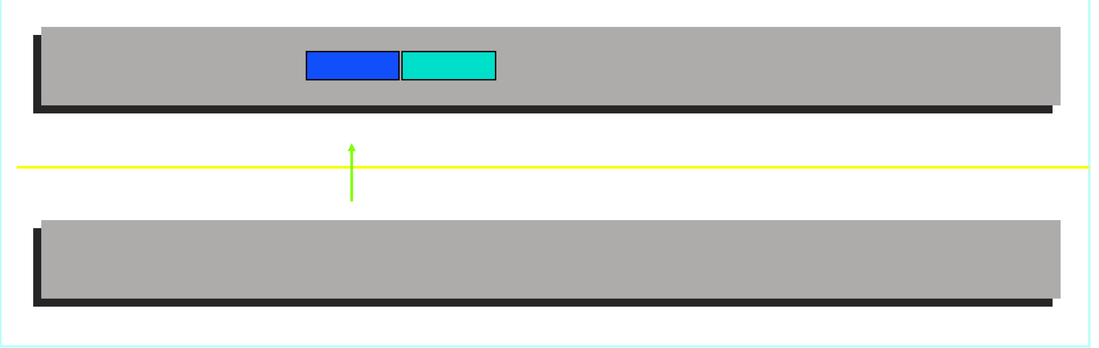


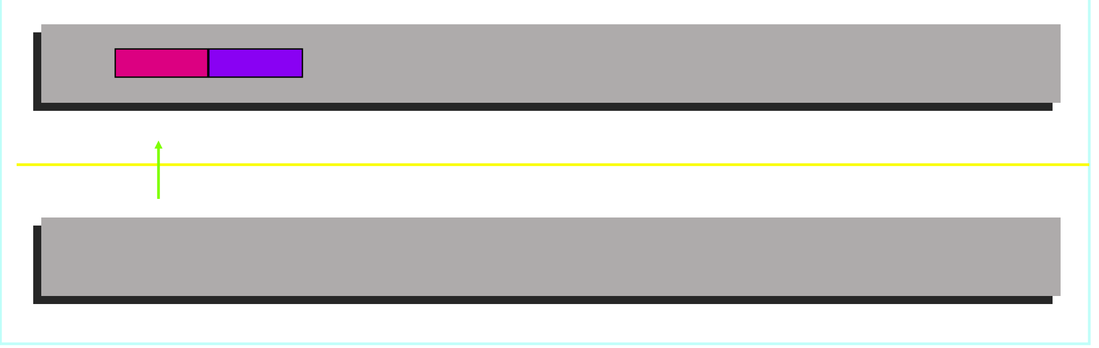


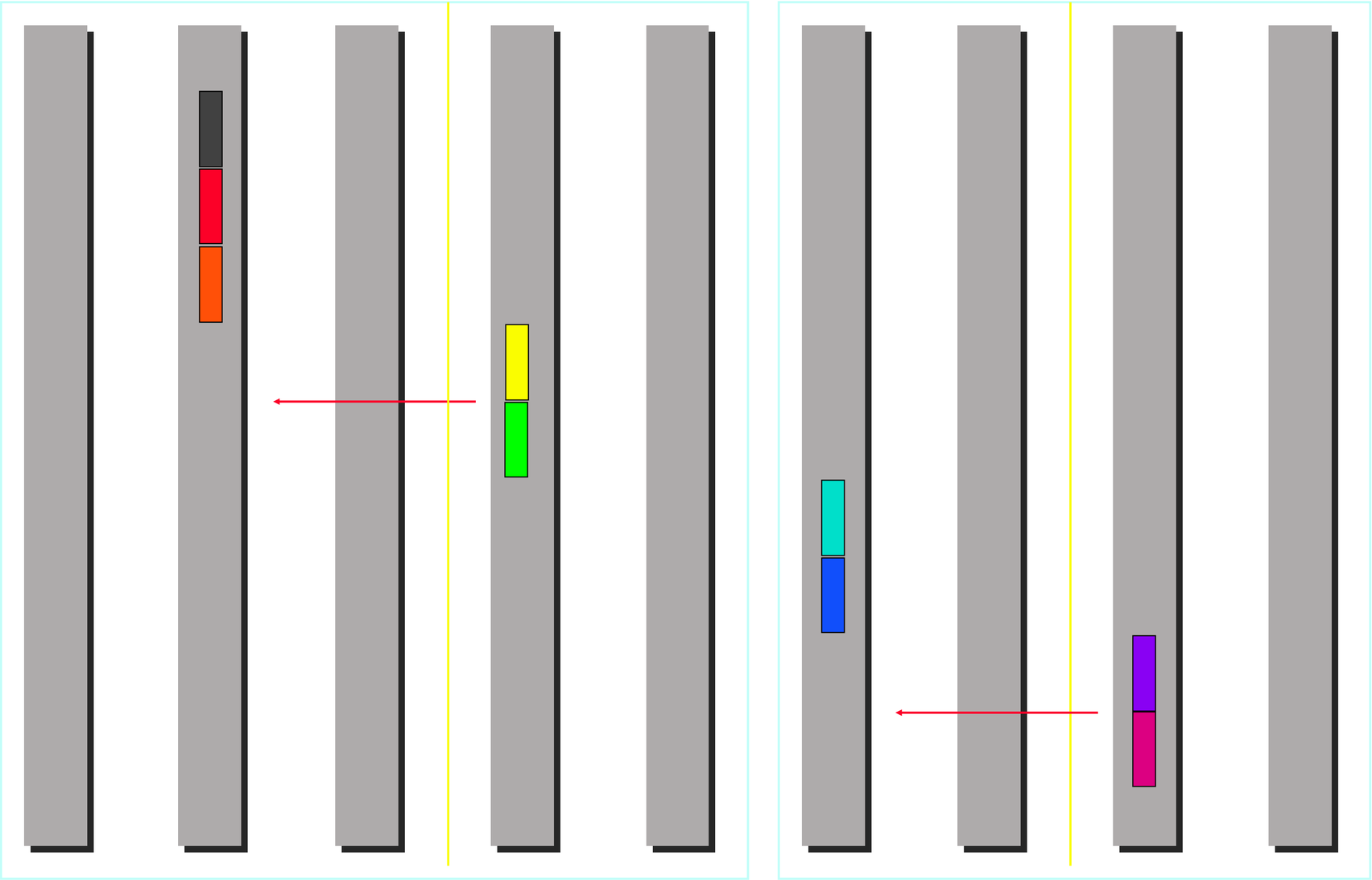


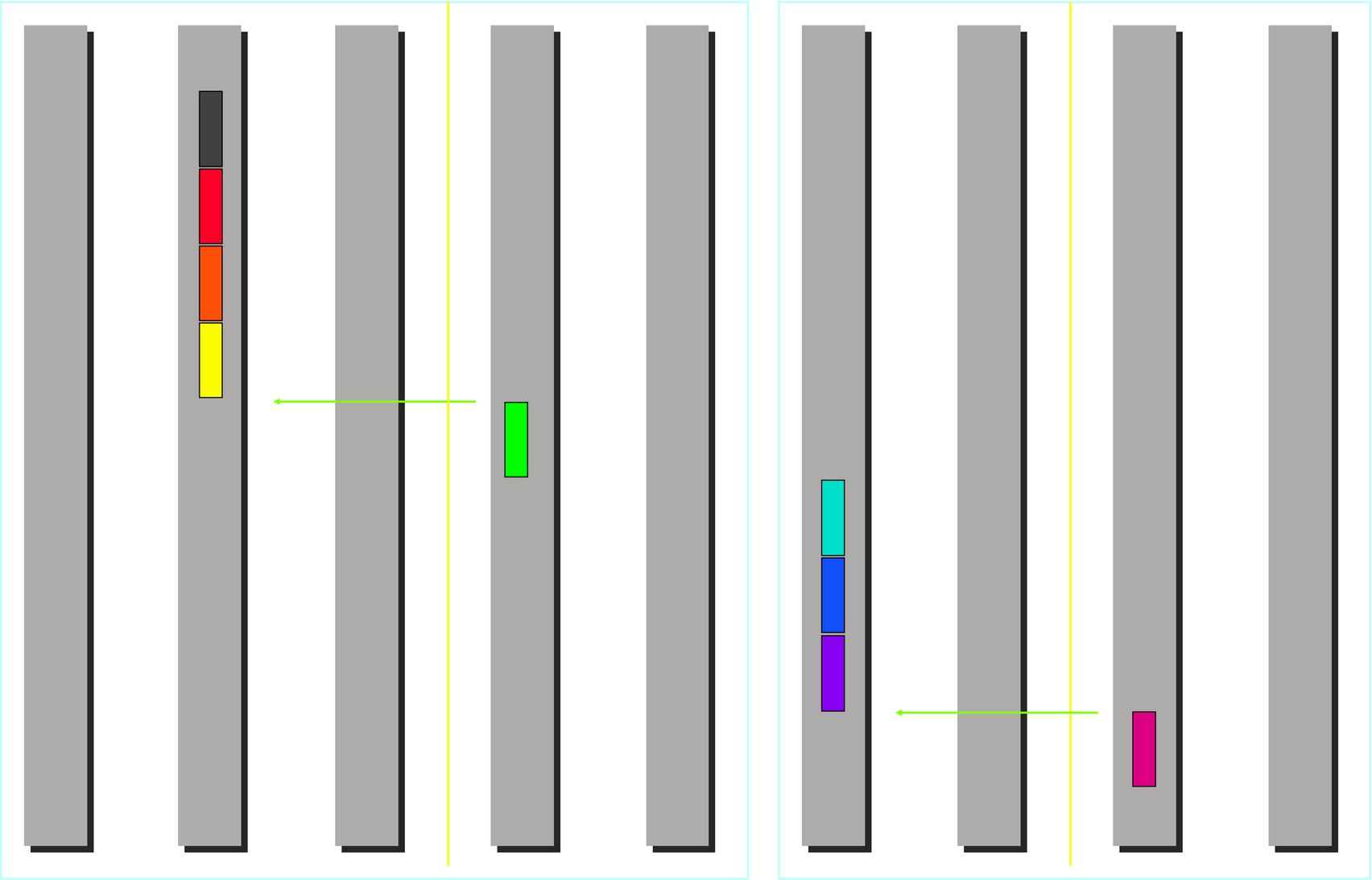


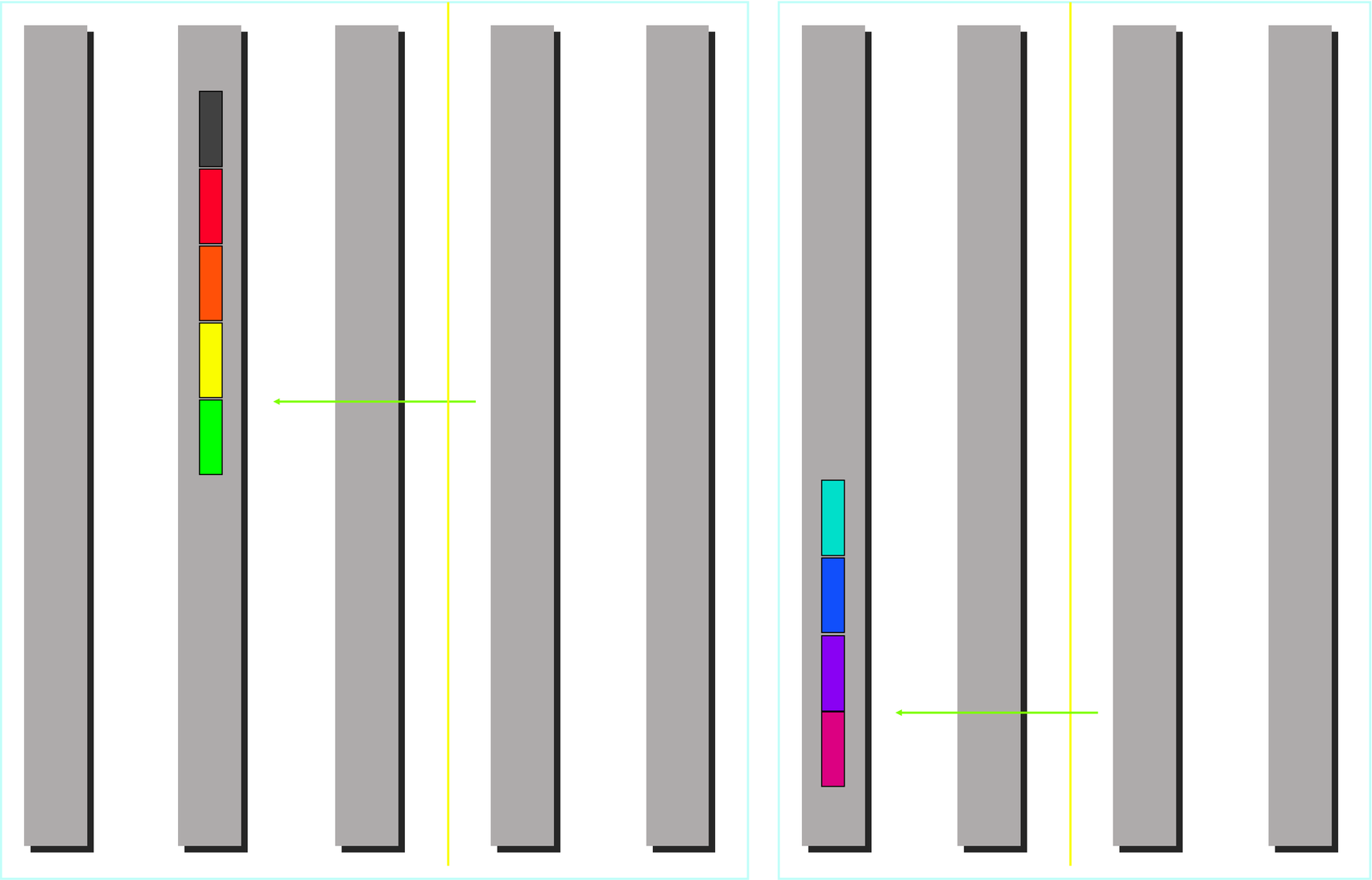


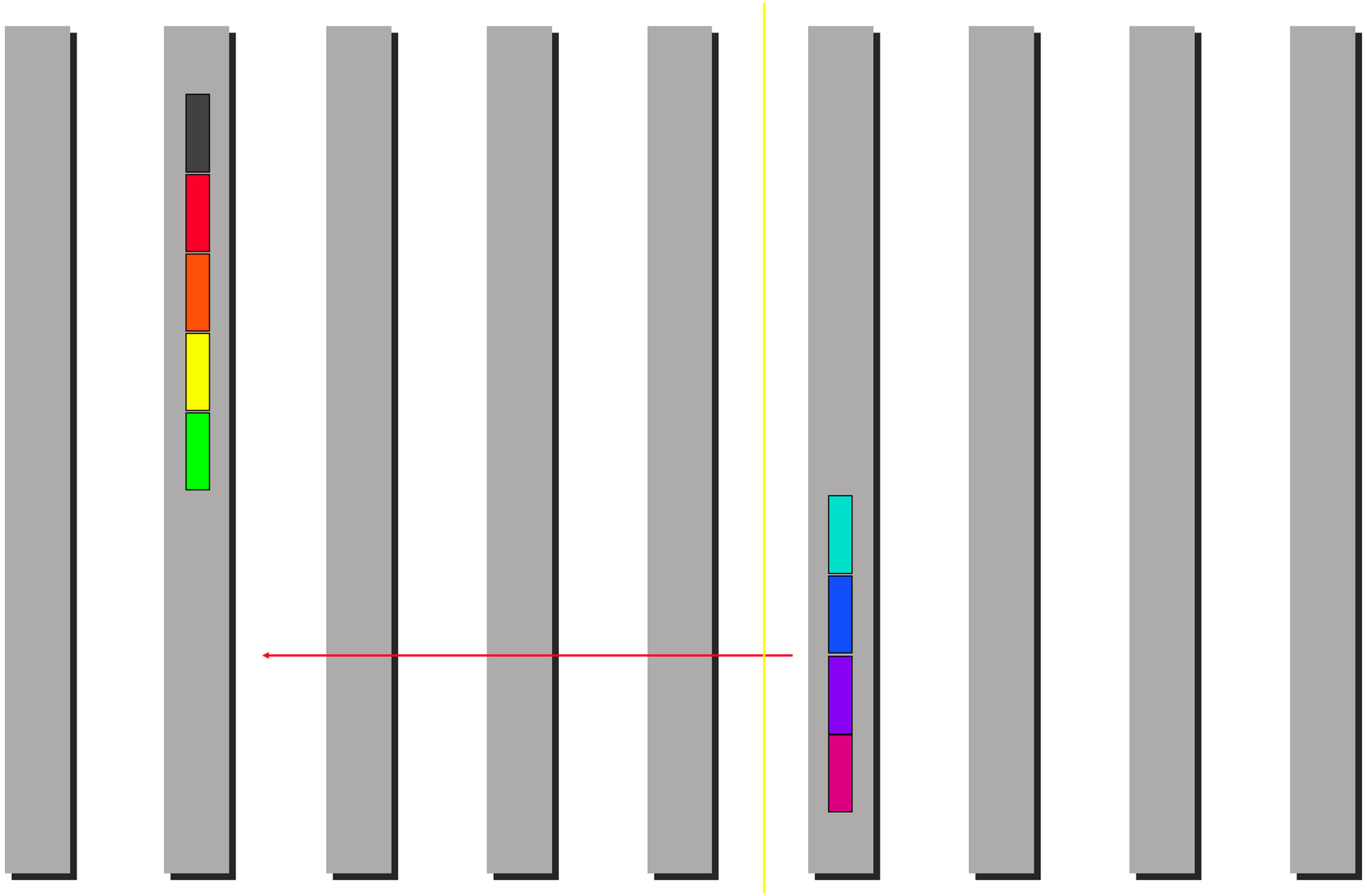


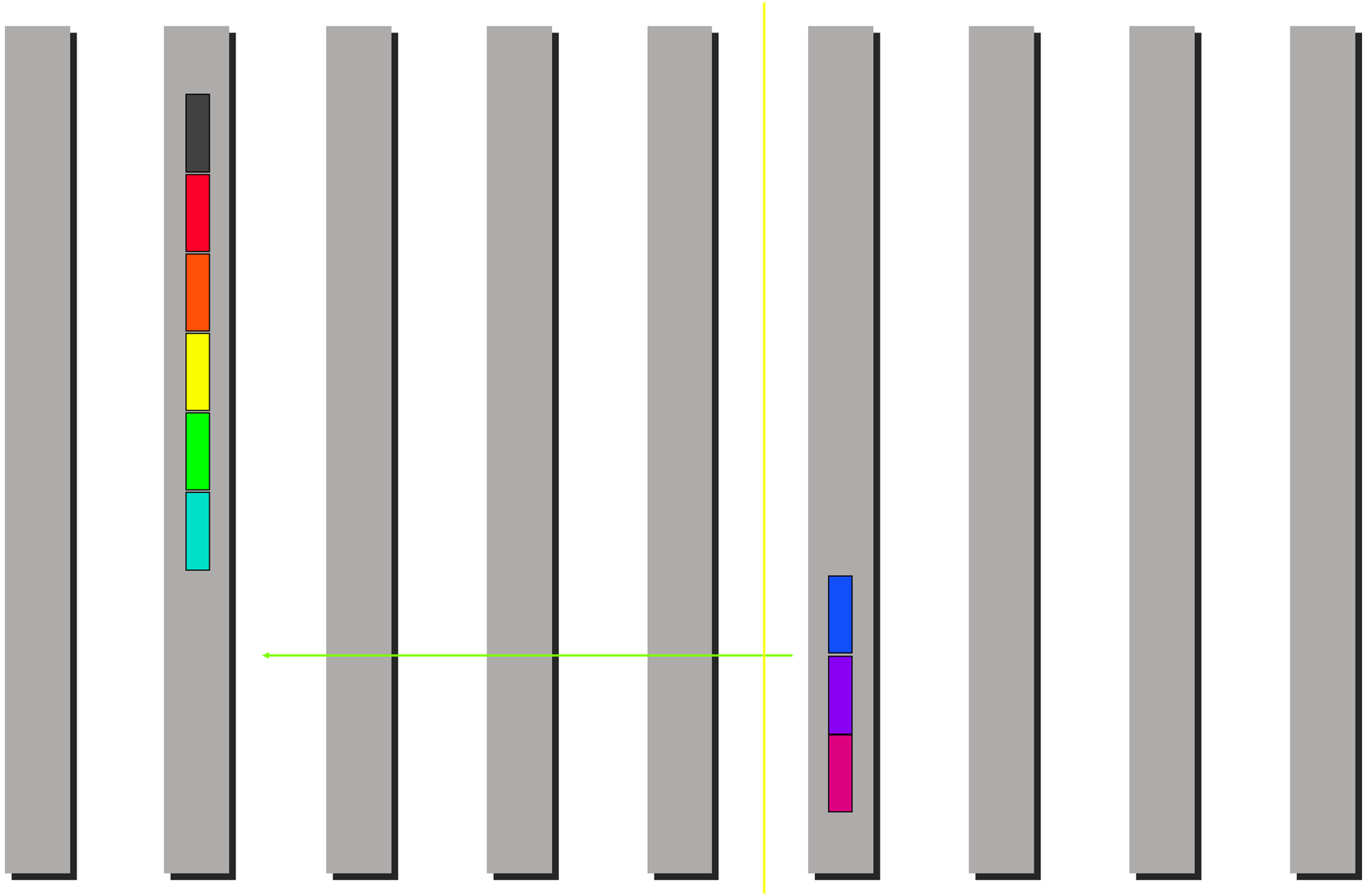


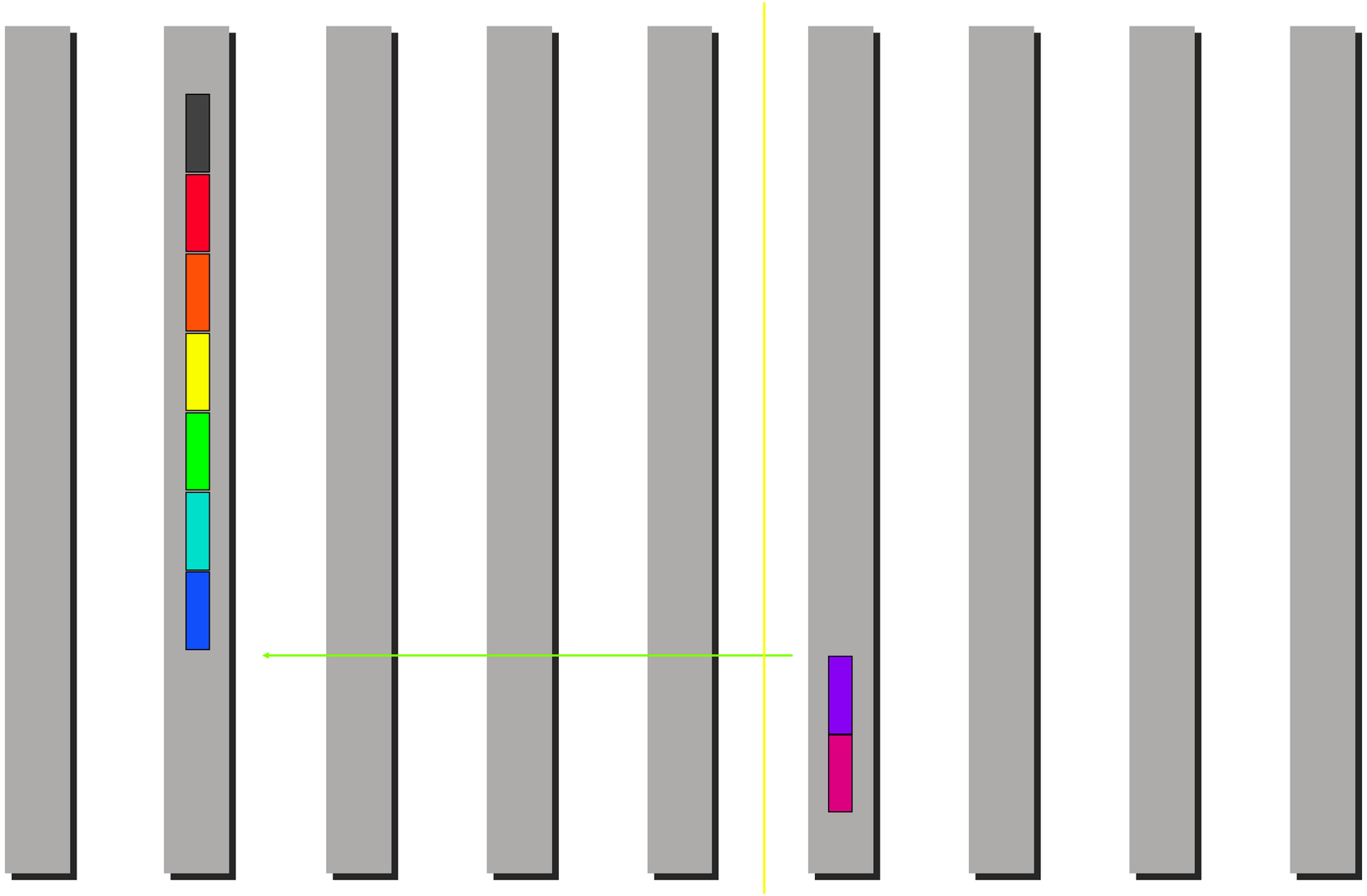


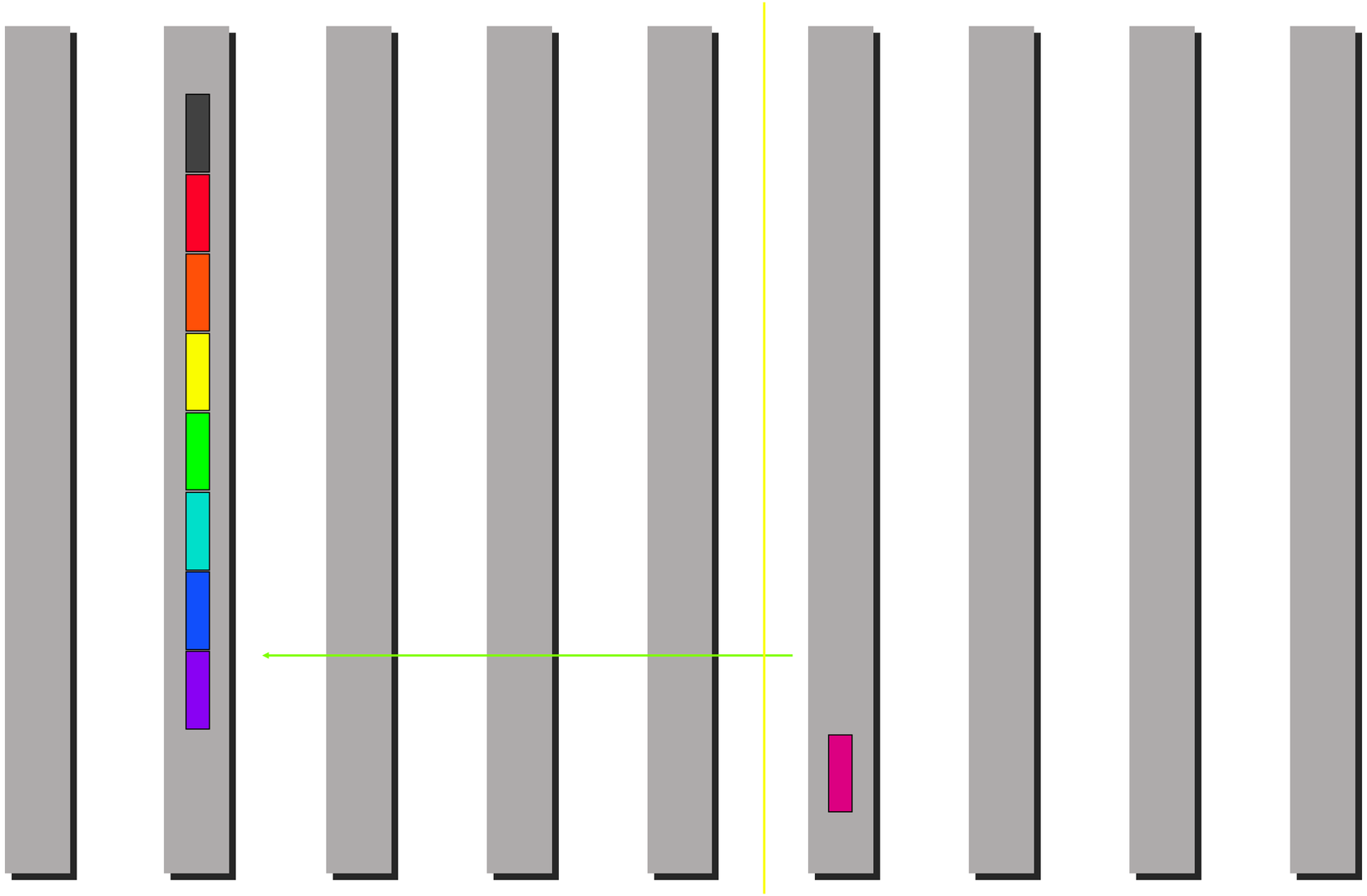


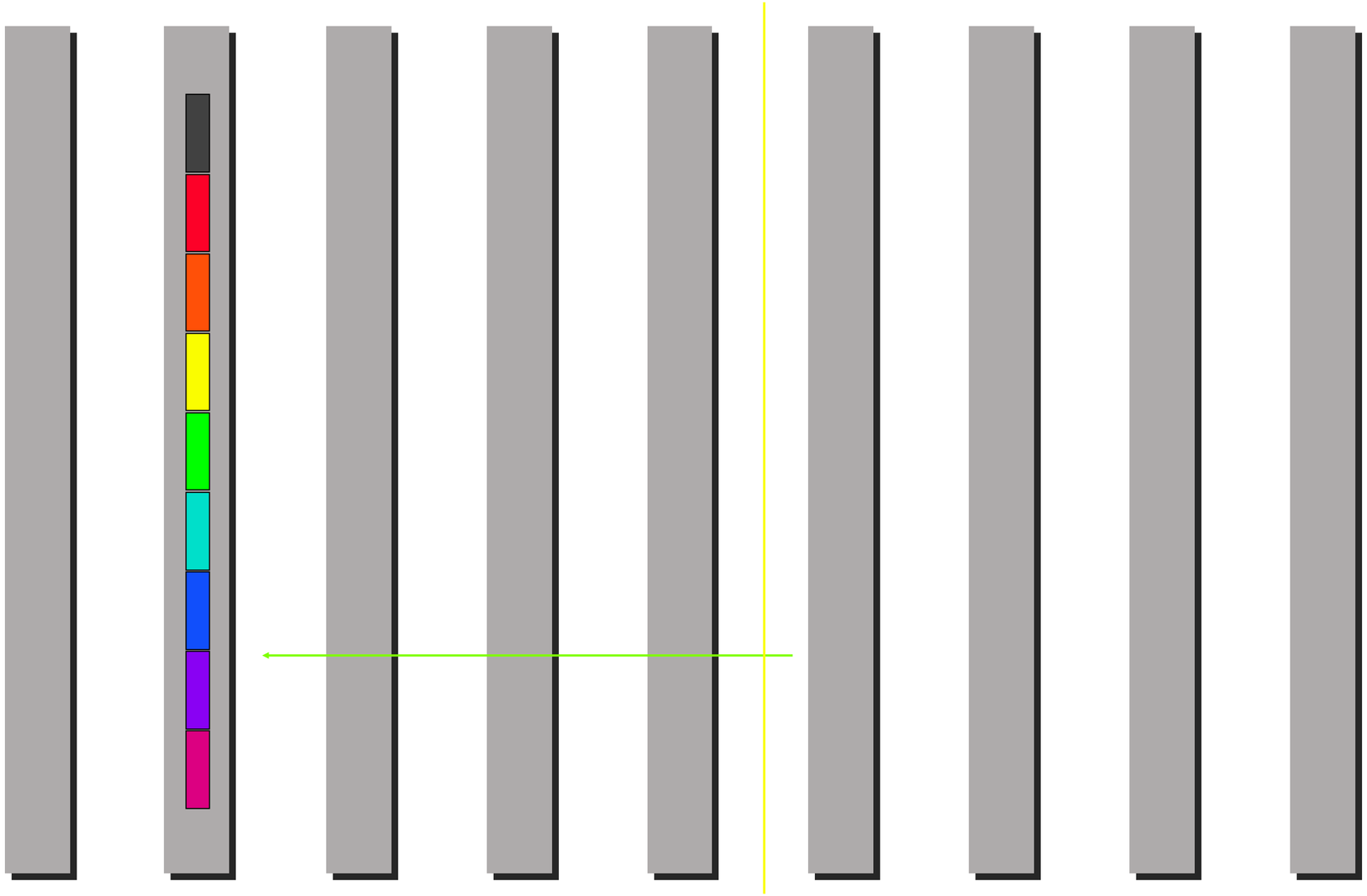


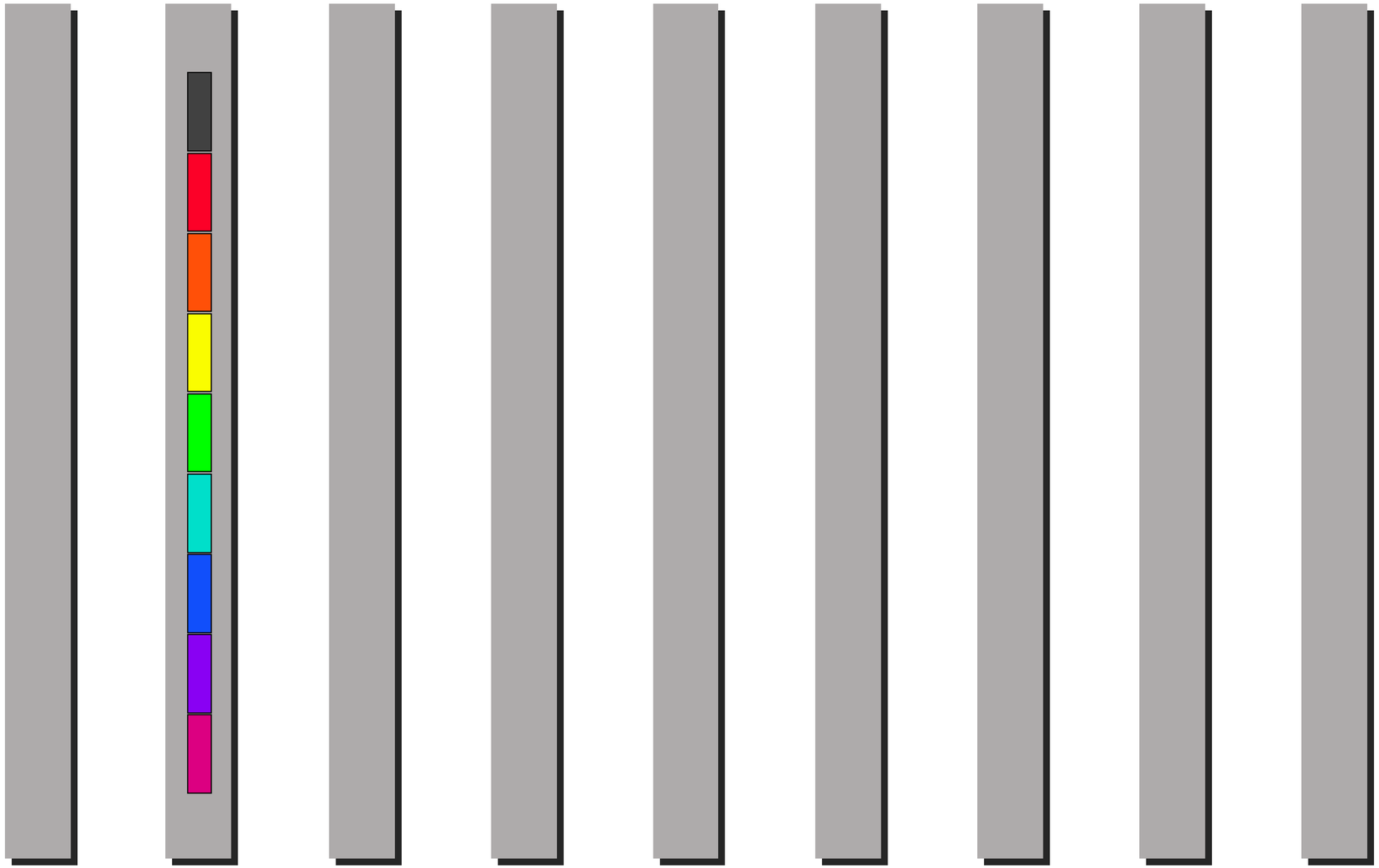












Cost of minimum spanning tree gather

- Assumption: power of two number of nodes

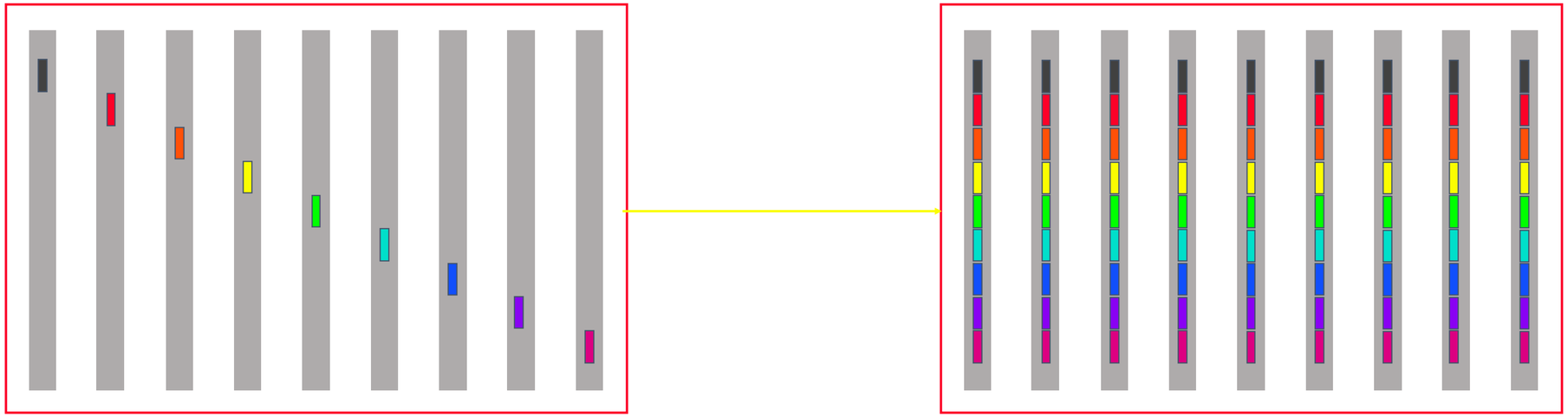
$$\sum_{k=1}^{log(p)} \left(\alpha + \frac{n}{2^k} \beta \right)$$

$$=$$

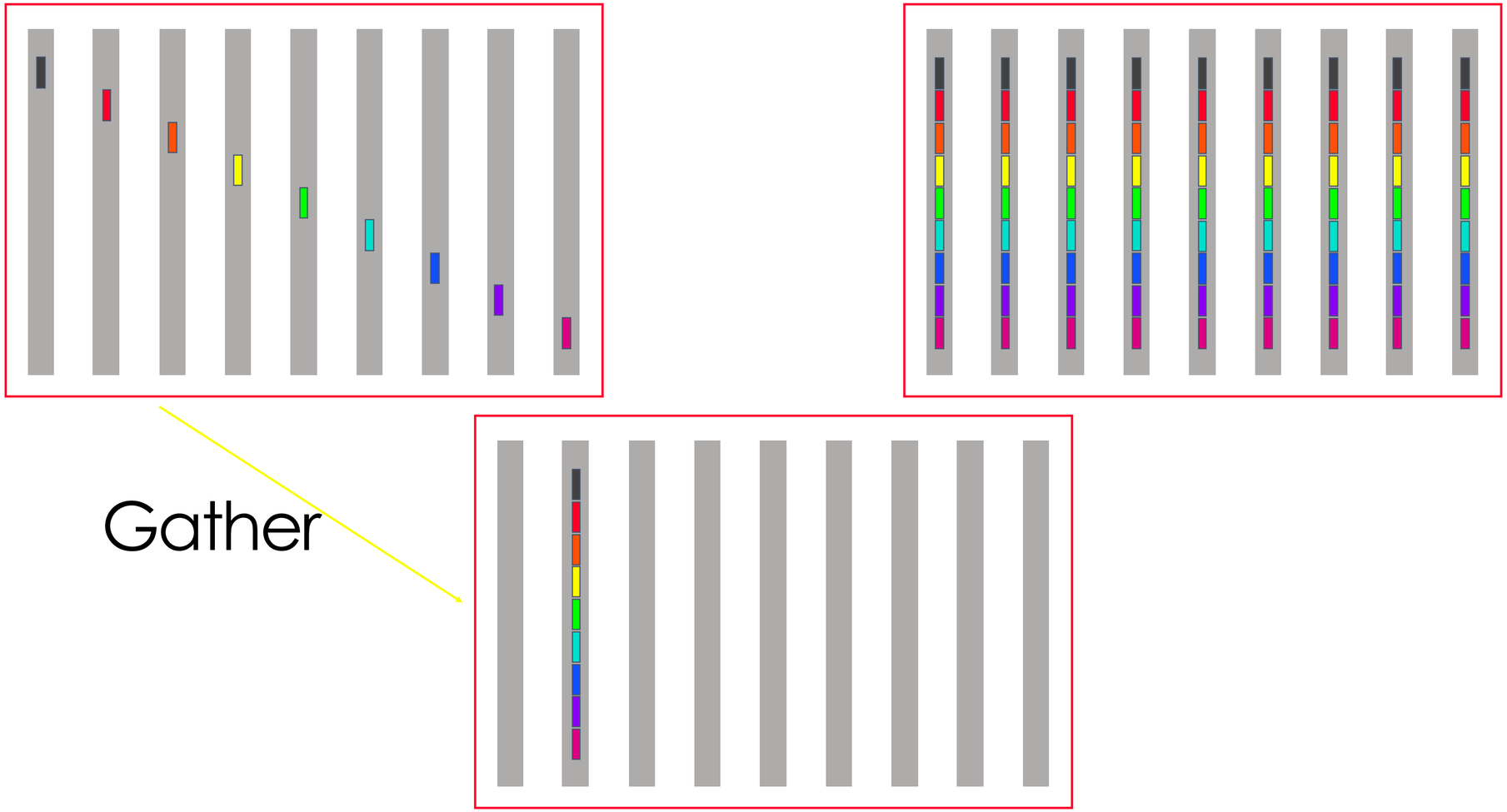
$$log(p)\ \alpha + \frac{p-1}{p} n\beta$$

Using the building blocks

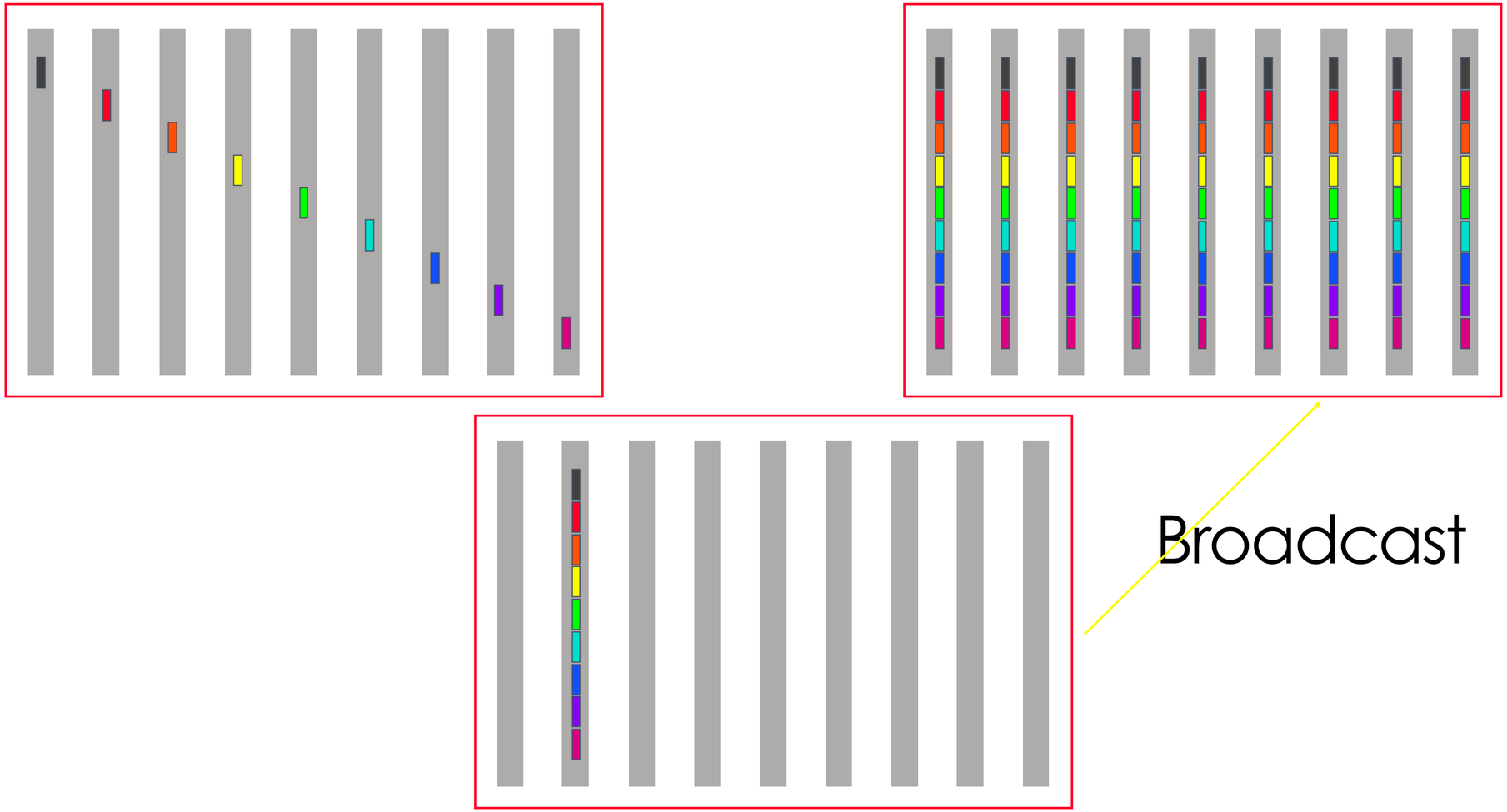
Allgather



Allgather



Allgather (short vector)



Cost of gather/broadcast allgather

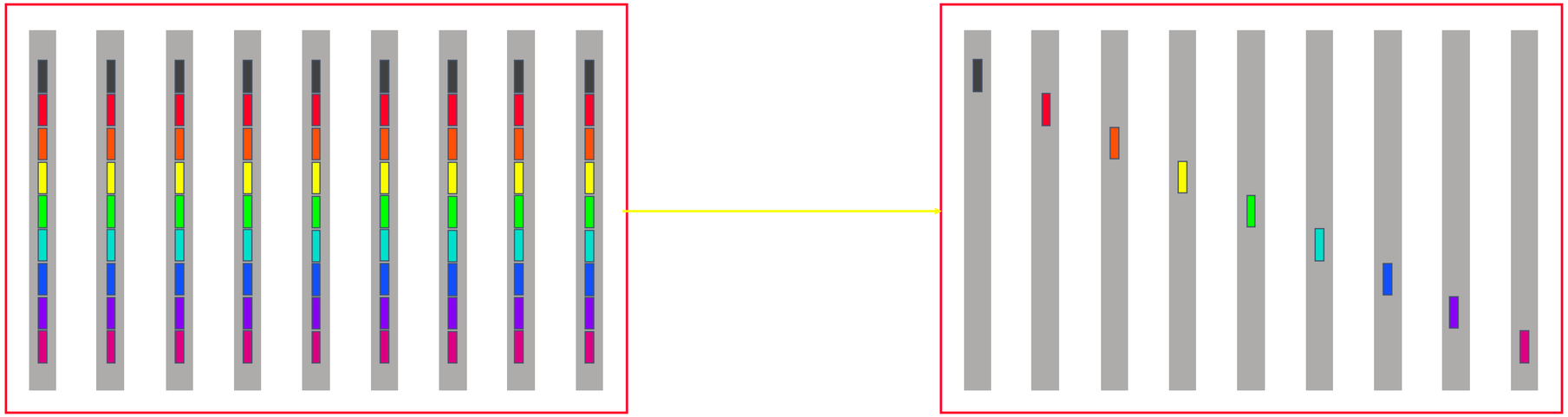
- Assumption: power of two number of nodes

gather $\log(p)\alpha + \frac{p-1}{p} n\beta$

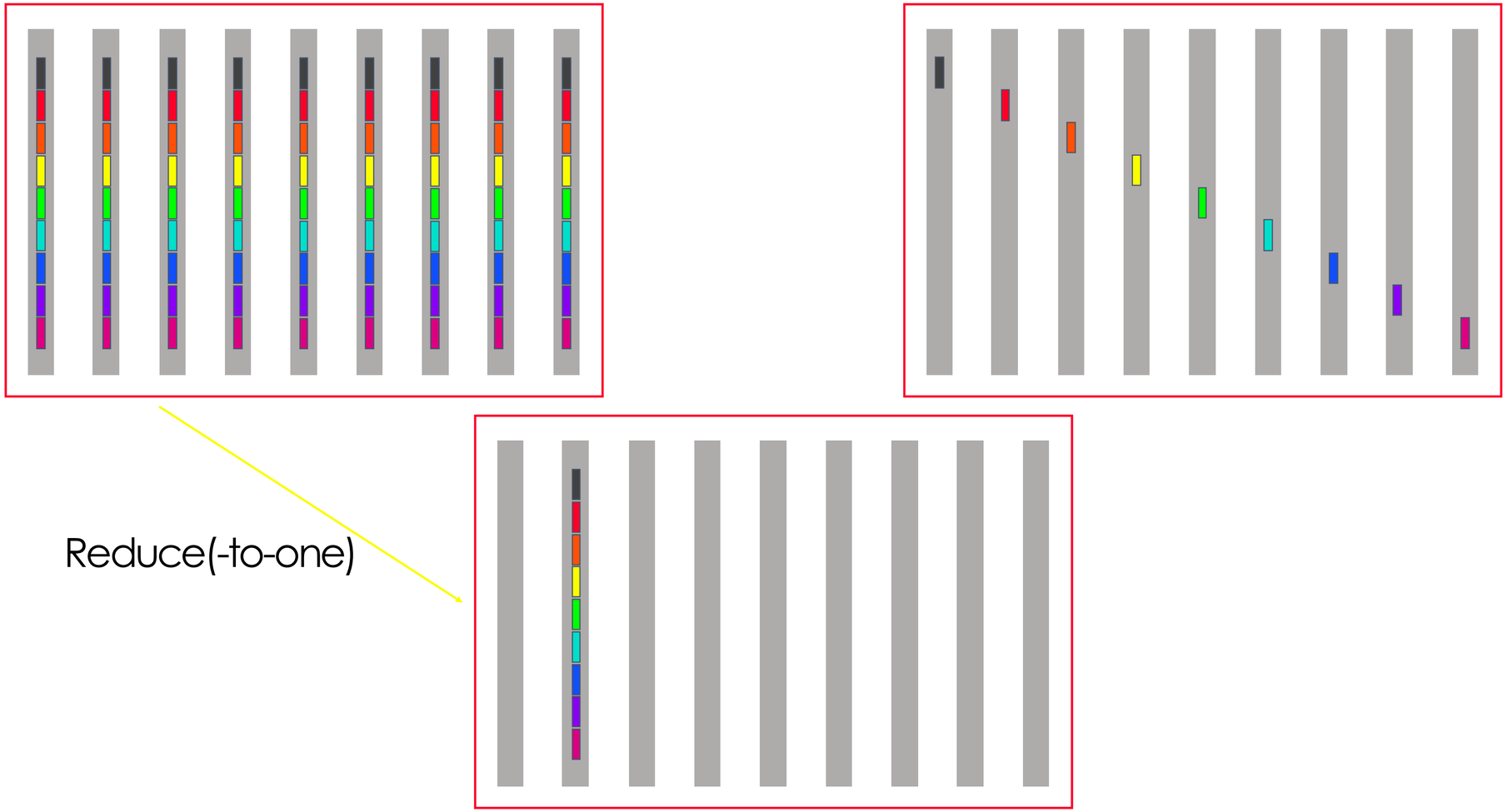
broadcast $\log(p)(\alpha + n\beta)$

$$2\log(p)\alpha + \left(\frac{p-1}{p} + \log(p)\right) n\beta$$

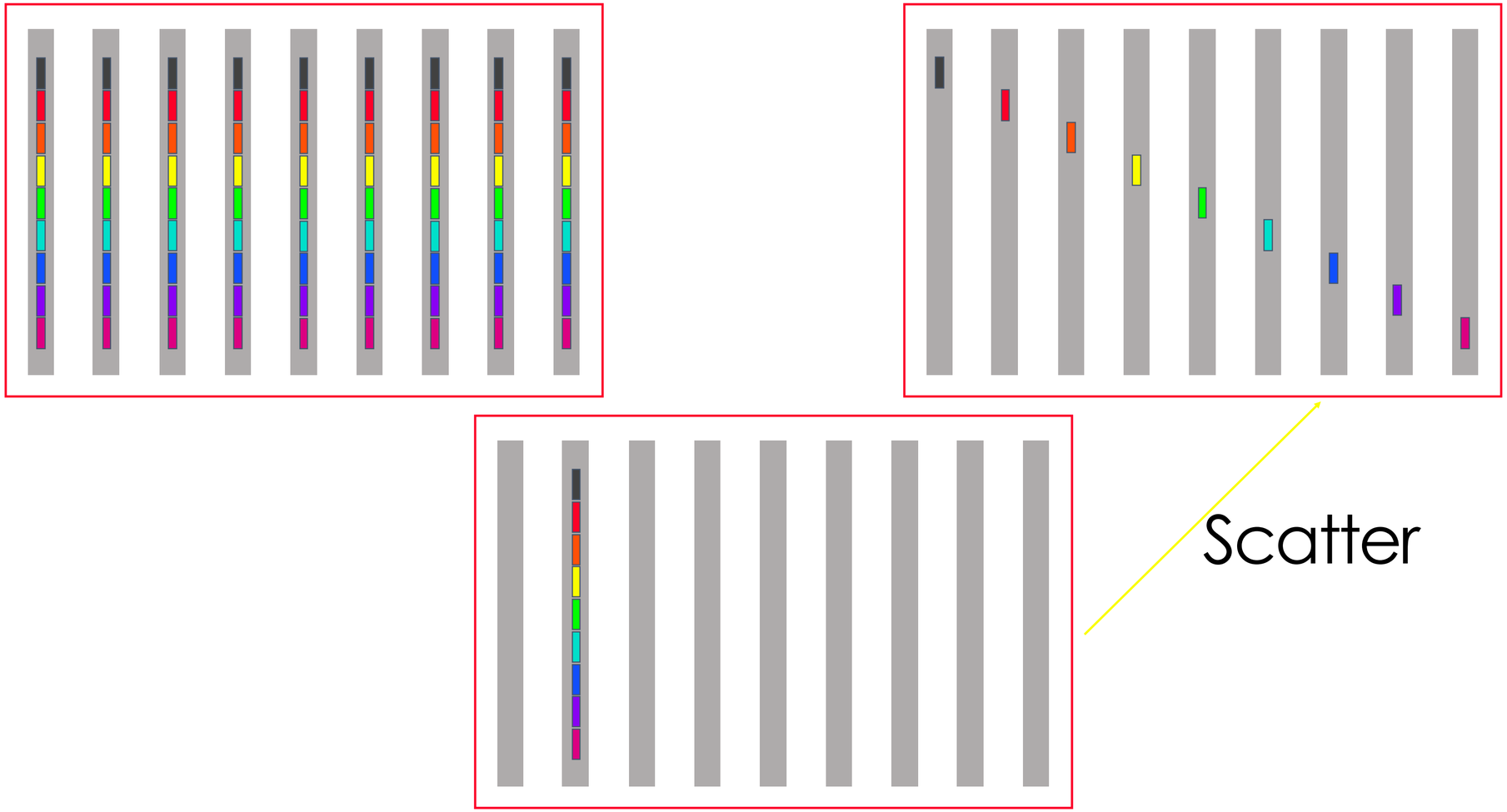
Reduce-scatter (small message)



Reduce-scatter
(short vector)



Reduce-scatter
(short vector)



Cost of Reduce(-to-one)/scatter Reduce-scatter

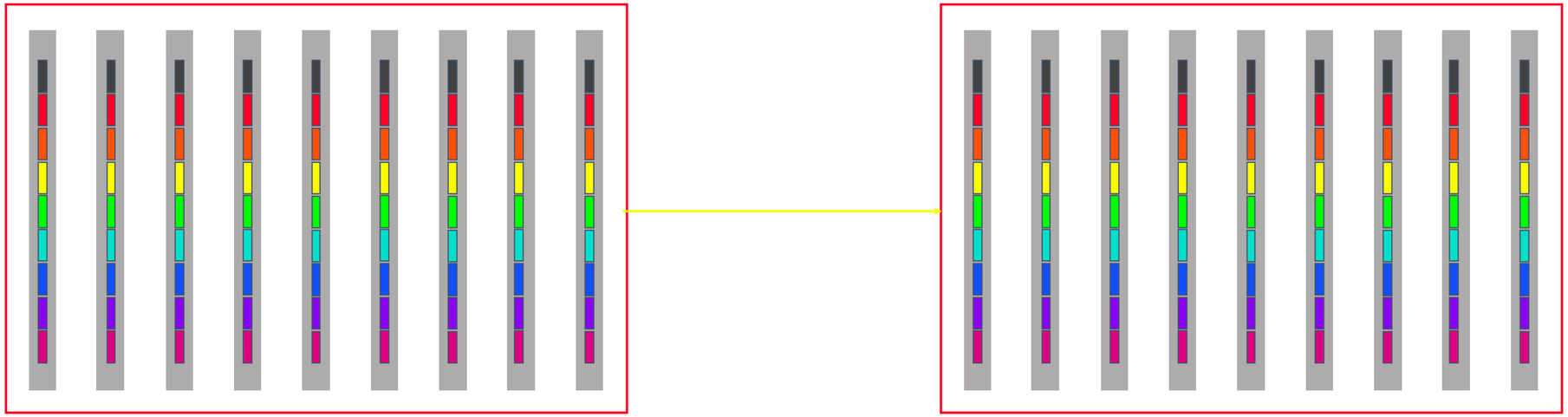
- Assumption: power of two number of nodes

Reduce(-to-one) $\log(p)(\alpha + n\beta + n\gamma)$

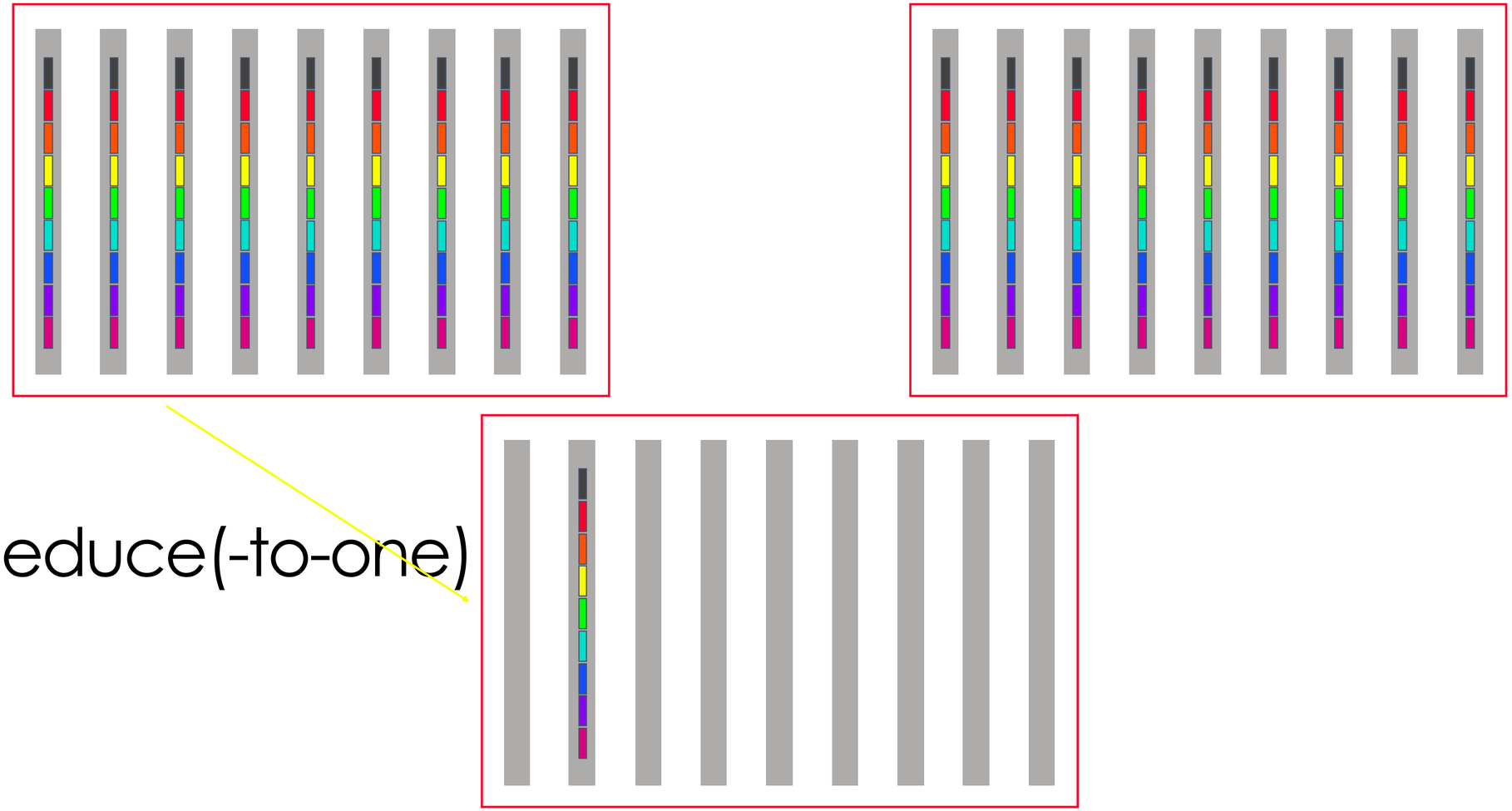
scatter $\log(p)\alpha + \frac{p-1}{p} n\beta$

$$2\log(p)\alpha + \left(\frac{p-1}{p} + \log(p)\right) n\beta + \log(p) n\gamma$$

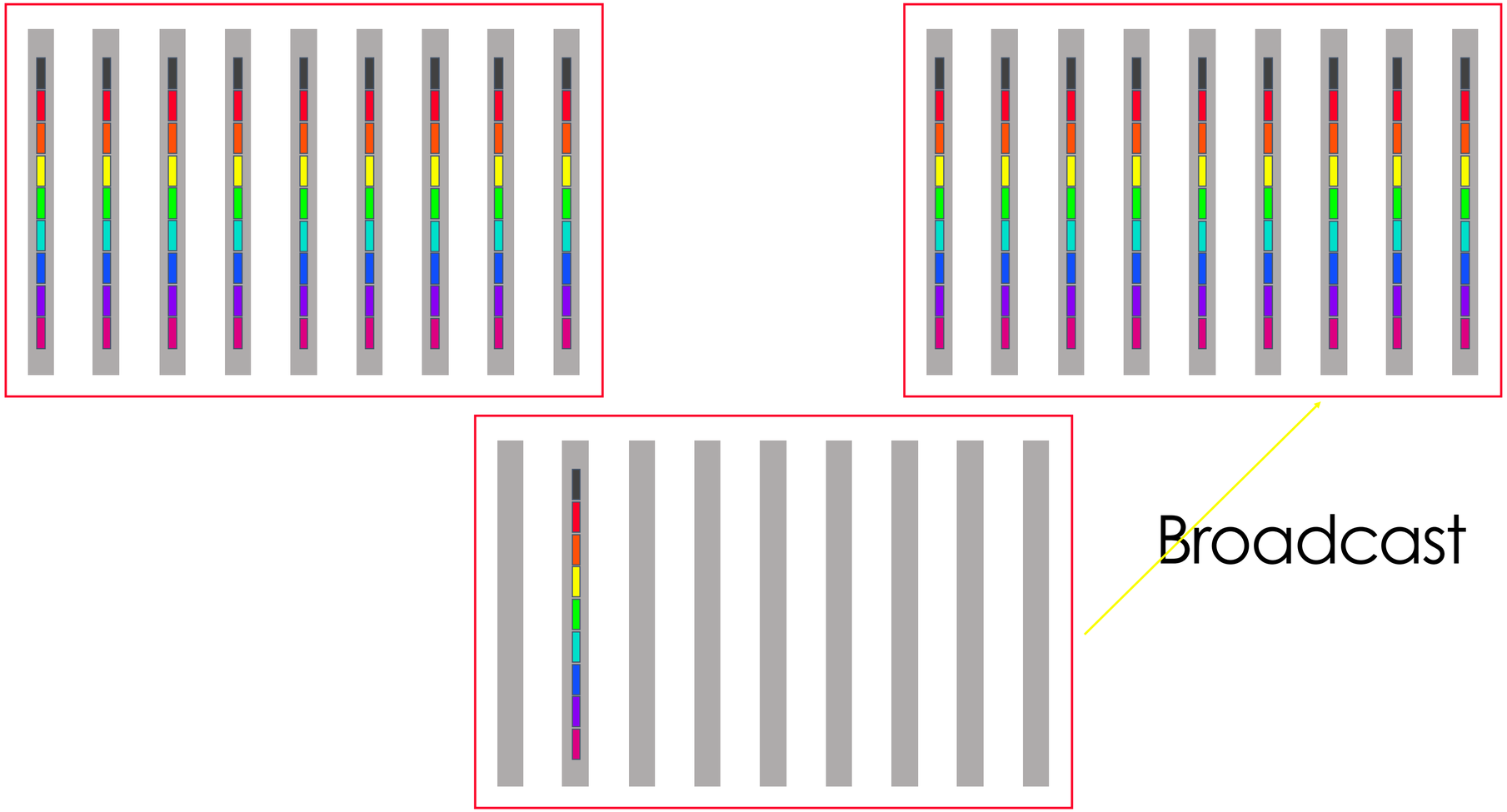
Allreduce (Latency-optimized)



Allreduce (Latency-optimized)



Allreduce
(short vector)



Cost of reduce(-to-one)/broadcast Allreduce

- Assumption: power of two number of nodes

Reduce(-to-one) $log(p)(\alpha + n\beta + n\gamma)$

broadcast $log(p)(\alpha + n\beta)$

$$2log(p)\alpha + 2log(p)n\beta + log(p)n\gamma$$

Recap

Reduce(-to-one)

$$log(p)(\alpha + n\beta + n\gamma)$$

Scatter

$$log(p)\cancel{\alpha + \frac{p-1}{p} n\beta}$$

Gather

$$log(p)\cancel{\alpha + \frac{p-1}{p} n\beta}$$

Broadcast

$$log(p)(\alpha + n\beta)$$

Reduce-scatter

Allreduce

Allgather

Recap

Reduce(-to-one)

$$log(p)(\alpha + n\beta + n\gamma)$$

Scatter

$$log(p)\cancel{\alpha + \frac{p-1}{p} n\beta}$$

Gather

$$log(p)\cancel{\alpha + \frac{p-1}{p} n\beta}$$

Broadcast

$$log(p)(\alpha + n\beta)$$

Reduce-scatter

$$2log(p)\alpha + log(p)n(\beta + \gamma)\cancel{+ \frac{p-1}{p} n\beta}$$

Allreduce

Allgather

Recap

Reduce(-to-one)

$$\log(p)(\alpha + n\beta + n\gamma)$$

Scatter

$$\log(p)\cancel{\alpha + \frac{p-1}{p} n\beta}$$

Gather

$$\log(p)\cancel{\alpha + \frac{p-1}{p} n\beta}$$

Broadcast

$$\log(p)(\alpha + n\beta)$$

Reduce-scatter

$$2\log(p)\alpha + \log(p)n(\beta + \gamma)\cancel{+ \frac{p-1}{p} n\beta}$$

Allreduce

$$2\log(p)\alpha + \log(p)n(2\beta + \gamma)$$

Allgather

$$2\log(p)\alpha + \log(p)n\beta\cancel{+ \frac{p-1}{p} n\beta}$$

Recap

Reduce(-to-one)

$$\log(p)(\alpha + n\beta + n\gamma)$$

Scatter

$$\log(p)\cancel{\alpha + \frac{p-1}{p} n\beta}$$

Gather

$$\log(p)\cancel{\alpha + \frac{p-1}{p} n\beta}$$

Broadcast

$$\log(p)(\alpha + n\beta)$$

Reduce-scatter

$$2\log(p)\alpha + \log(p)n(\beta + \gamma)\cancel{+ \frac{p-1}{p} n\beta}$$

Allreduce

$$2\log(p)\alpha + \log(p)n(2\beta + \gamma)$$

Allgather

Recap

Reduce(-to-one)

$$\log(p)(\alpha + n\beta + n\gamma)$$

Scatter

$$\log(p)\cancel{\alpha + \frac{p-1}{p} n\beta}$$

Gather

$$\log(p)\cancel{\alpha + \frac{p-1}{p} n\beta}$$

Broadcast

$$\log(p)(\alpha + n\beta)$$

Reduce-scatter

$$2\log(p)\alpha + \log(p)n(\beta + \gamma)\cancel{+ \frac{p-1}{p} n\beta}$$

Allreduce

$$2\log(p)\alpha + \log(p)n(2\beta + \gamma)$$

Allgather

$$2\log(p)\alpha + \log(p)n\cancel{\beta + \frac{p-1}{p} n\beta}$$

Summary of MST algorithms

- Small message: Minimum Spanning Tree algorithm
  - Emphasize **low latency**

- **Can we do better?**

- Problem of Minimum Spanning Tree Algorithm?
  - It prioritize latency rather than bandwidth
  - Hence: Some links are idle

- Next: Large message size algorithm

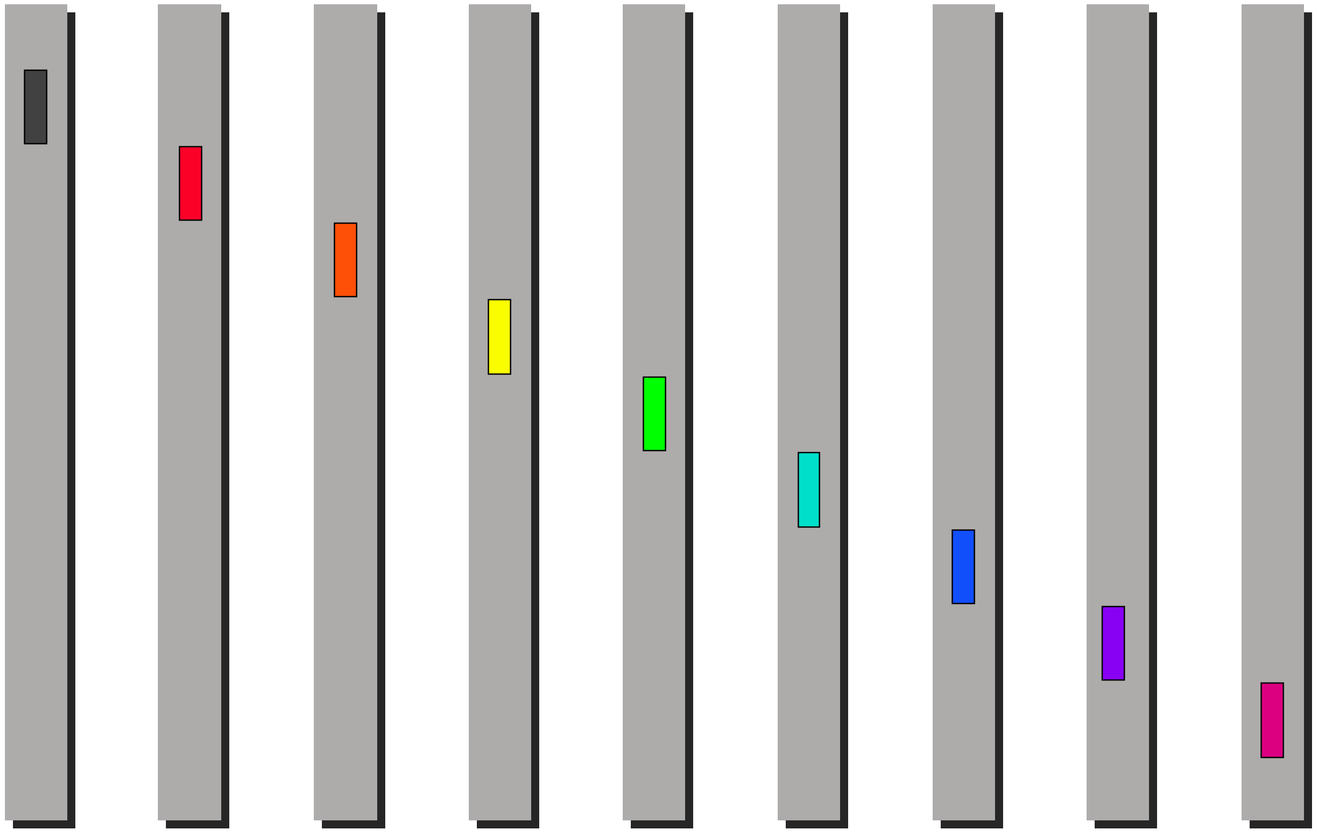
Large Message

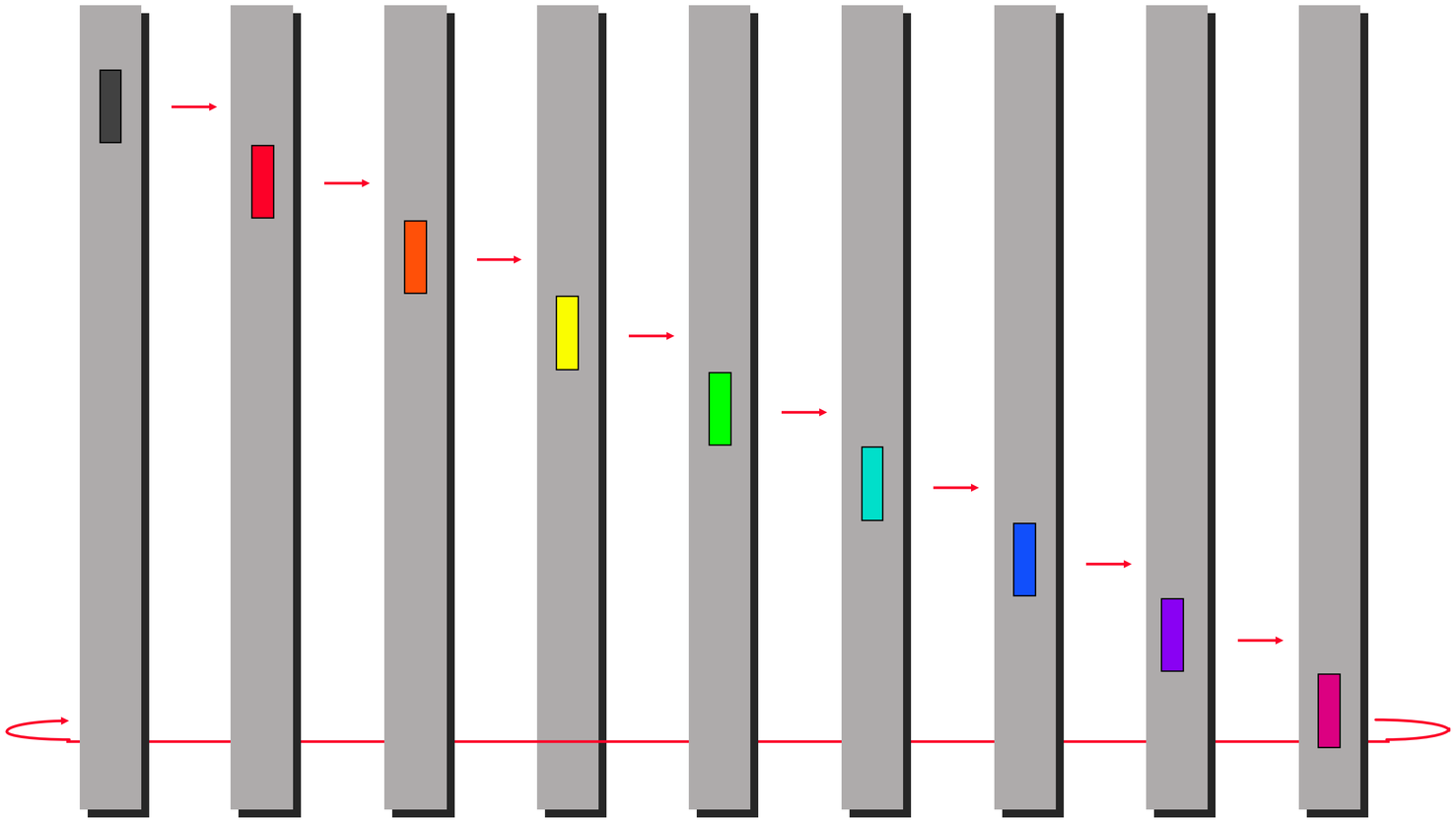
Communication Model: $\alpha + n\,\beta, \beta = \frac{1}{B}$

- The second term dominates – we want to minimize the second term
  - We want to utilize the bandwidth as much as possible

General principles

- Use all the links between every two nodes
- A logical ring can be embedded in a physical linear array with worm-hole routing, since the “wrap-around” message doesn’t conflict





- A logical ring can be embedded in a physical linear array with worm-hole routing, since the “wrap-around” message doesn’t conflict