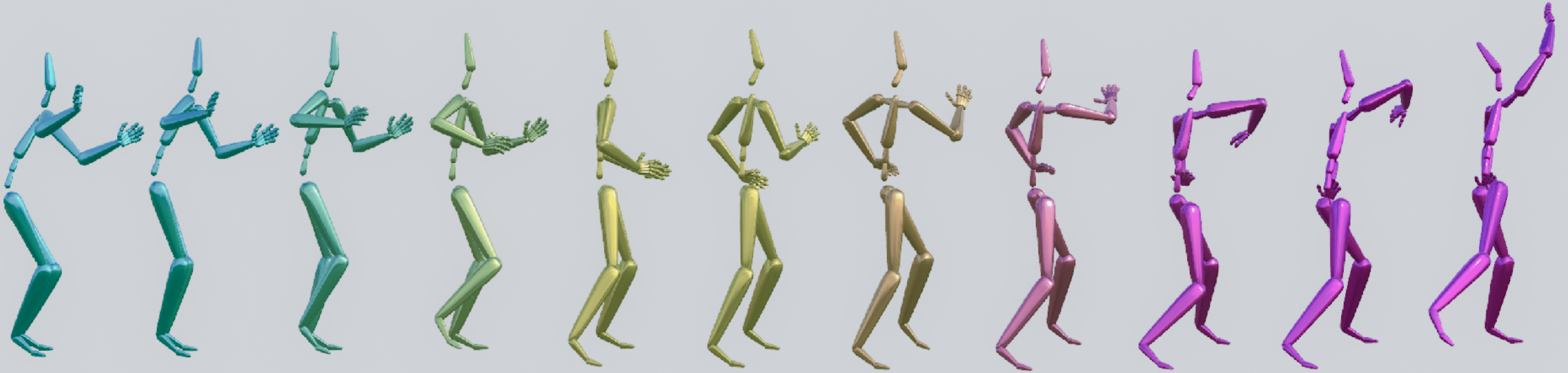
The Visual Computer (CGI 2019)

Precomputed Optimal One-Hop Motion Transition for Responsive Character Animation

Yuki Koyama & Masataka Goto

AIST



Target Problem

State Machine-Based Control for Interactive Apps

State machine-based control is widely used.

Each state is associated with a single motion clip (in the basic usage).

In response to run-time user input, the state is changed.

Triggered by user input
at an arbitrary frame

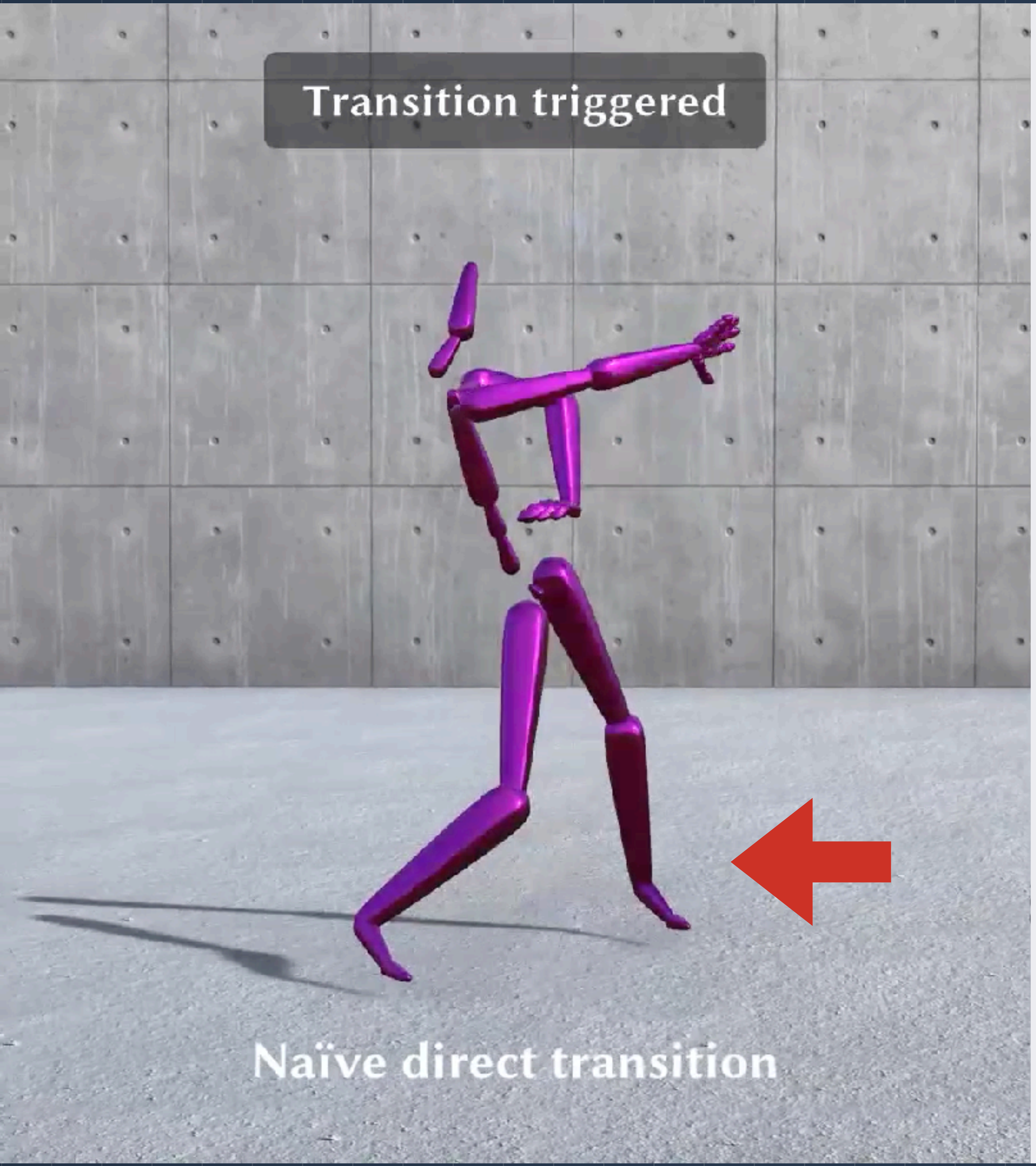
Transition

Source (**src**) motion

Destination (**dst**) motion

Goal: Achieve High-Quality Motion Transition

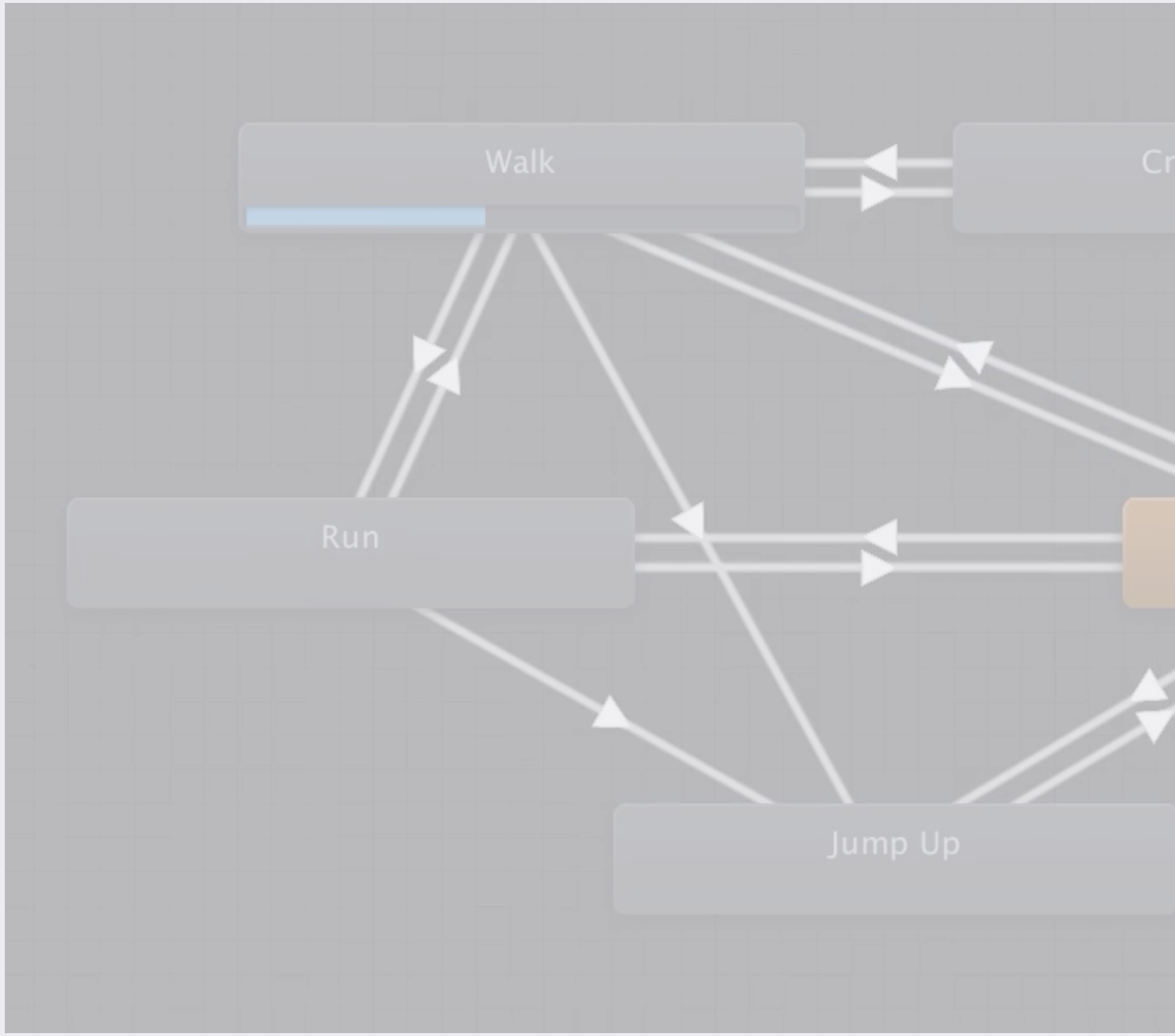
Direct Approach (Current Practice)



- **Direct approach:**
  - Begin transition **immediately** after receiving the trigger
  - **Directly** interpolate the **src** and **dst** motions
- **Problem:**
  - Visual artifacts (e.g., foot skate)

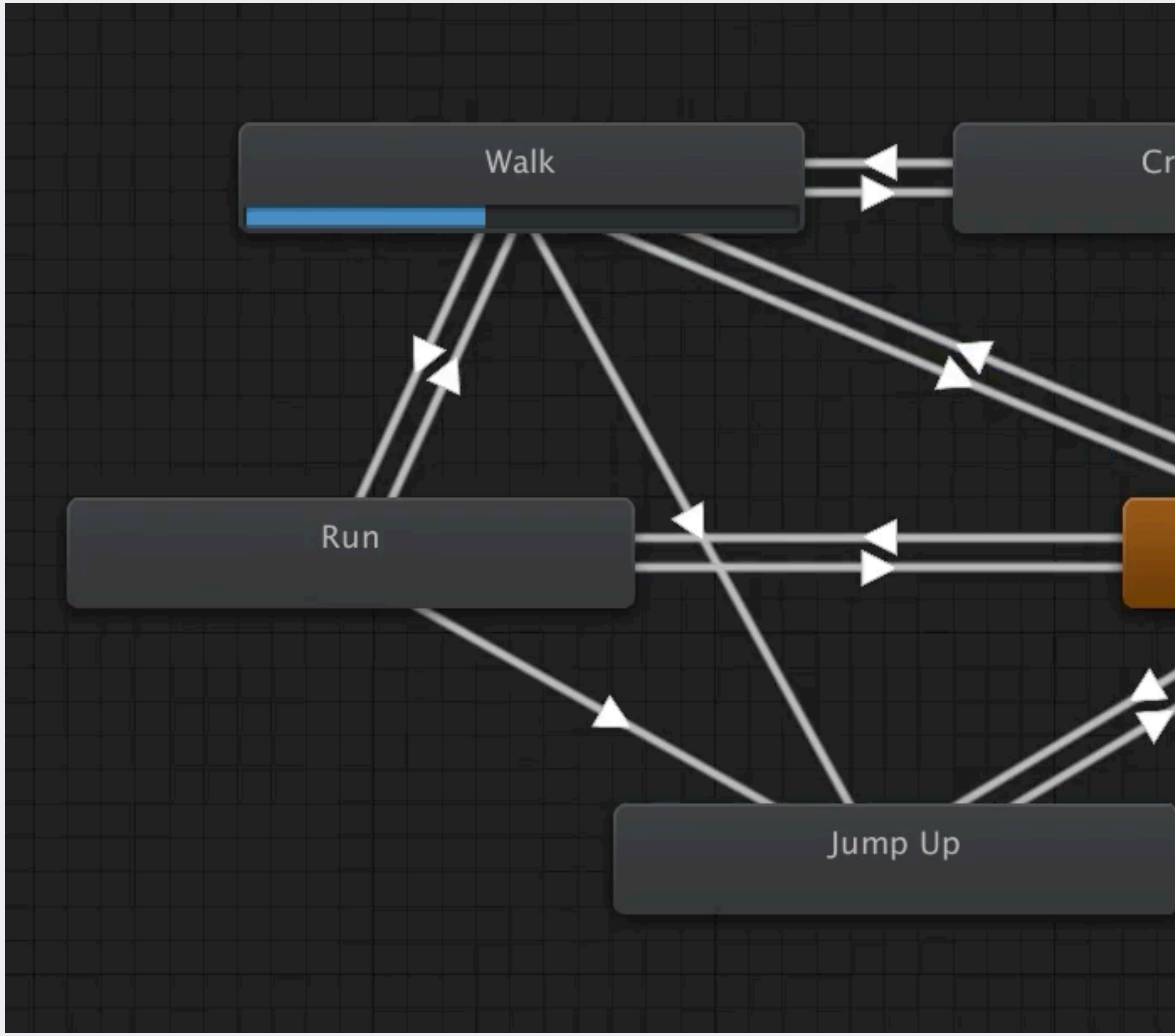
Requirements for a New Method

- **A new method should ...**
  - Be compatible with the **state machine**-based workflow
  - Run in interactive **real-time** apps
  - Finish the transition as **quickly** as possible



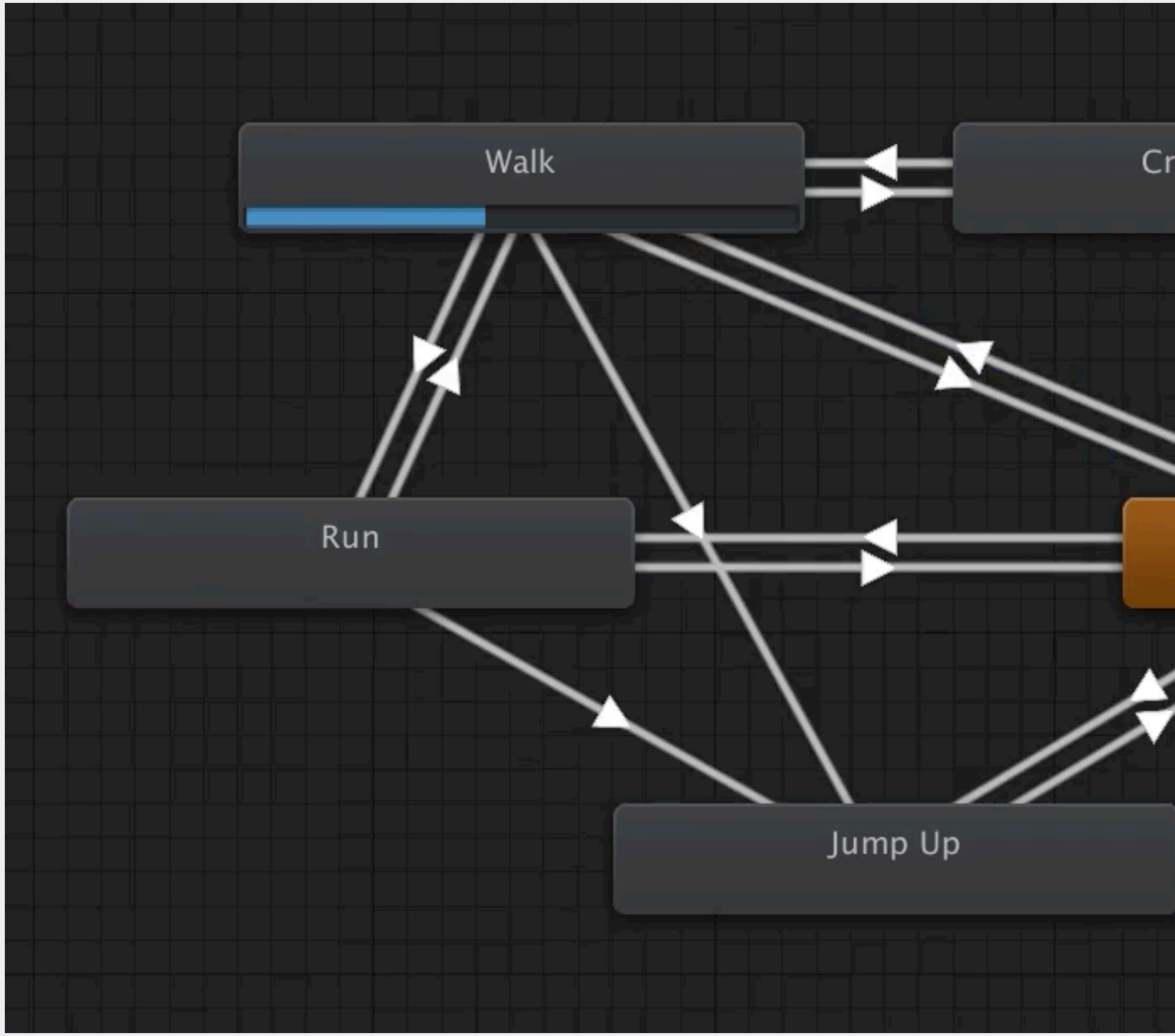
Requirements for a New Method

- **A new method should …**
  - Be compatible with the **state machine**-based workflow
  - Run in interactive **real-time** apps
  - Finish the transition as **quickly** as possible



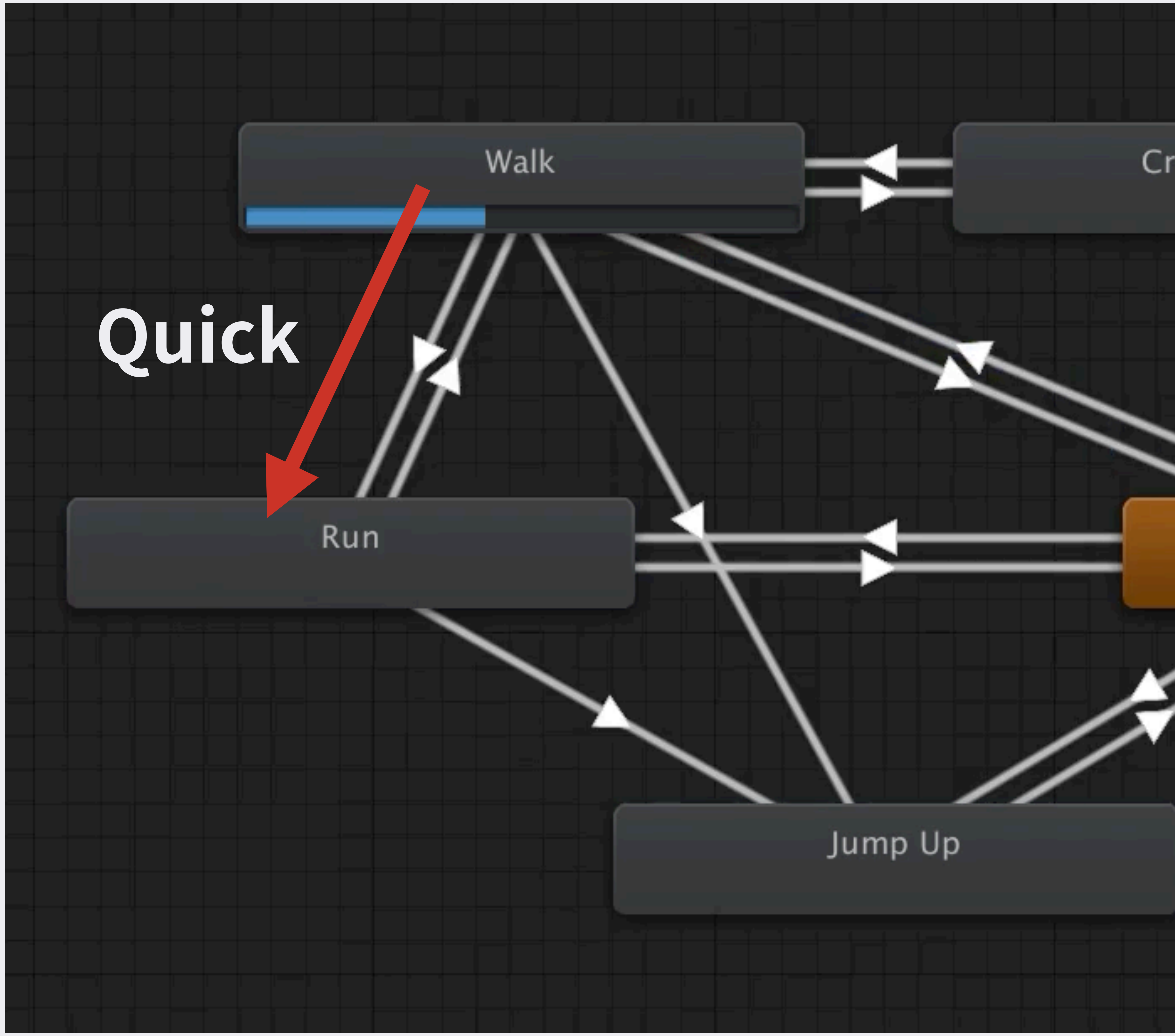
Requirements for a New Method

- **A new method should …**
  - Be compatible with the **state machine**-based workflow
  - Run in interactive **real-time** apps
  - Finish the transition as **quickly** as possible



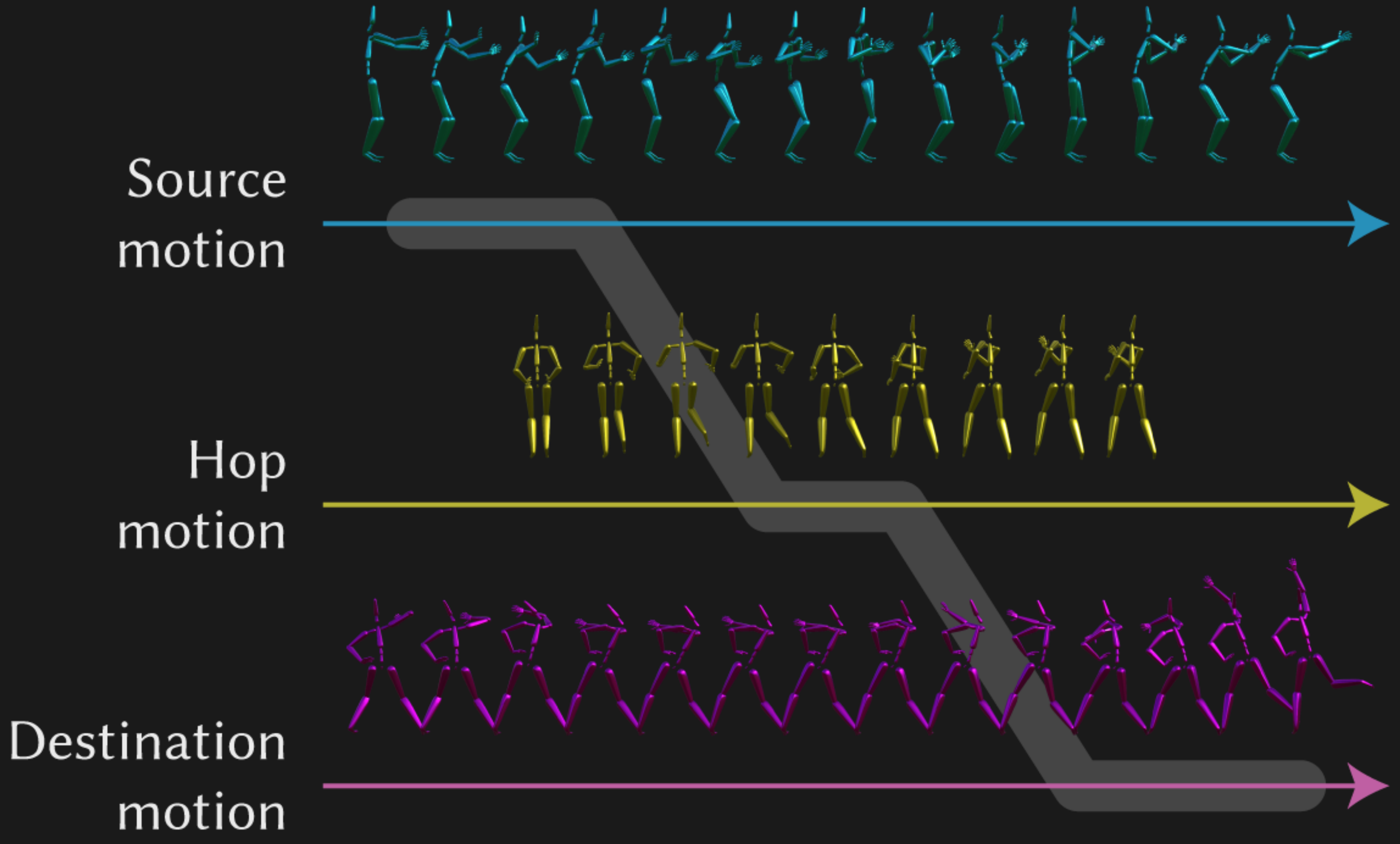
Requirements for a New Method

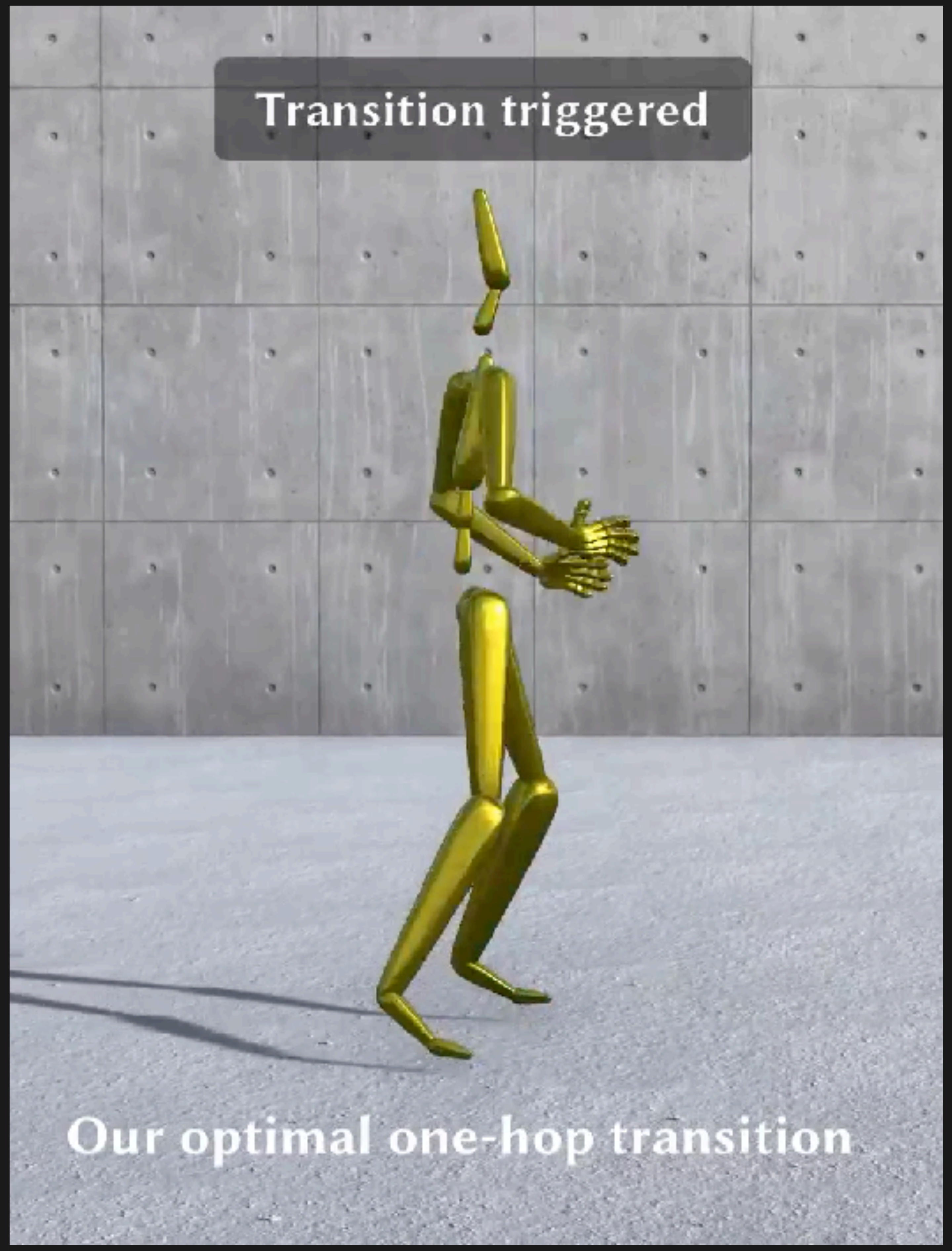
- **A new method should ...**
  - Be compatible with the **state machine**-based workflow
  - Run in interactive **real-time** apps
  - Finish the transition as **quickly** as possible



Overview of Our Method

Our Approach: “One-Hop” Motion Transition






Transition triggered
Transition triggered
Naïve direct transition
Our optimal one-hop transition

Overview of This Work

- **Technical Challenges:**
  1. How should we **choose a hop motion and arrange it**?
  2. How can we enable this in **real-time** apps?

- **Our Solutions:**
  1. Formulating a **search problem**
  2. Solving the searches in **precomputation**

Overview of This Work

- **Technical Challenges:**

1. How should we **choose a hop motion and arrange it**?
2. How can we enable this in **real-time** apps?

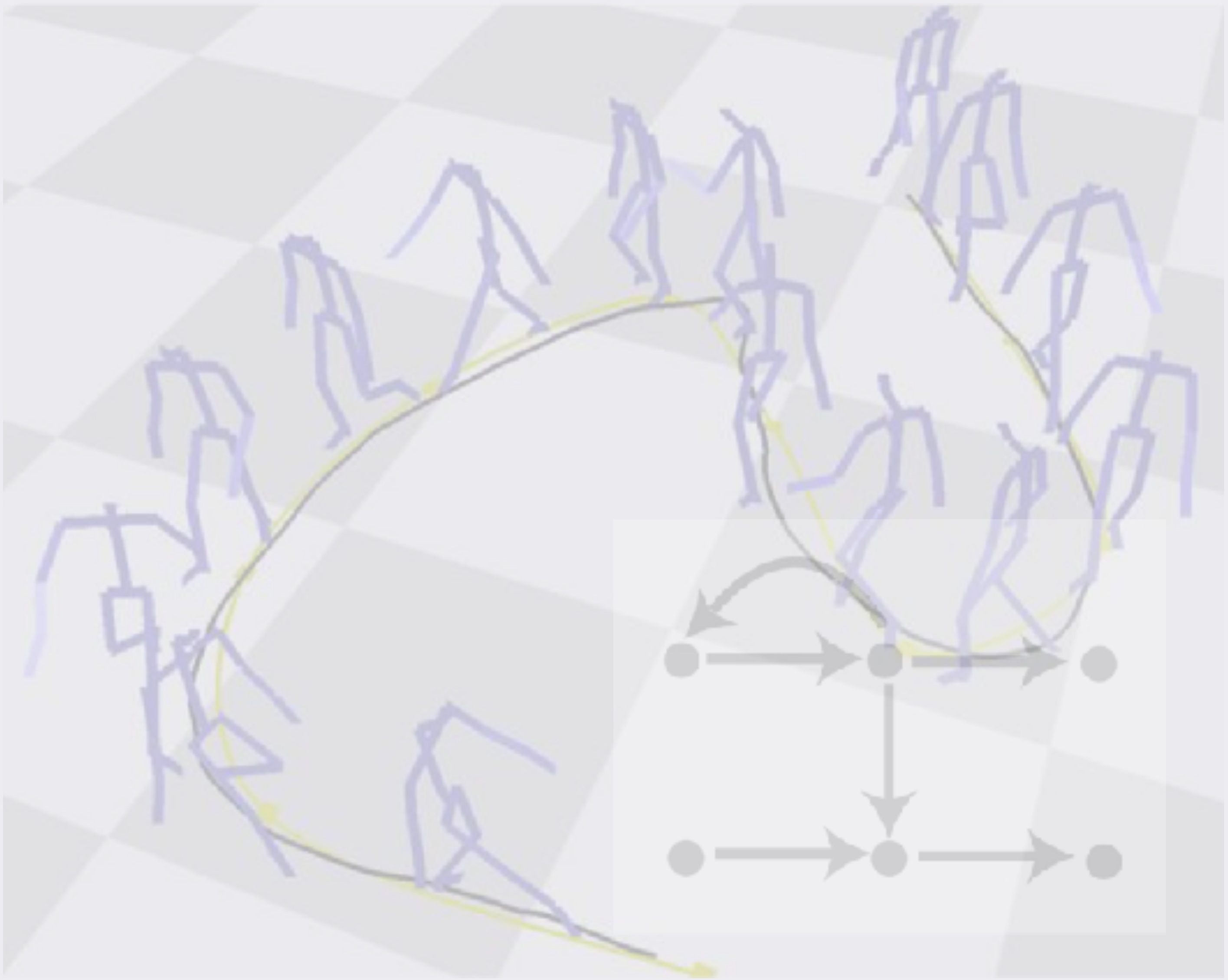
- **Our Solutions:**

1. Formulating a **search problem**
2. Solving the searches in **precomputation**

Related Work

Related Frameworks

Motion graphs

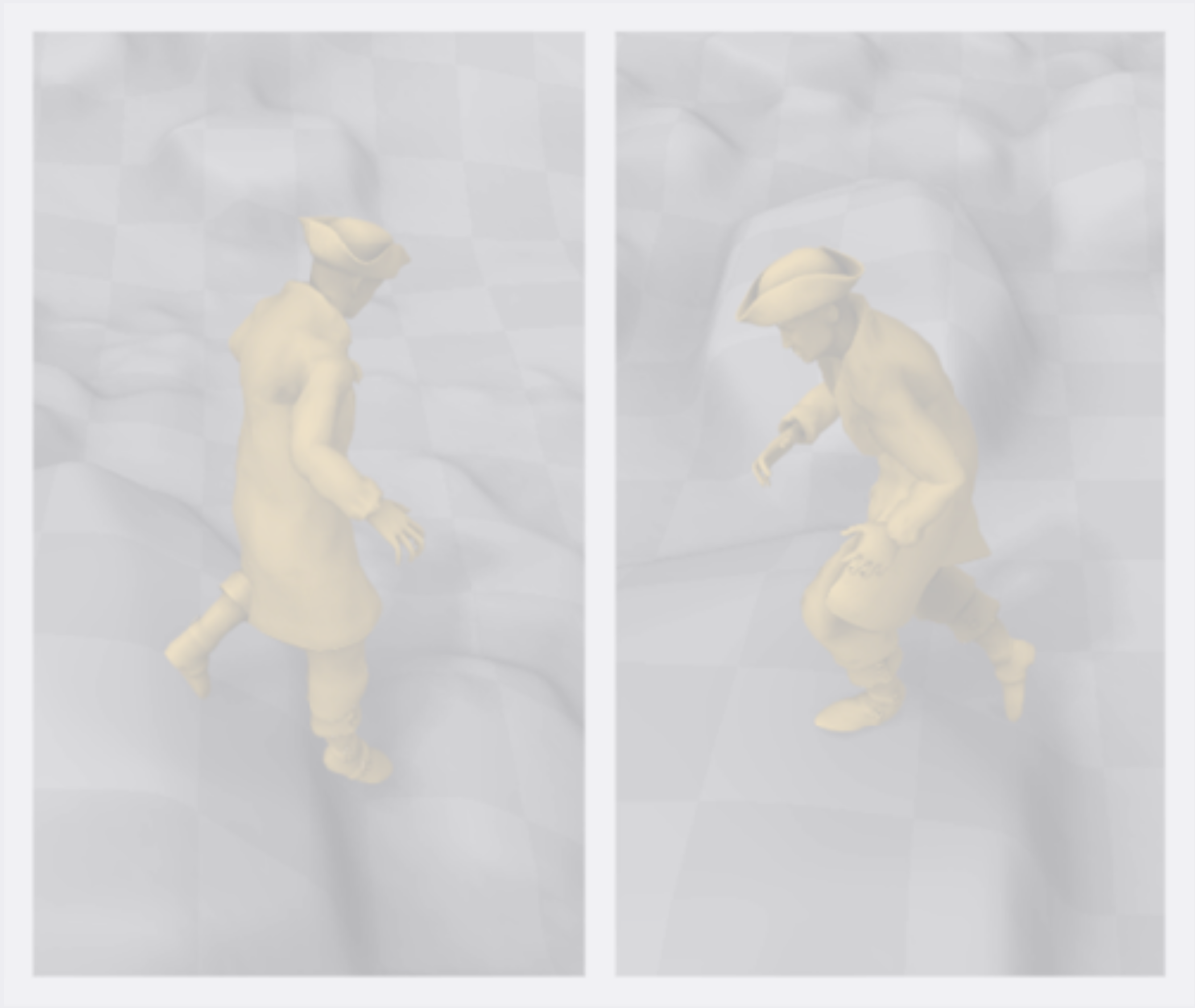


[Kovar+02, …]

Motion matching

[Zadziuk16, …]

Statistical models

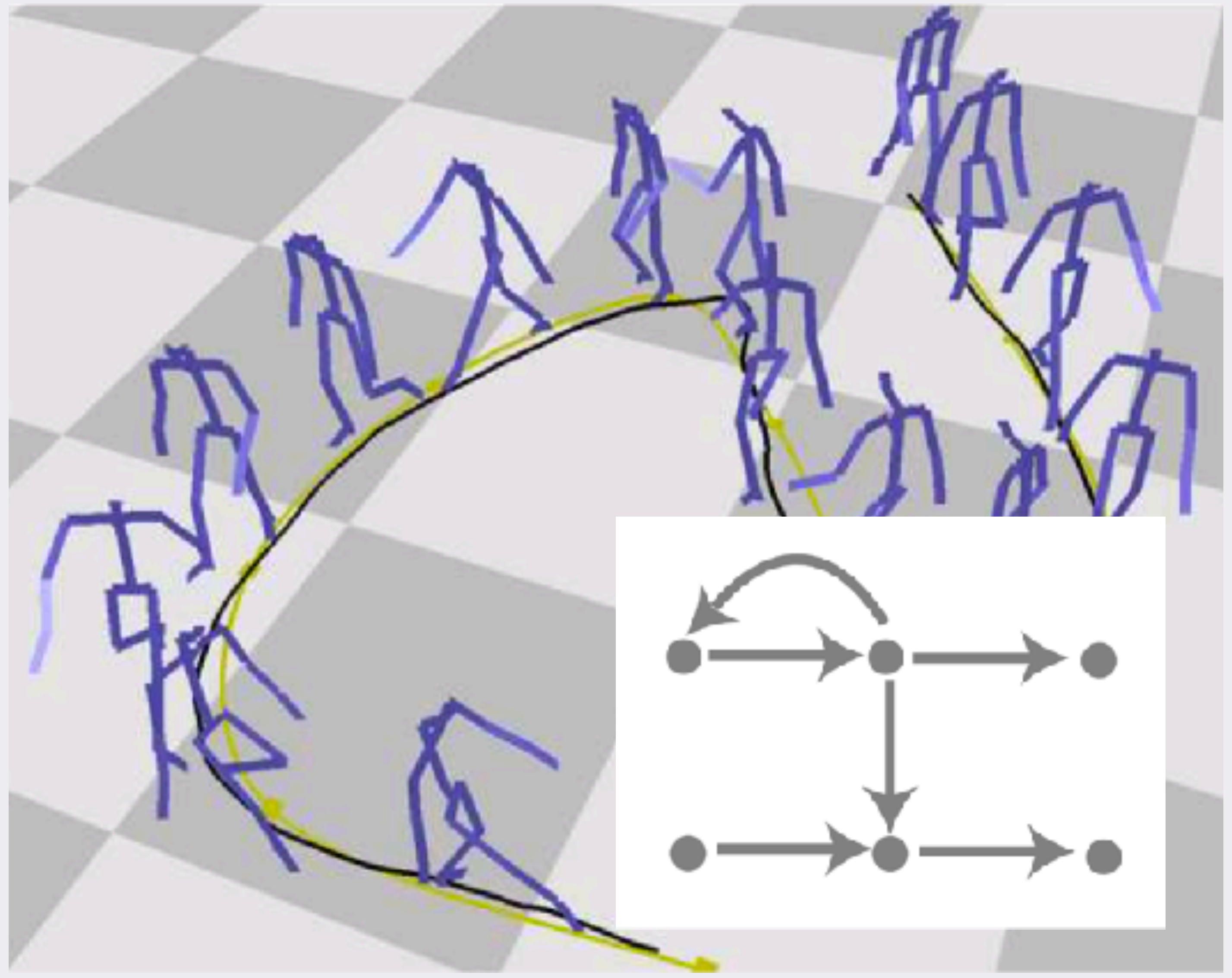


[Holden+17, …]

➡ **These are incompatible with the state machine-based workflow**

Related Frameworks

Motion graphs

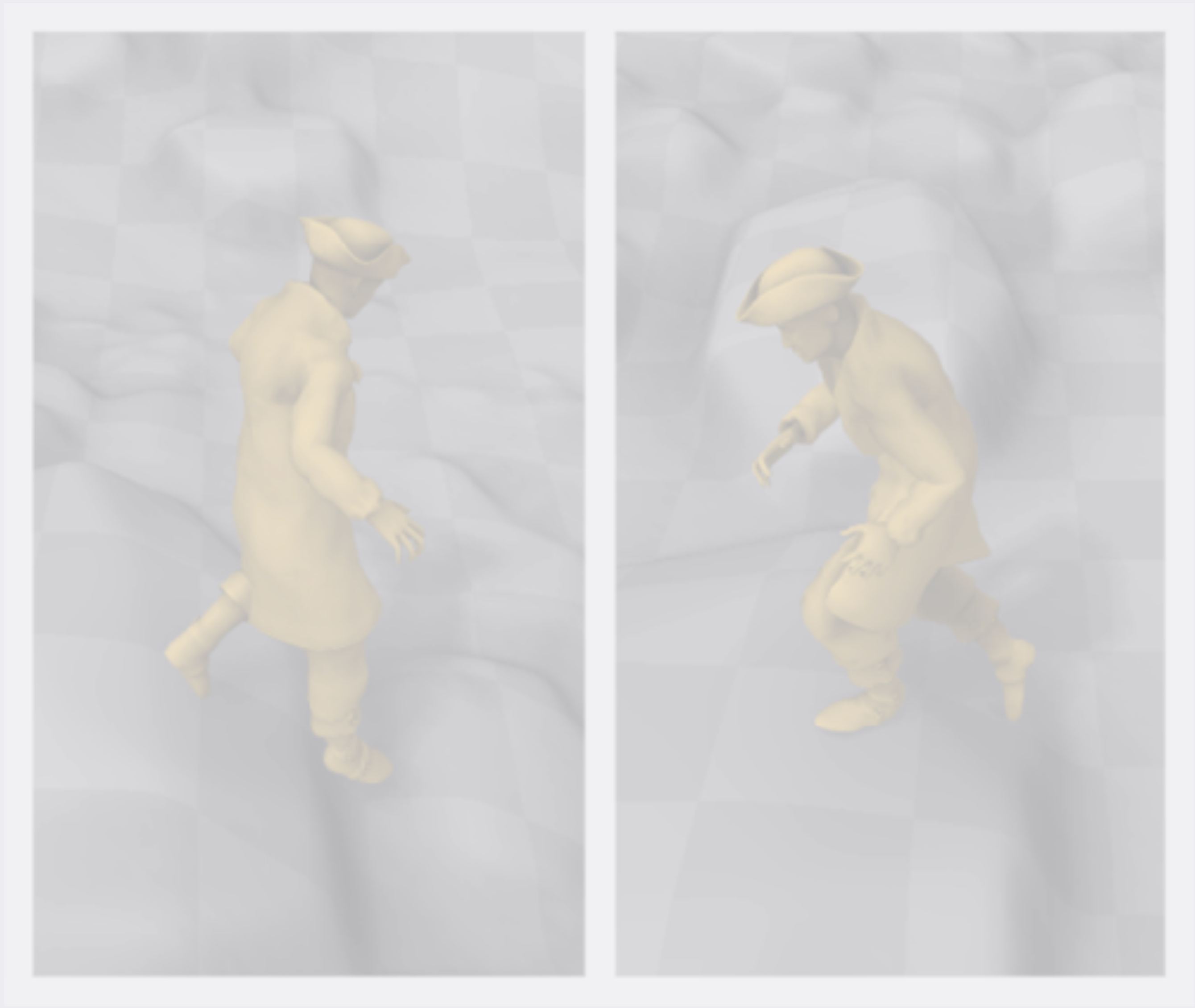


[Kovar+02, …]

Motion matching

[Zadziuk16, …]

Statistical models

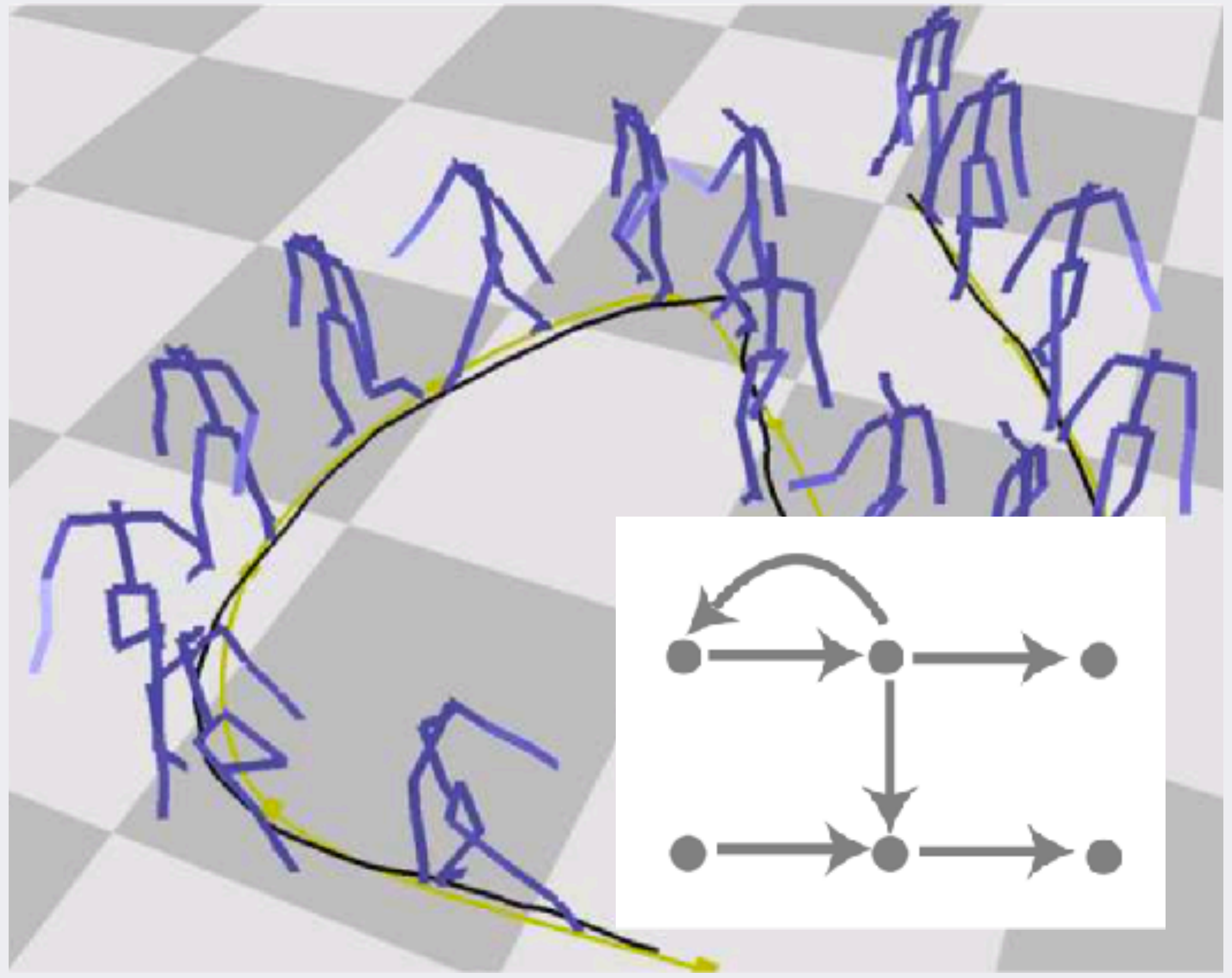


[Holden+17, …]

➡ **These are incompatible with the state machine-based workflow**

Related Frameworks

Motion graphs

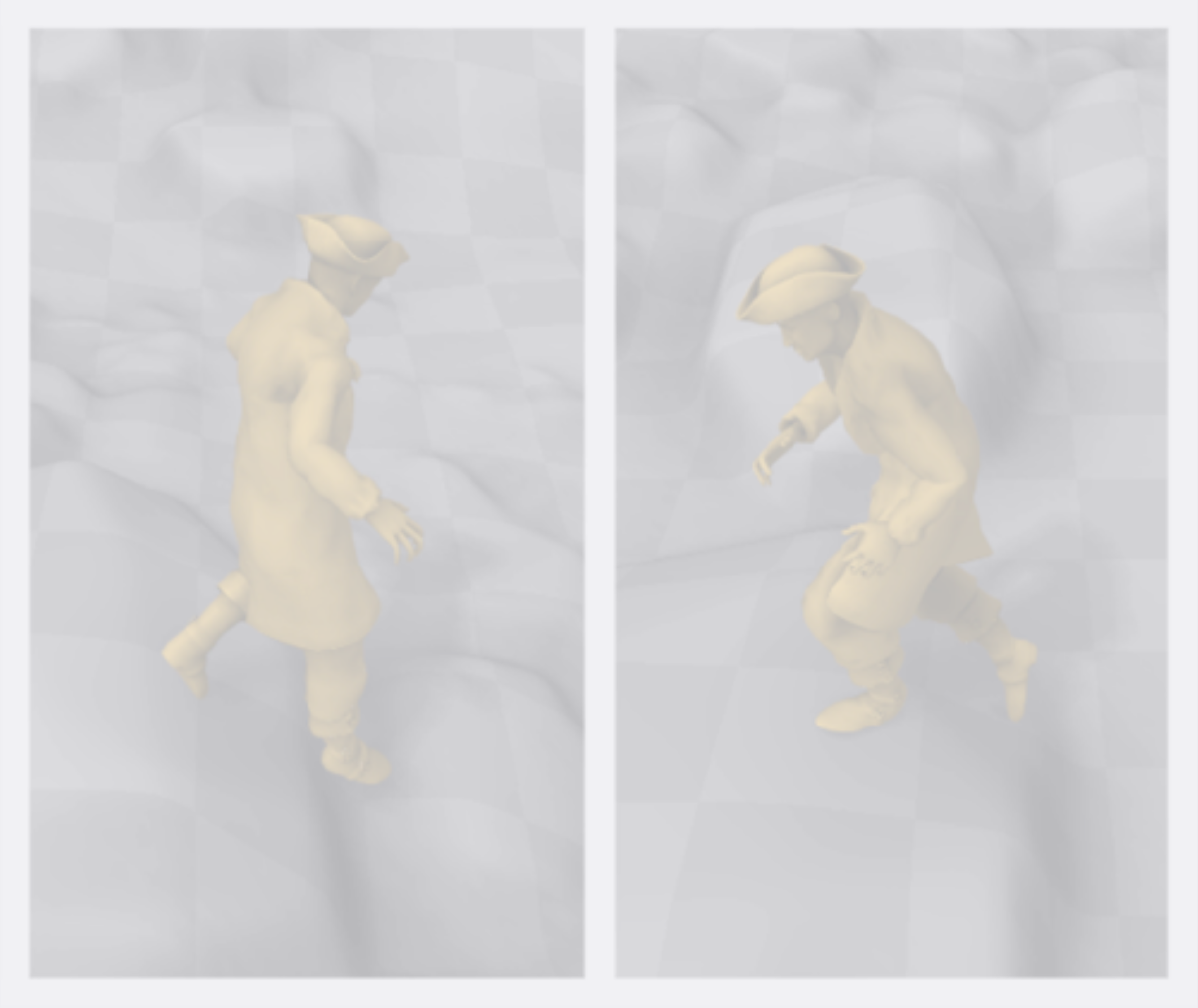


[Kovar+02, …]

Motion matching

[Zadziuk16, …]

Statistical models

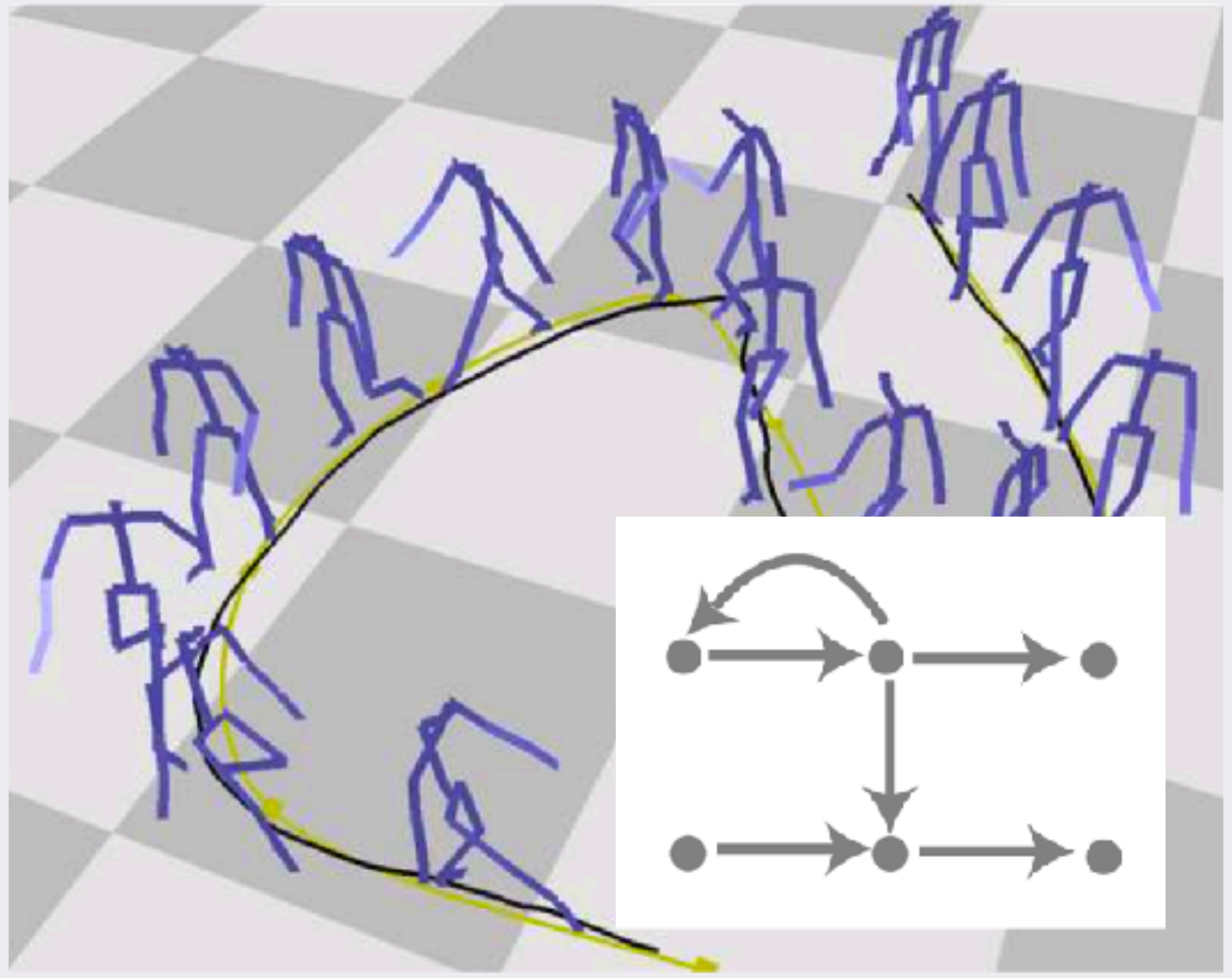


[Holden+17, …]

➡ **These are incompatible with the state machine-based workflow**

Related Frameworks

Motion graphs

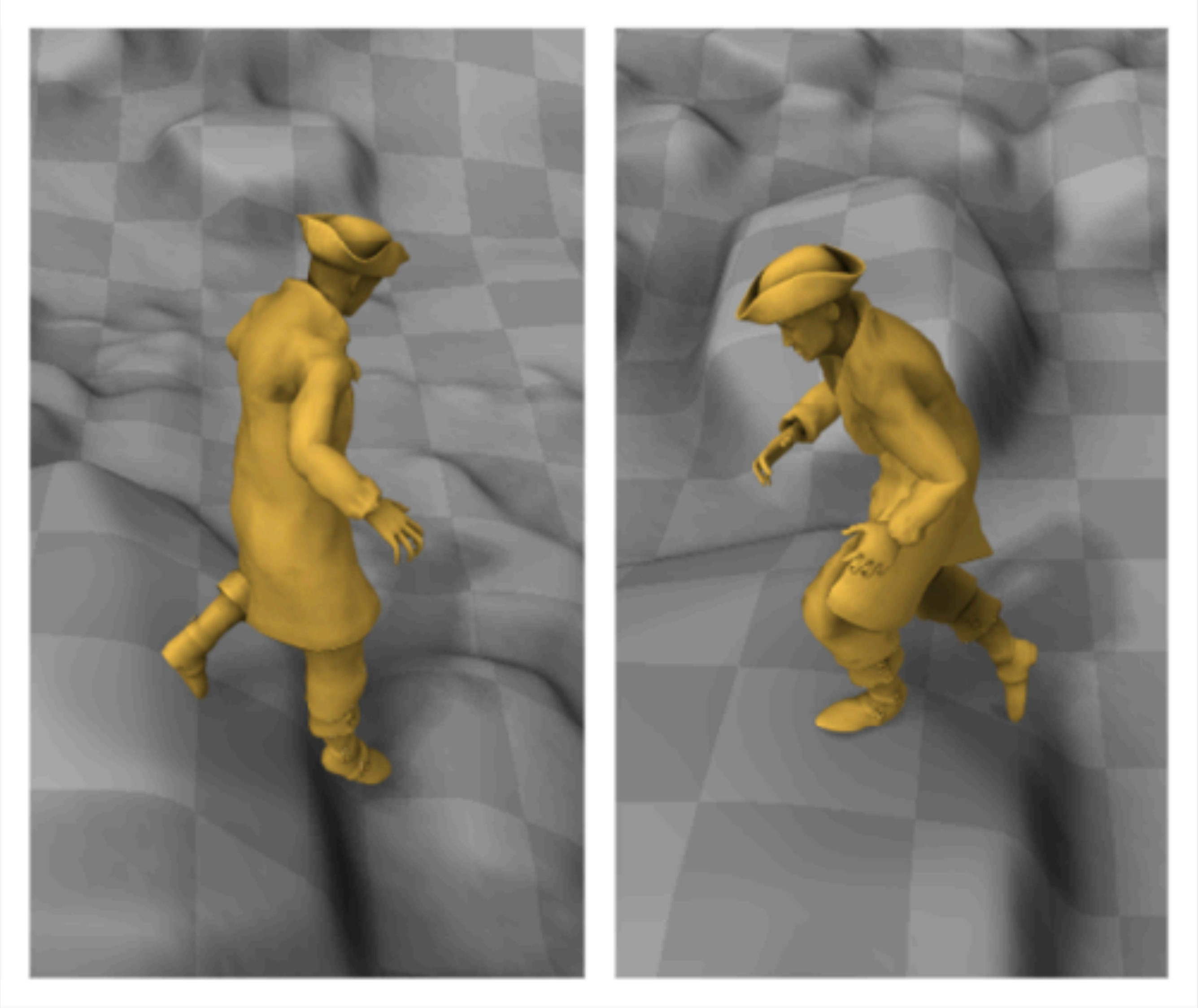


[Kovar+02, …]

Motion matching

[Zadziuk16, …]

Statistical models

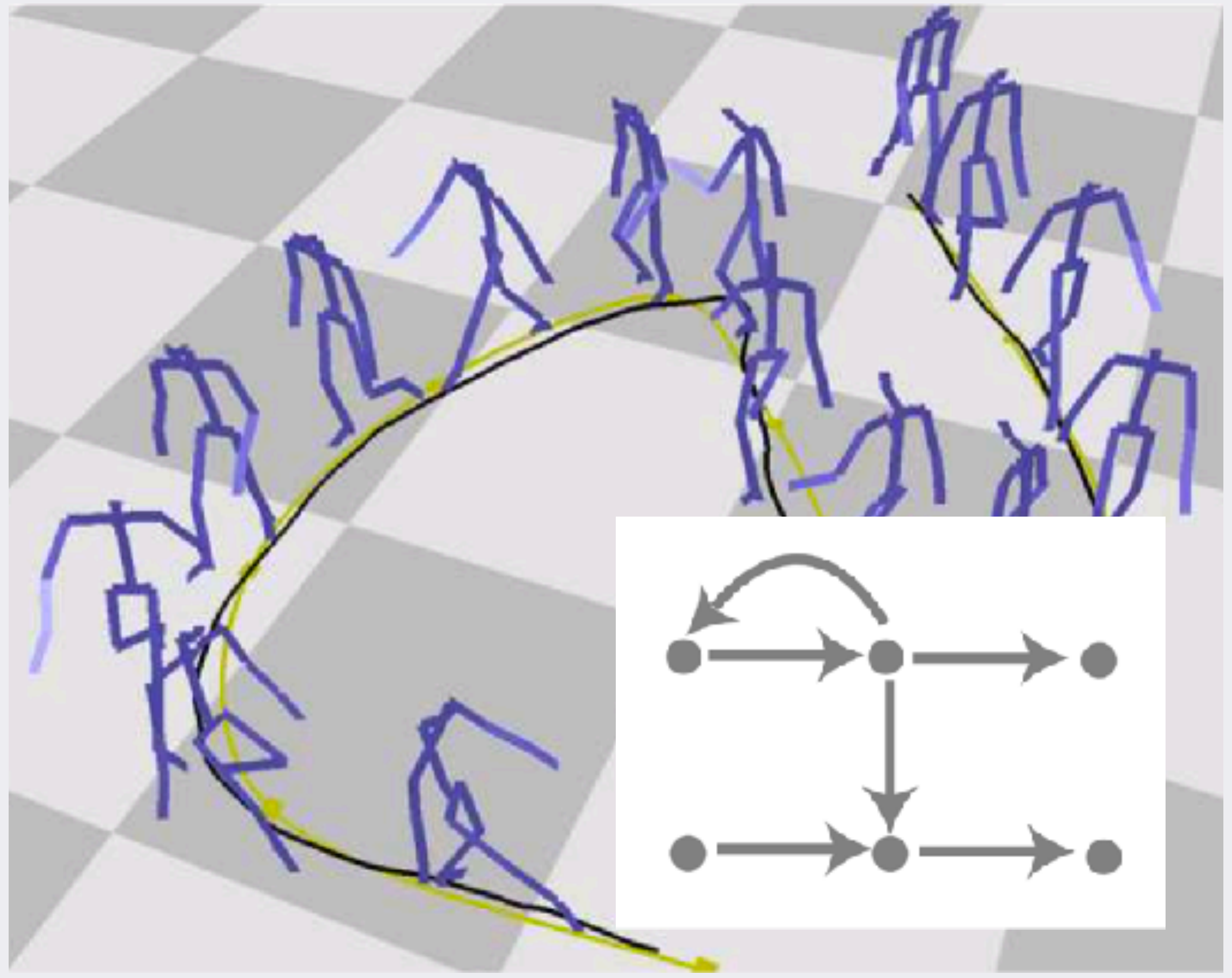


[Holden+17, …]

➡ **These are incompatible with the state machine-based workflow**

Related Frameworks

Motion graphs

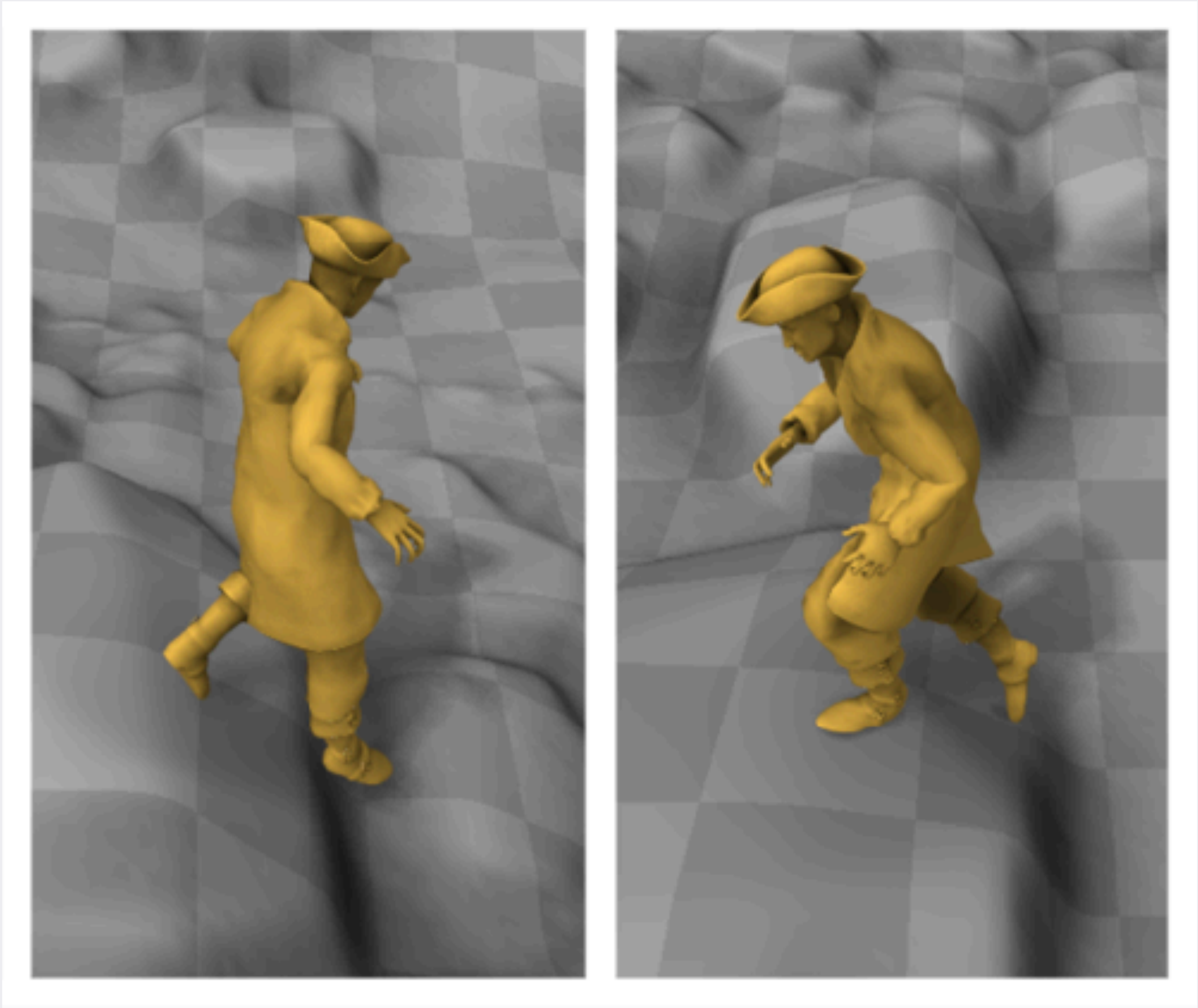


[Kovar+02, …]

Motion matching

[Zadziuk16, …]

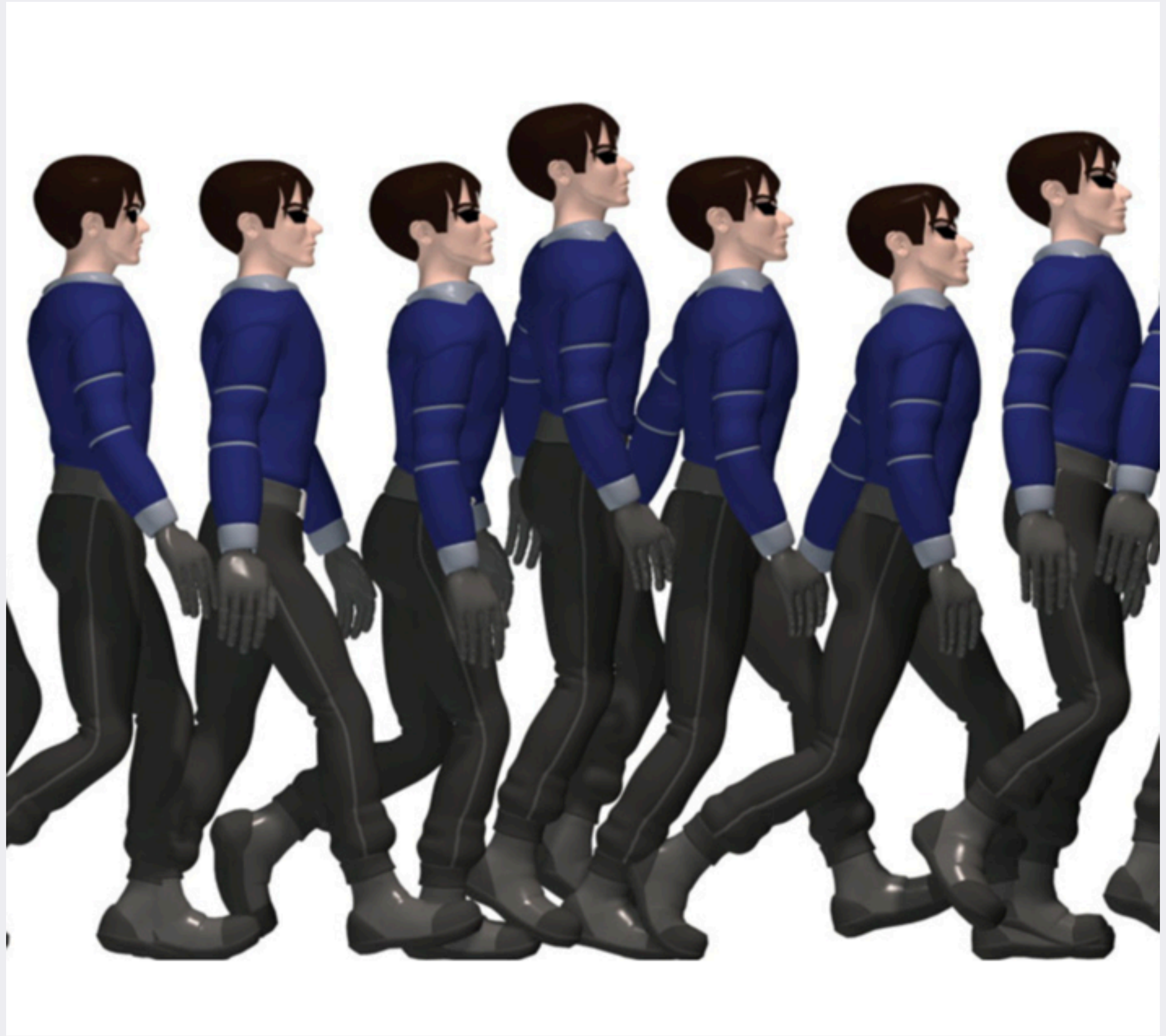
Statistical models



[Holden+17, …]

➡ **These are incompatible with the state machine-based workflow**

State Machine-Compatible Approach [Ikemoto+07]

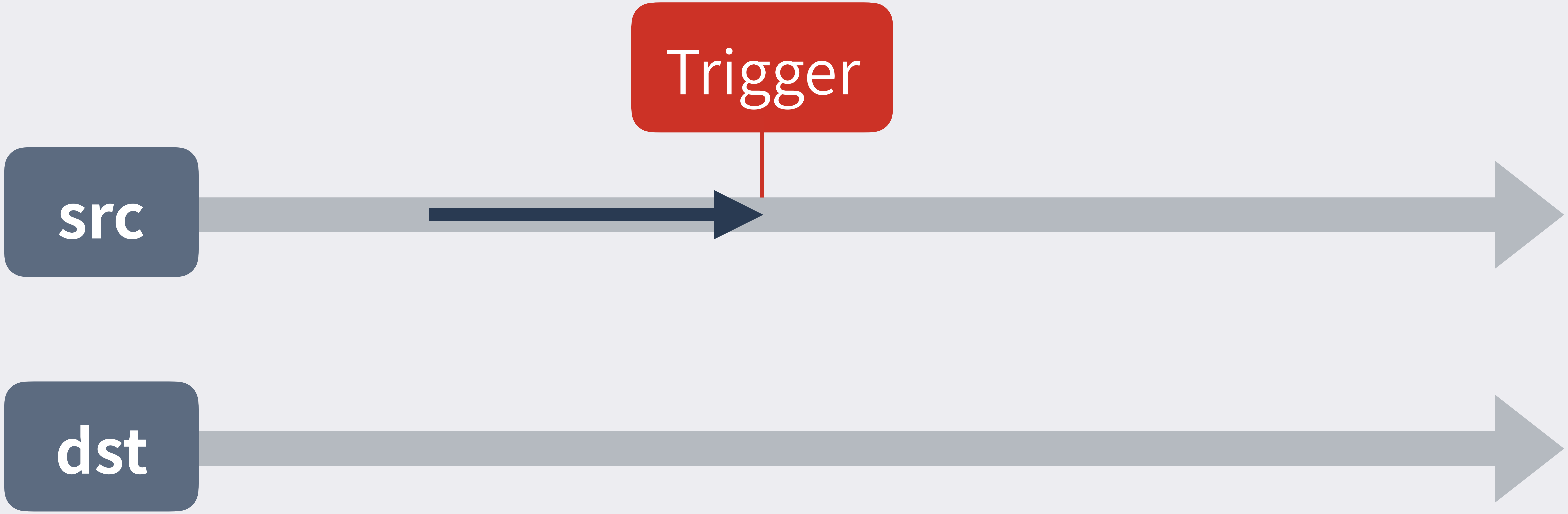


[Ikemoto+07]

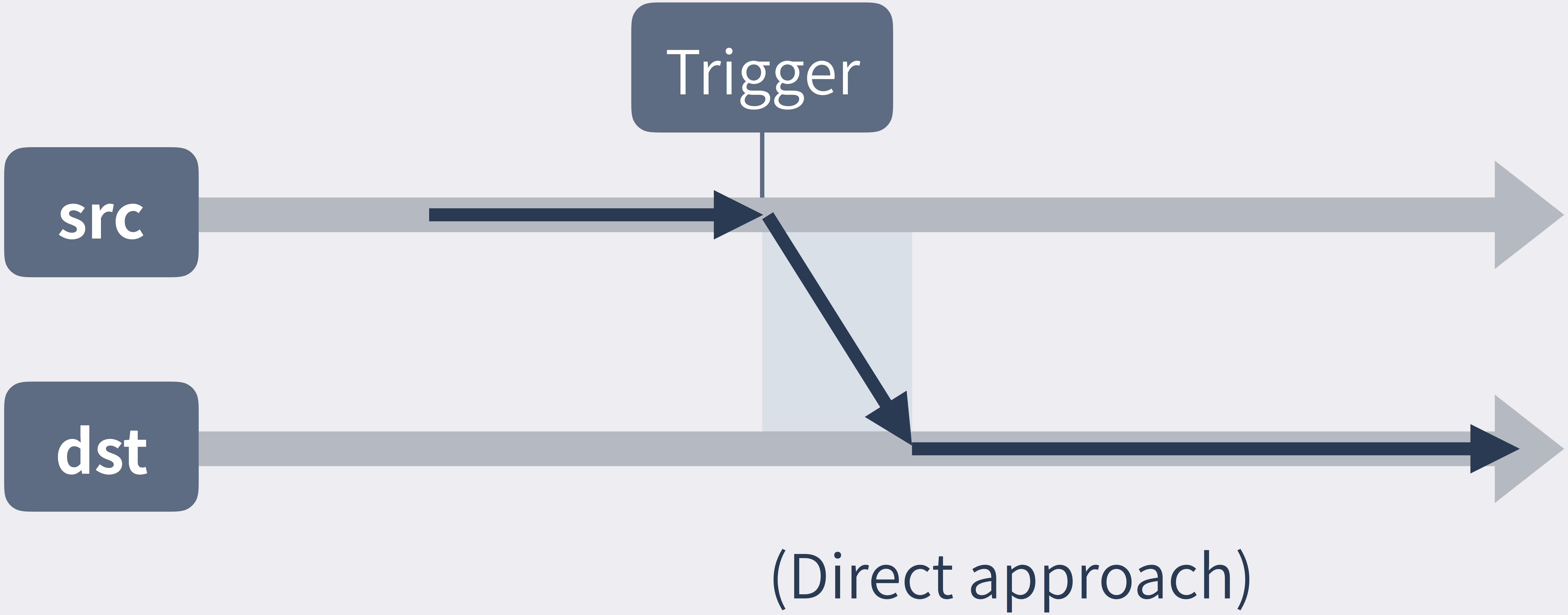
- **Similar to ours:**
  - They insert “hop” (intermediate) motions to achieve natural transitions
- **Key difference:**
  - We consider not only **naturalness** but also **quickness** to finish the overall transition (explained later)

1. Formulating a Search Problem

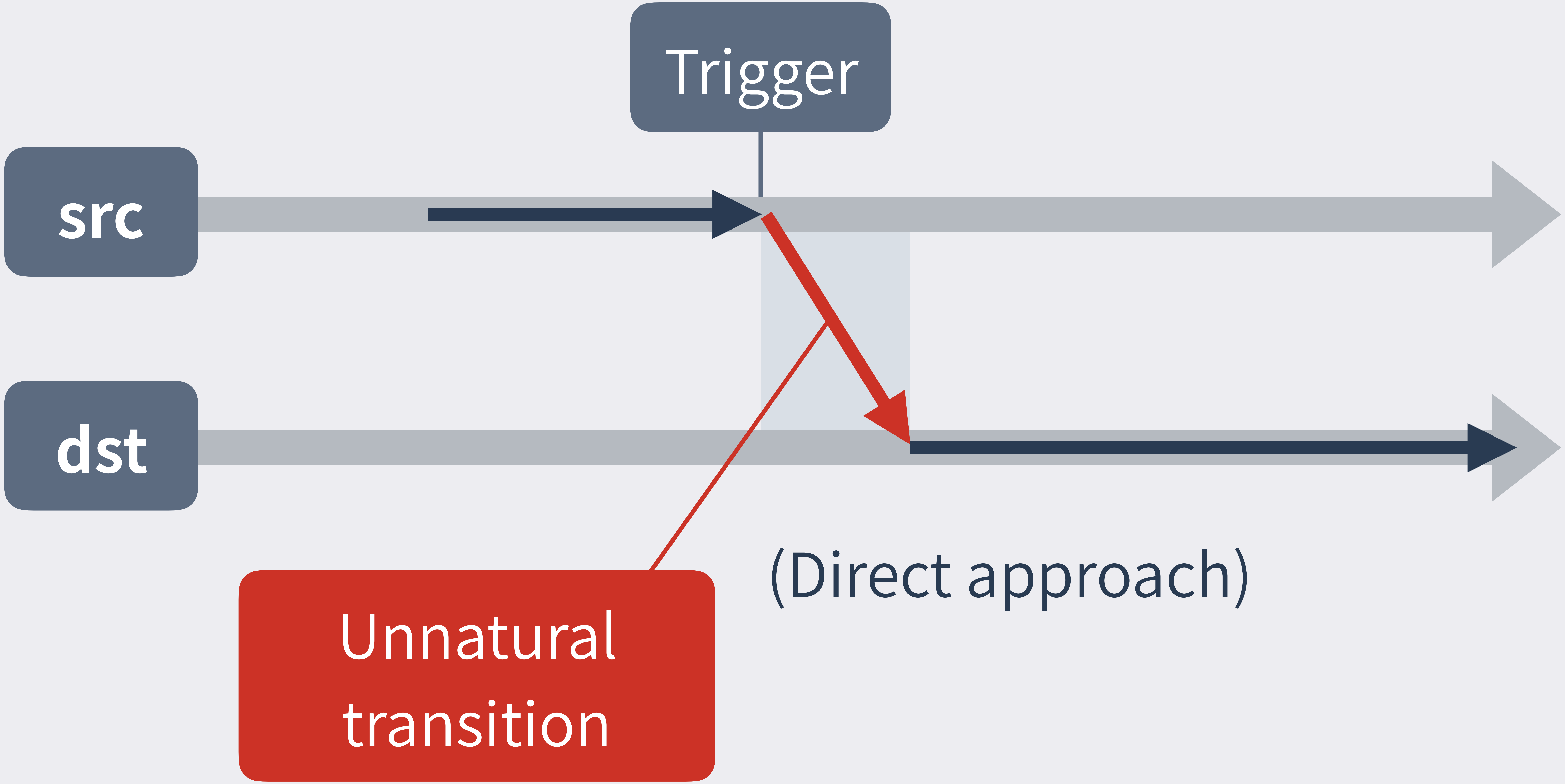
One-Hop Transition Setting



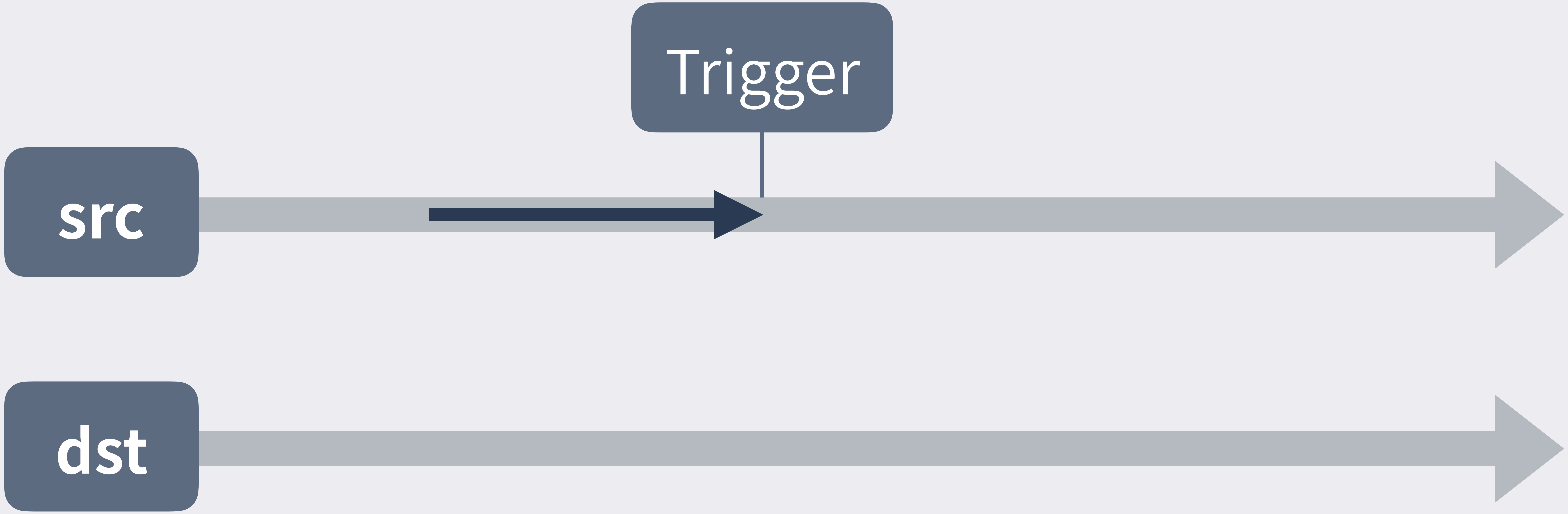
One-Hop Transition Setting



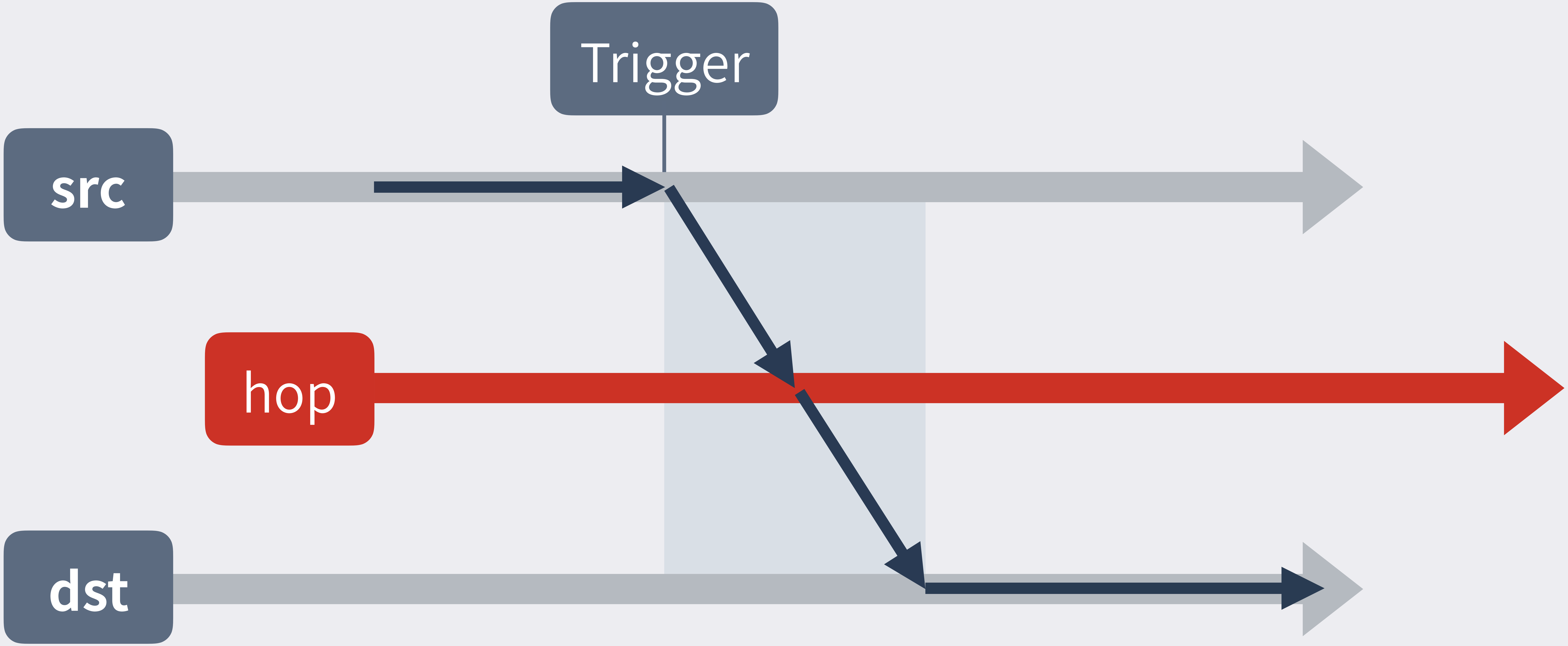
One-Hop Transition Setting



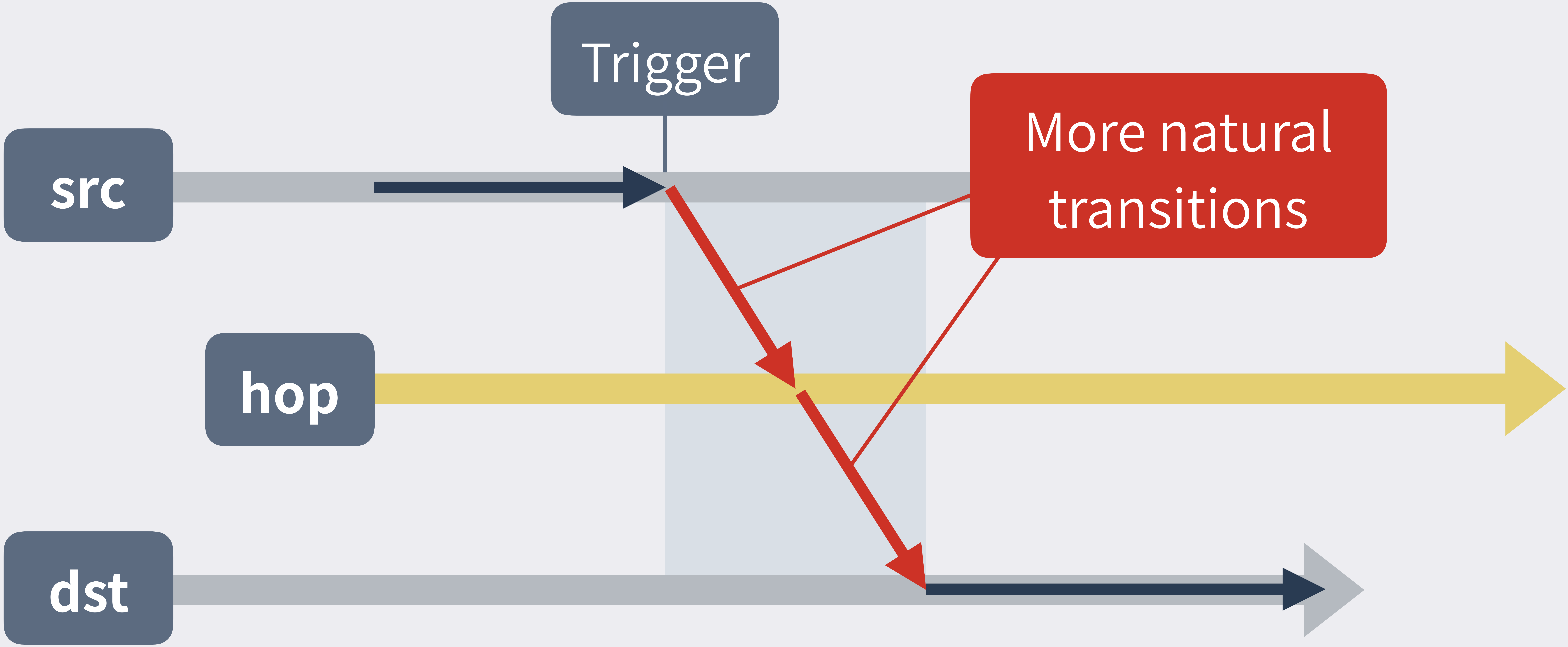
One-Hop Transition Setting



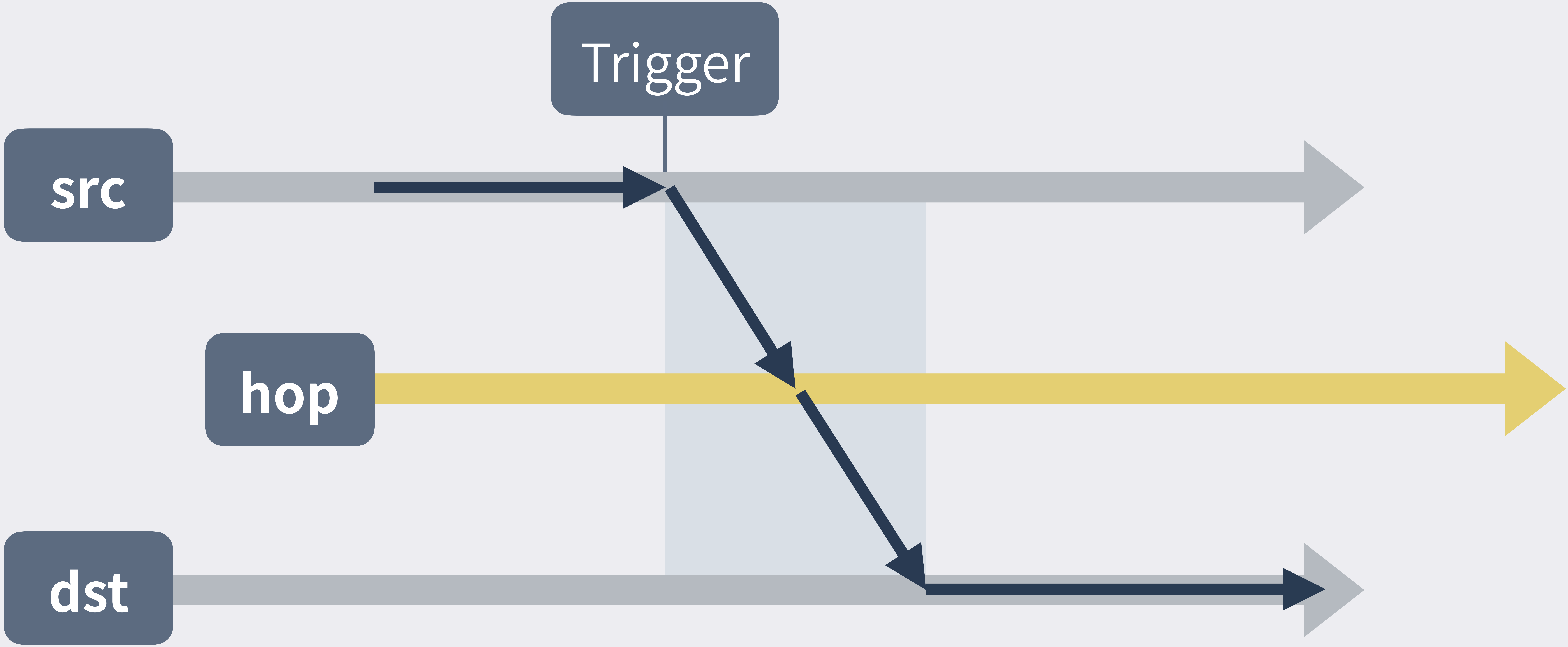
One-Hop Transition Setting



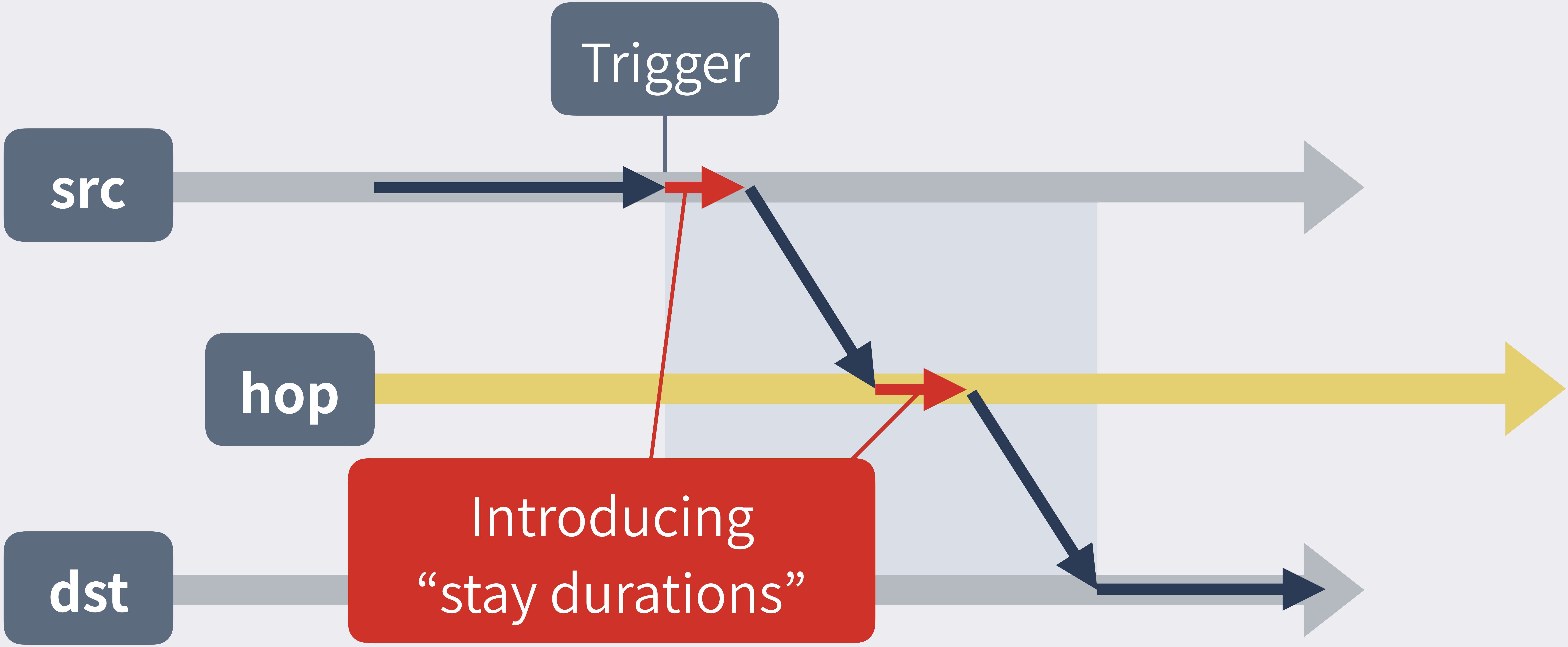
One-Hop Transition Setting



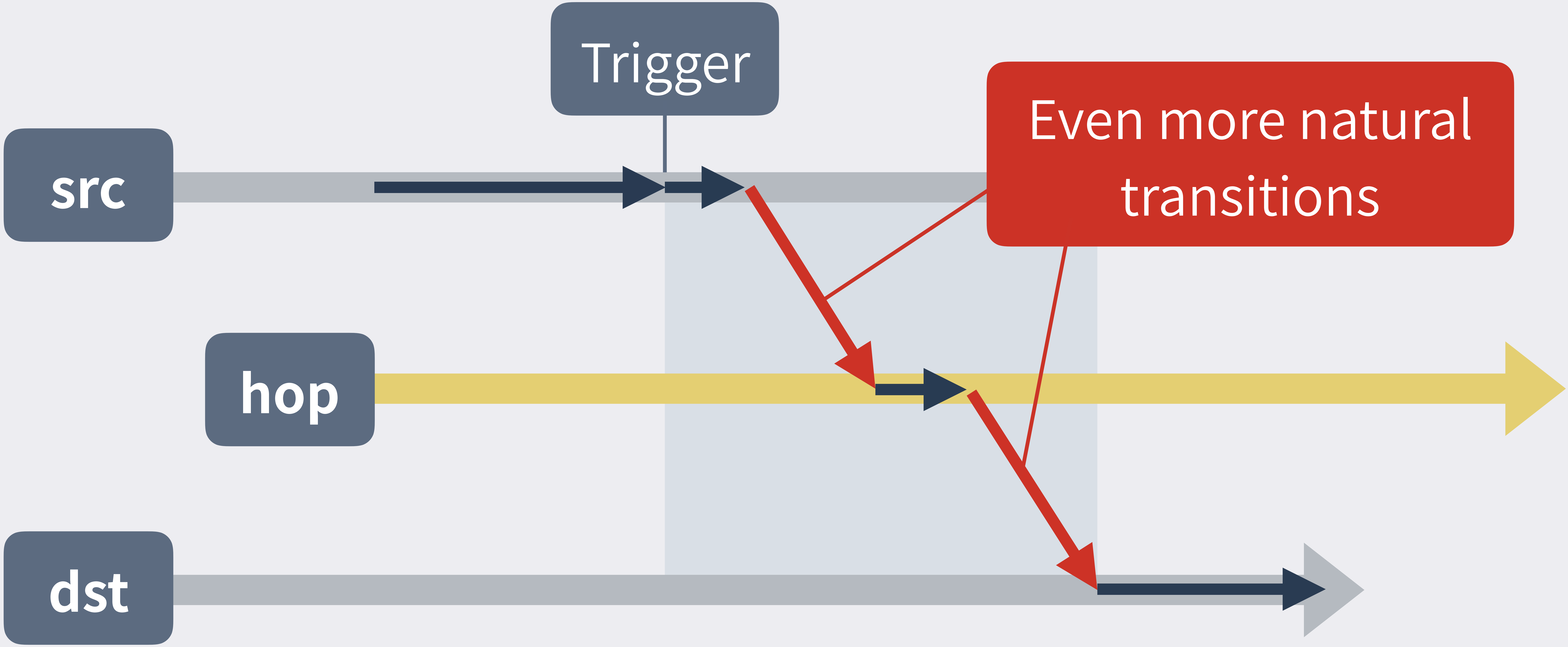
One-Hop Transition Setting



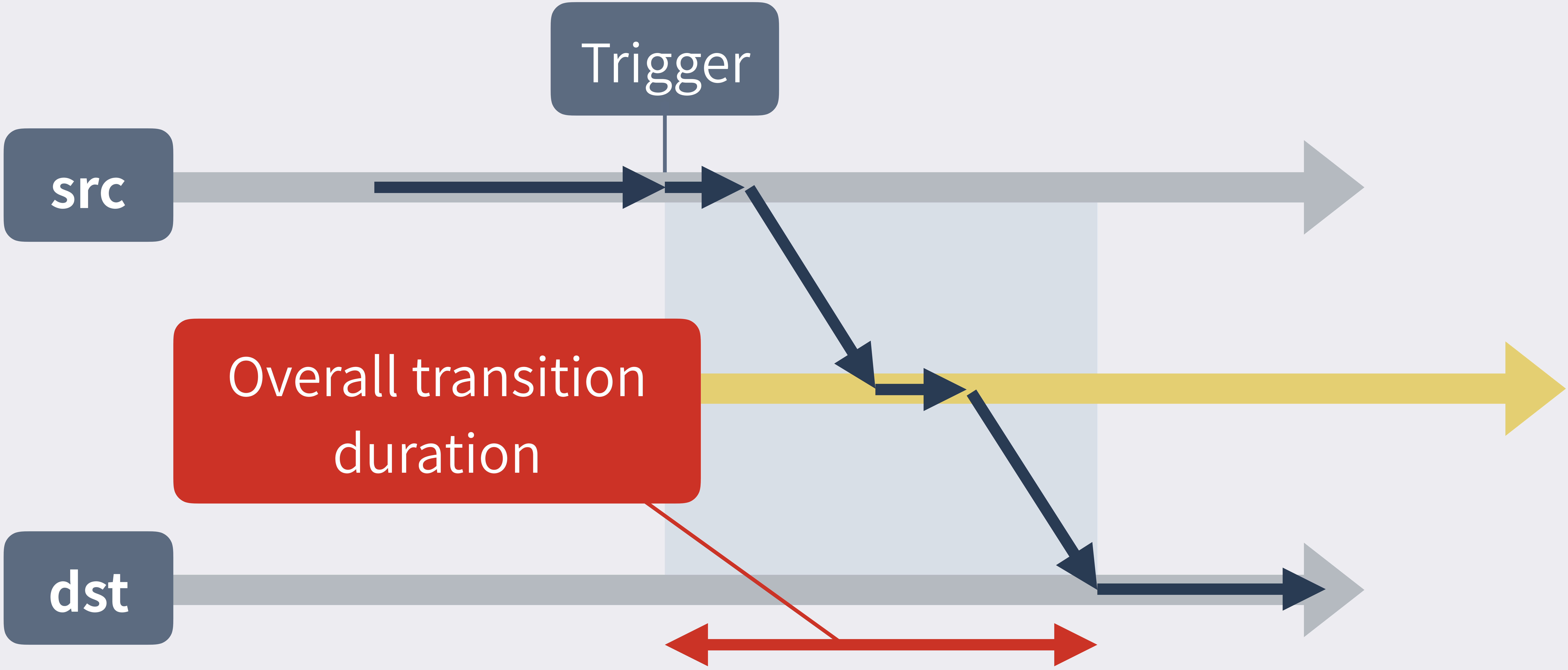
One-Hop Transition Setting



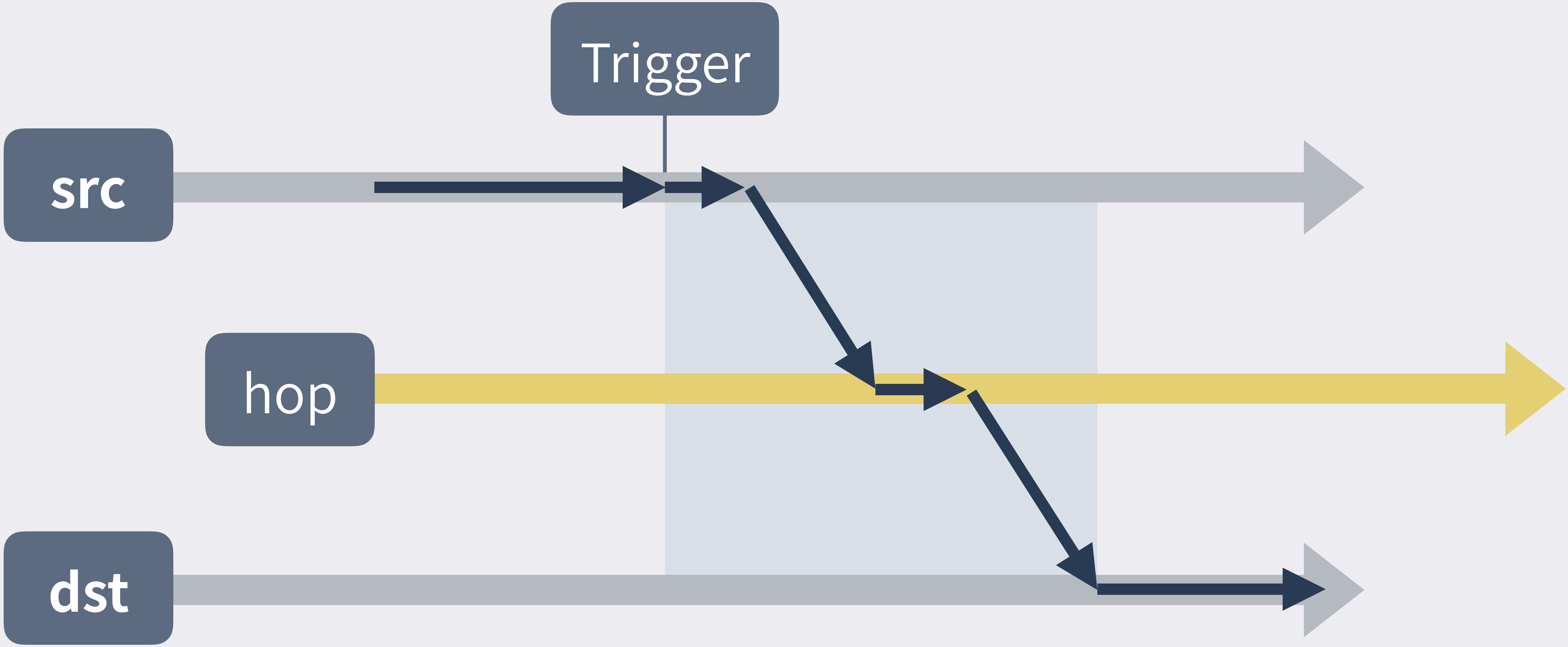
One-Hop Transition Setting



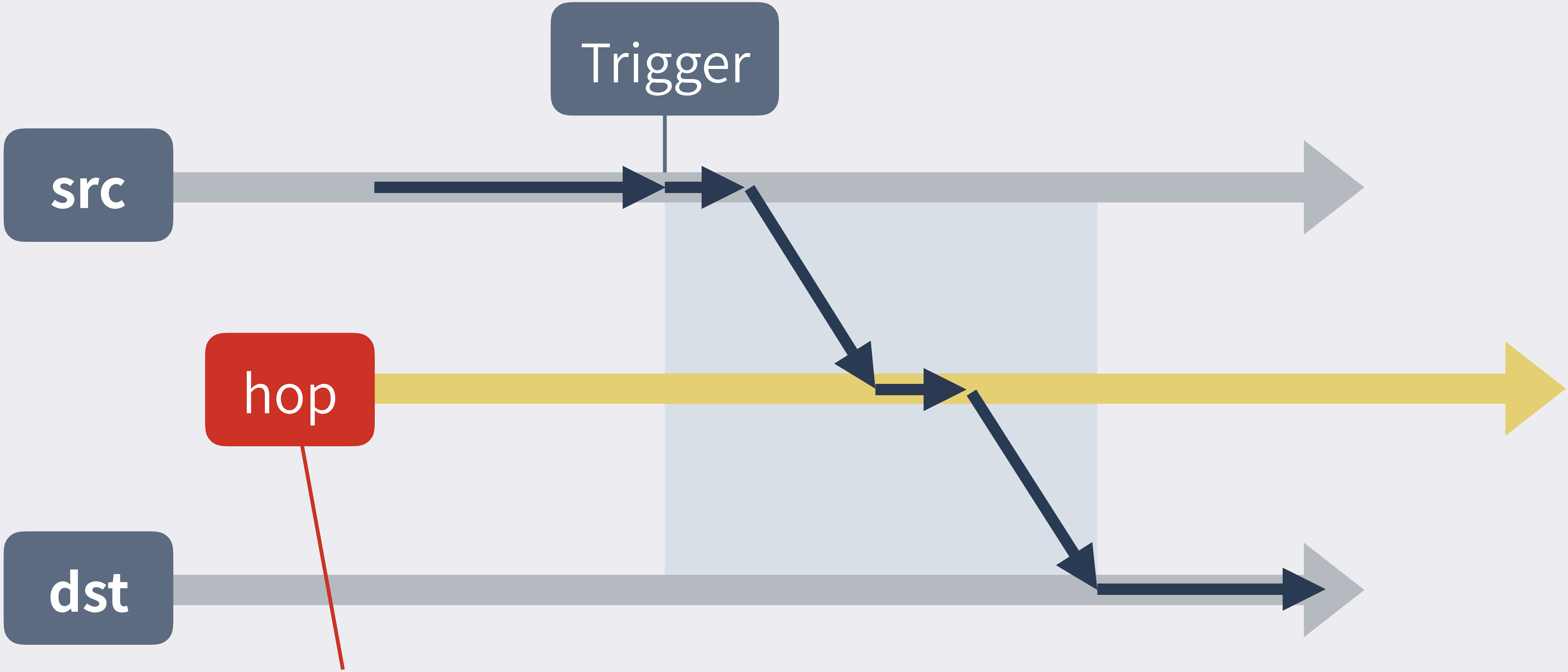
One-Hop Transition Setting



Search Variables

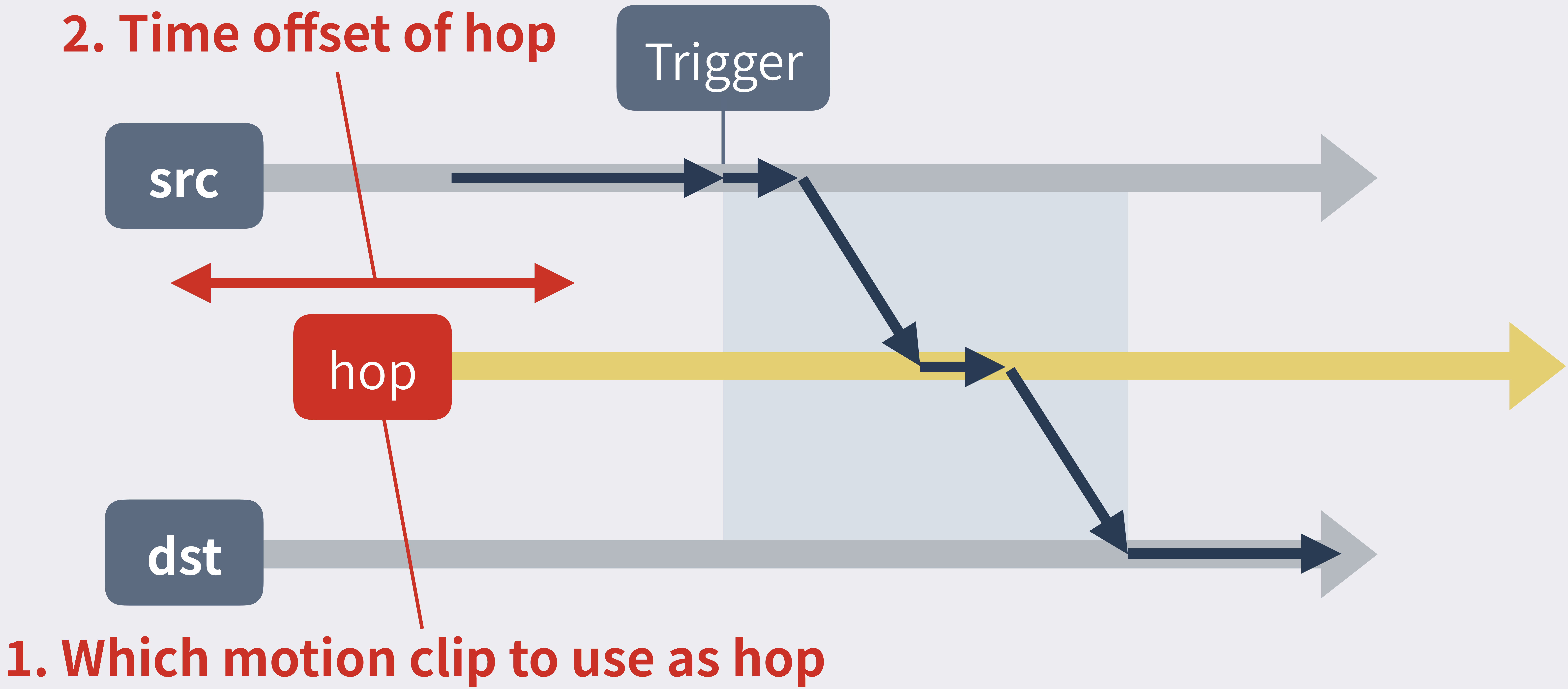


Search Variables

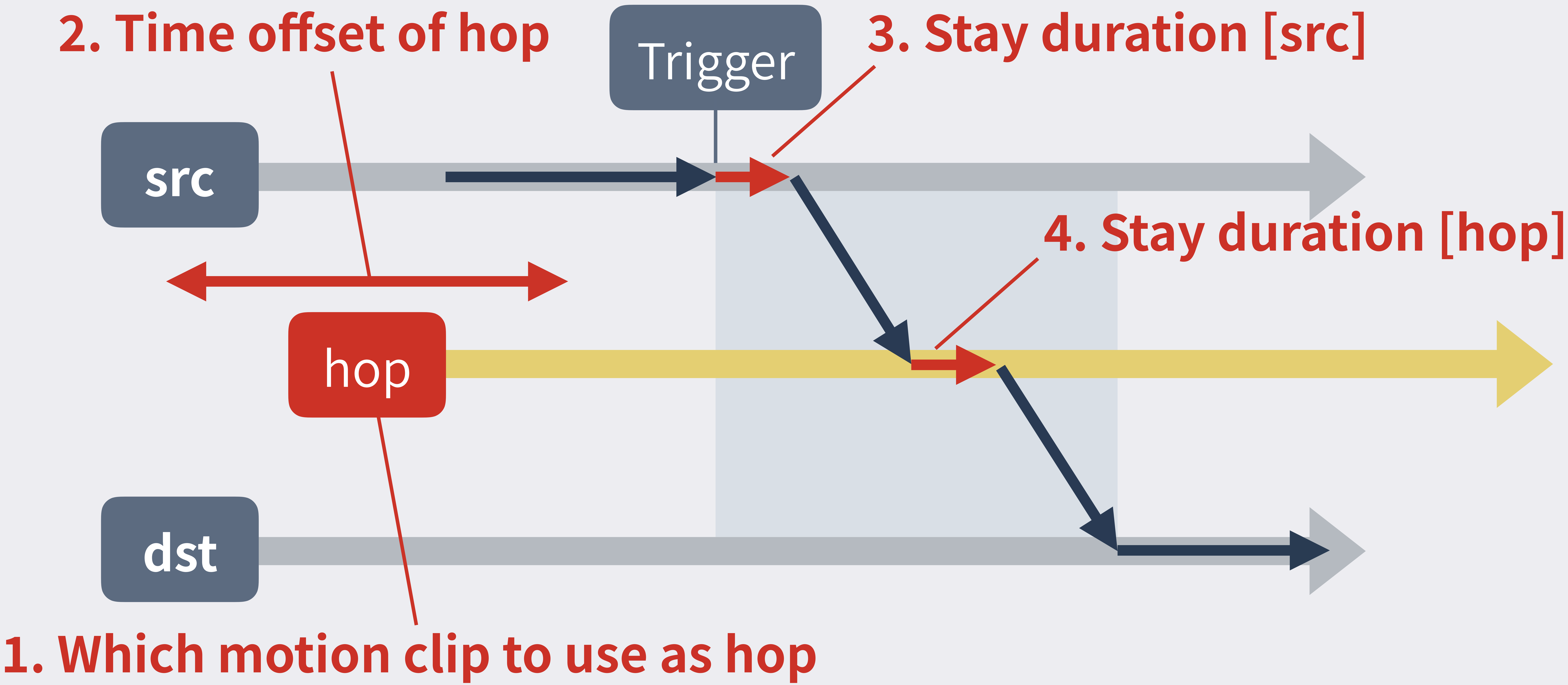


1. Which motion clip to use as hop

Search Variables



Search Variables



Search Objective

$$T_t^* = \arg\min_{T_t} \{ \; S(T_t) \; + \; w \; R(T_t) \; \}$$

Search Objective

$$T_t^* = \arg\min_{T_t} \{ \boxed{S(T_t)} + w \quad R(T_t) \}$$

Smoothness cost:
Encourage each transition to be as natural-looking as possible

Search Objective

$$T_t^* = \arg\min_{T_t}\{ \boxed{S(T_t)} + w \boxed{R(T_t)} \}$$

Smoothness cost:
Encourage each transition to be as natural-looking as possible

Responsiveness cost:
Encourage the overall transition to finish as quickly as possible

Search Objective

$$T_t^* = \arg\min_{T_t}\{ \boxed{S(T_t)} + \boxed{w} \boxed{R(T_t)} \}$$

Smoothness cost:
Encourage each transition to be as natural-looking as possible

Weight:
Control the trade-off

Responsiveness cost:
Encourage the overall transition to finish as quickly as possible

Search Objective

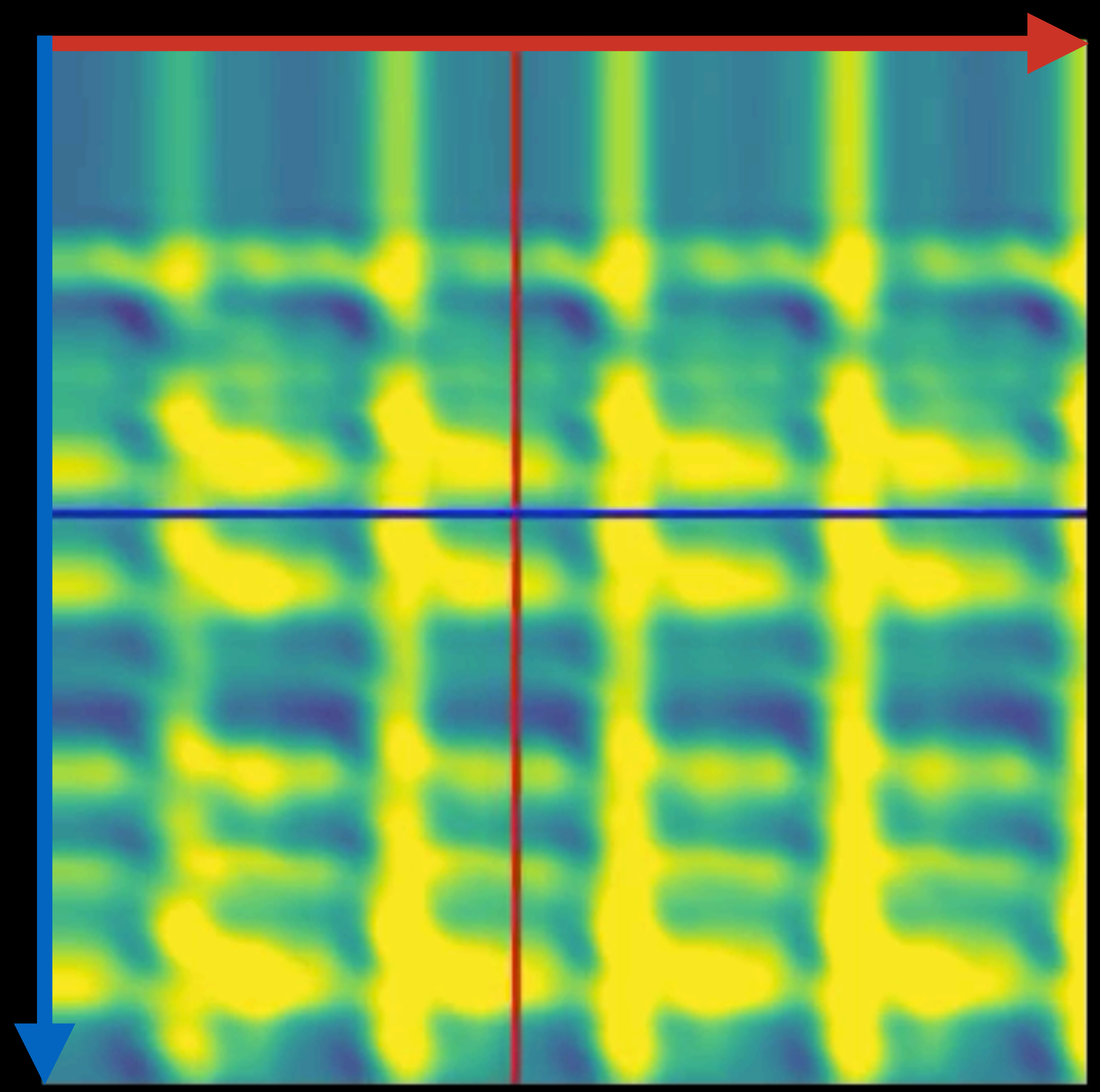
$$T_t^* = \arg\min_{T_t}\{ \boxed{S(T_t)} + \boxed{w} \boxed{R(T_t)} \}$$

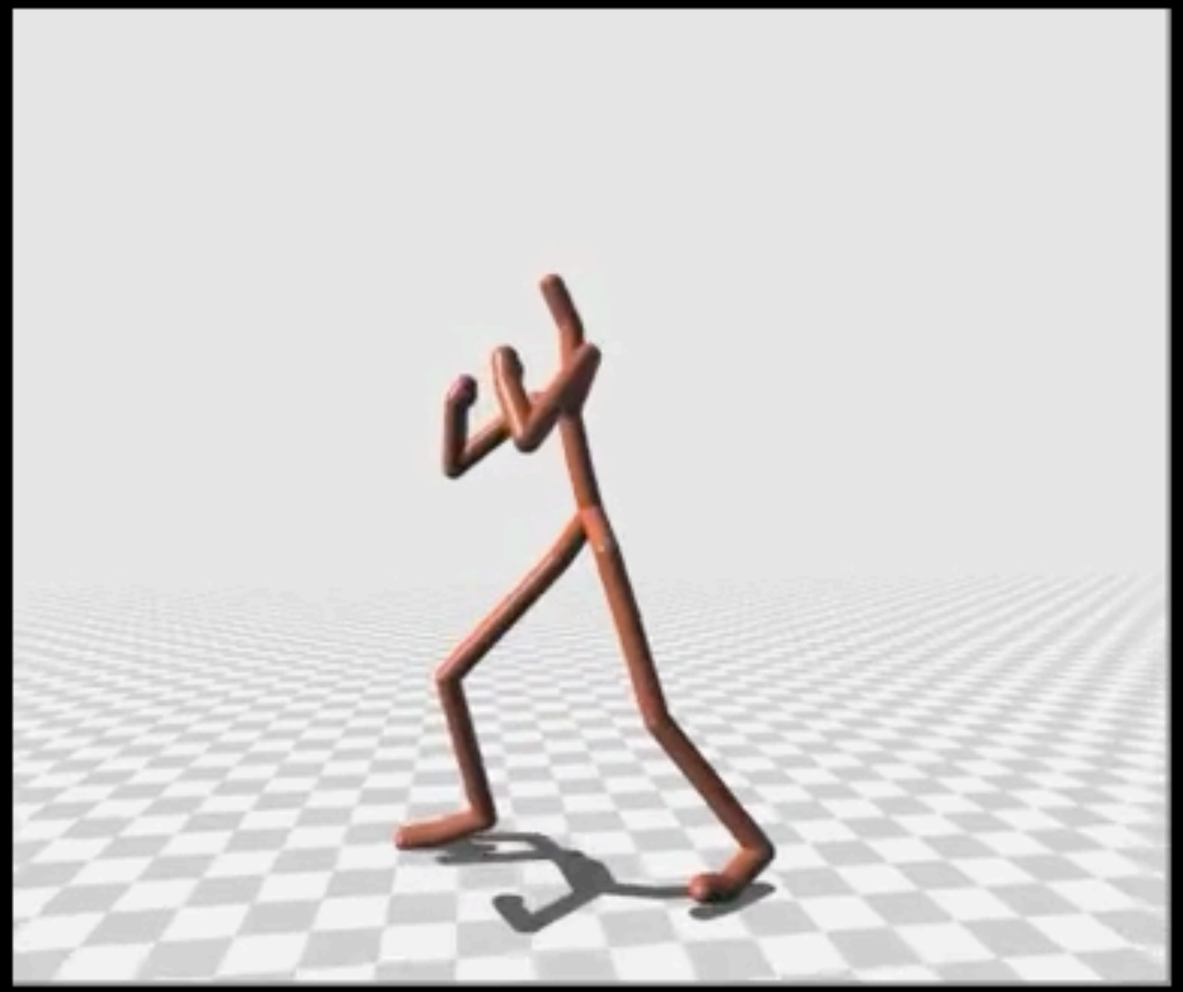
Smoothness cost: Encourage each transition to be as natural-looking as possible

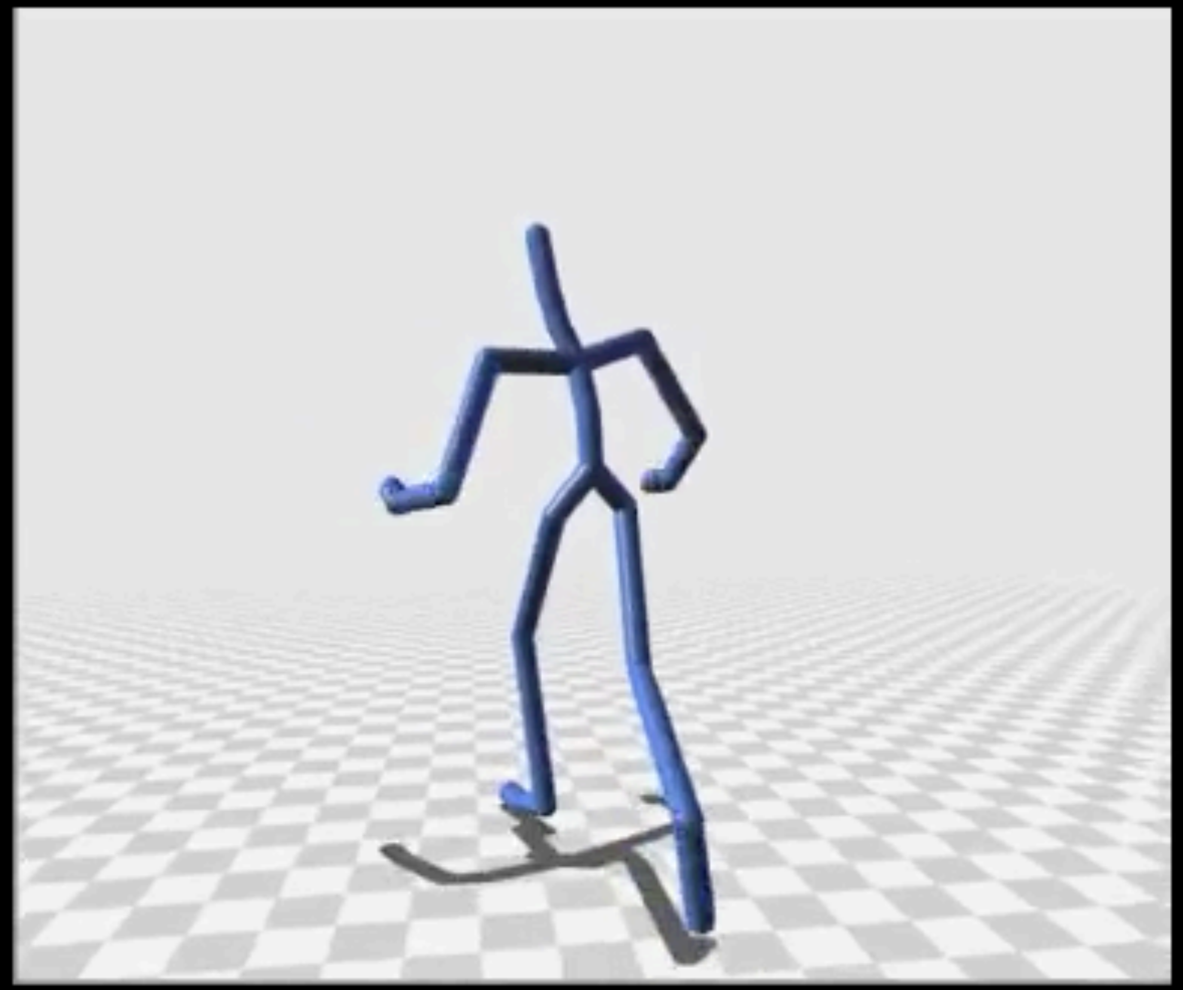
Weight: Control the trade-off

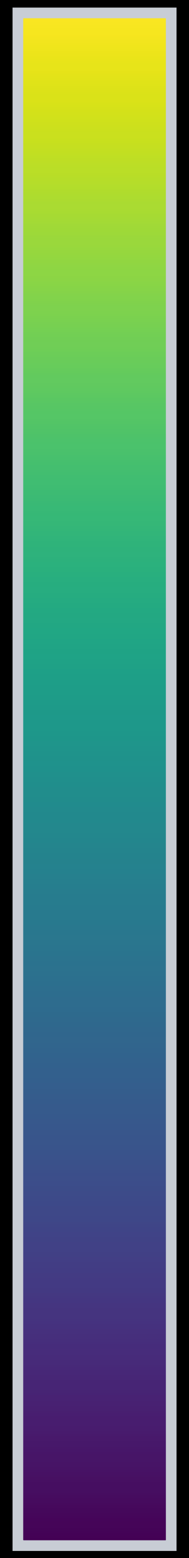
Responsiveness cost: Encourage the overall transition to finish as quickly as possible

Evaluating Smoothness of Transitions [Kovar+02]





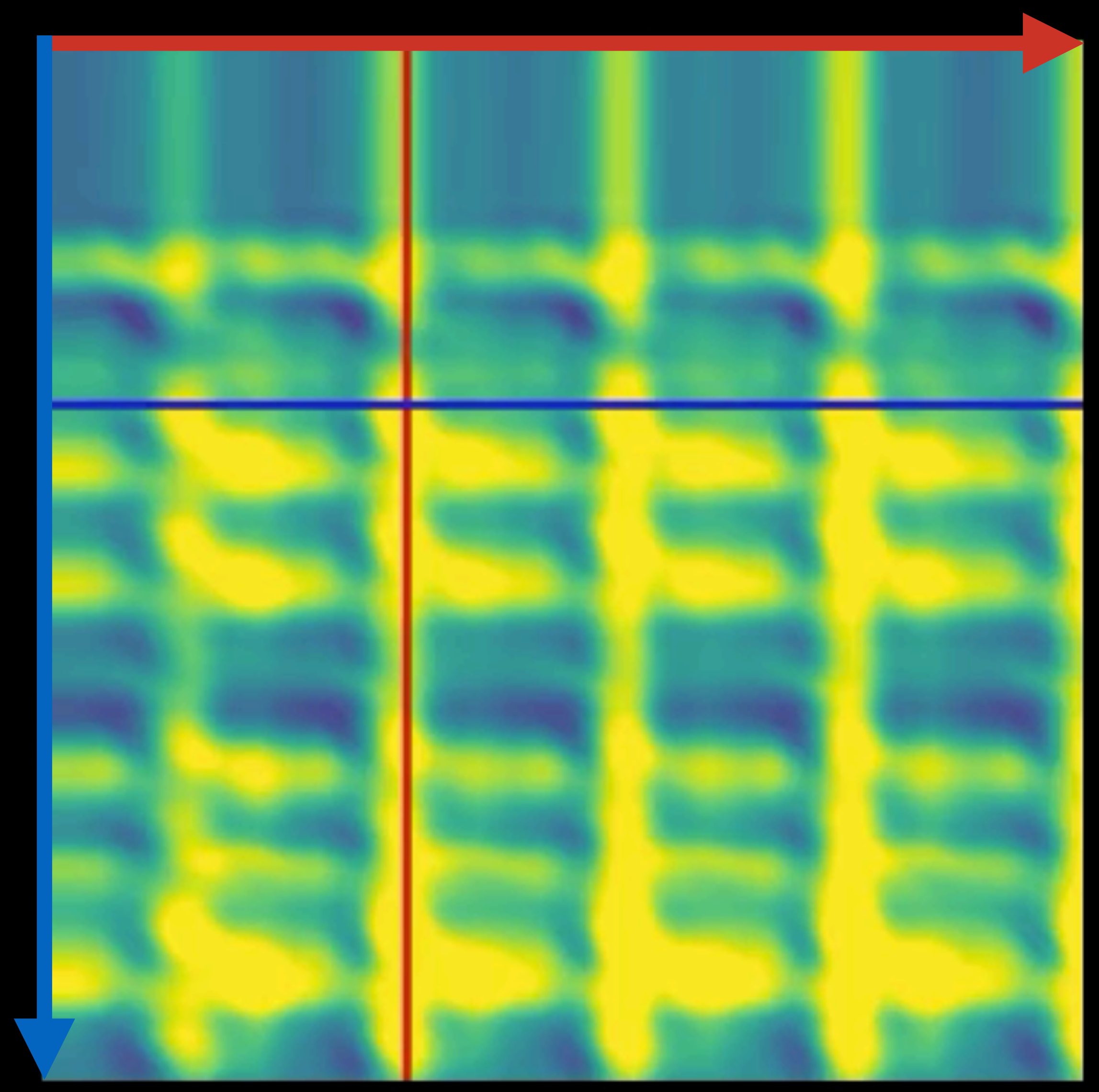




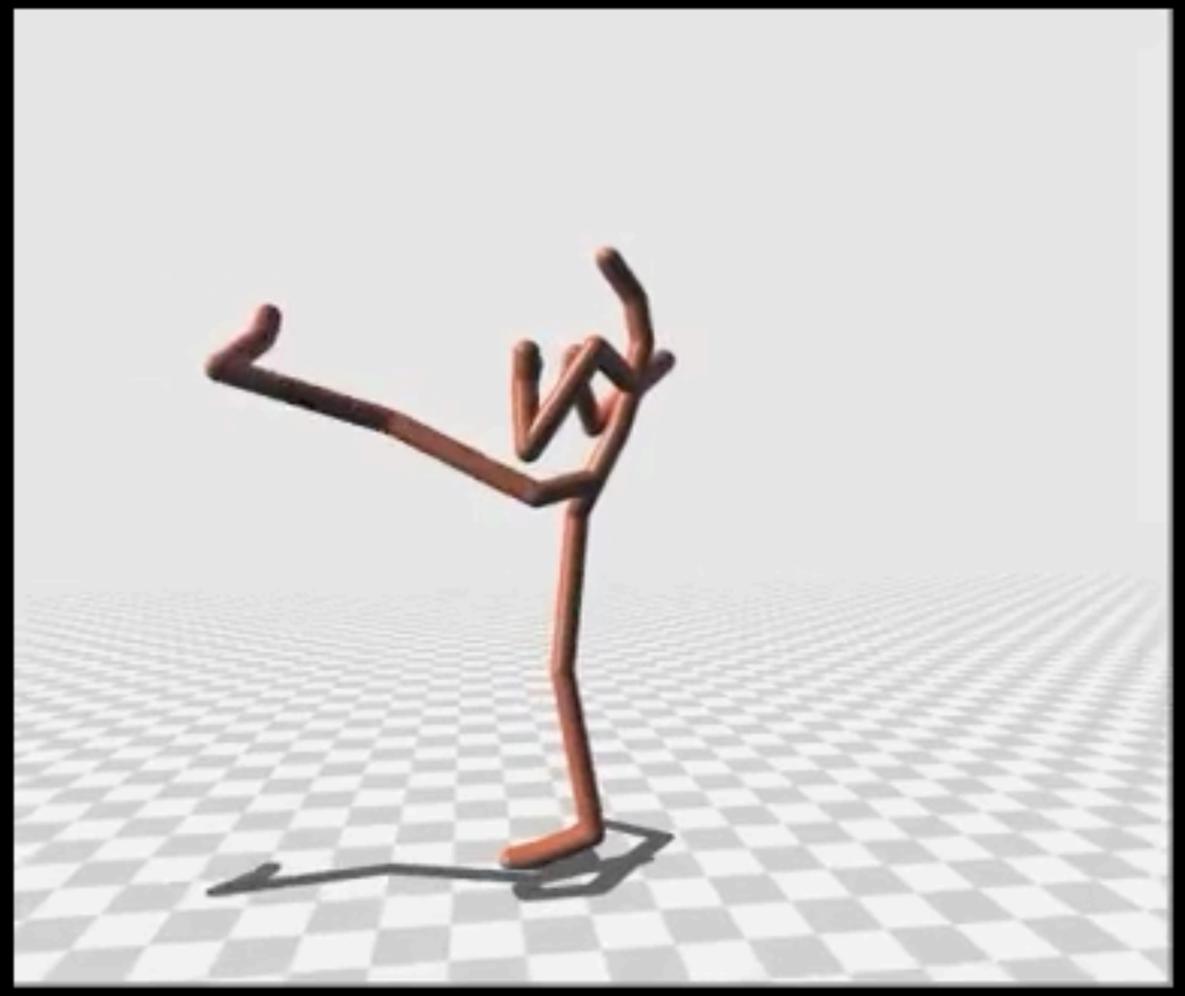
Dissimilar
➡ High cost for transition
(potentially non natural)

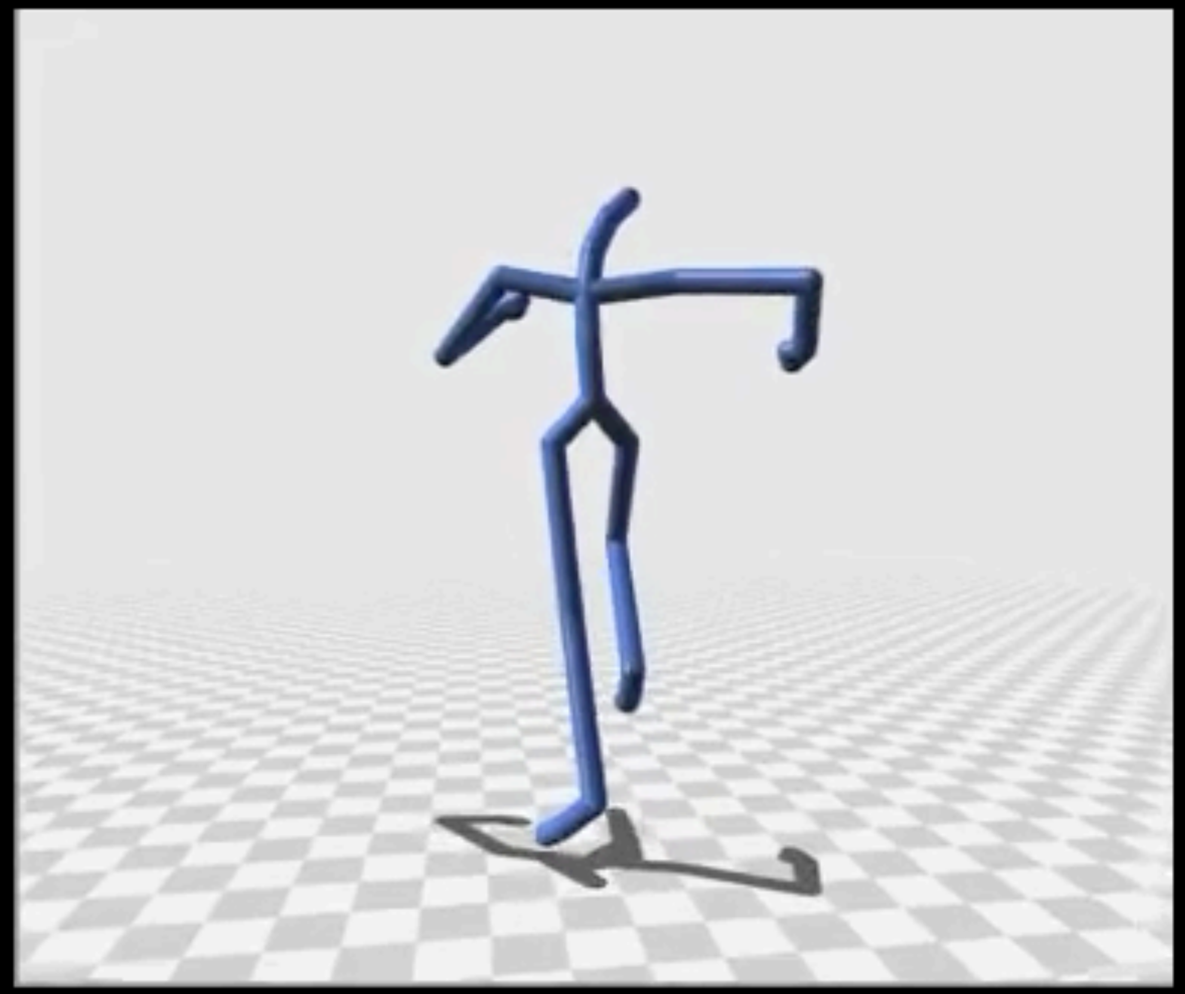
Similar
➡ Low cost for transition
(expected to be natural)

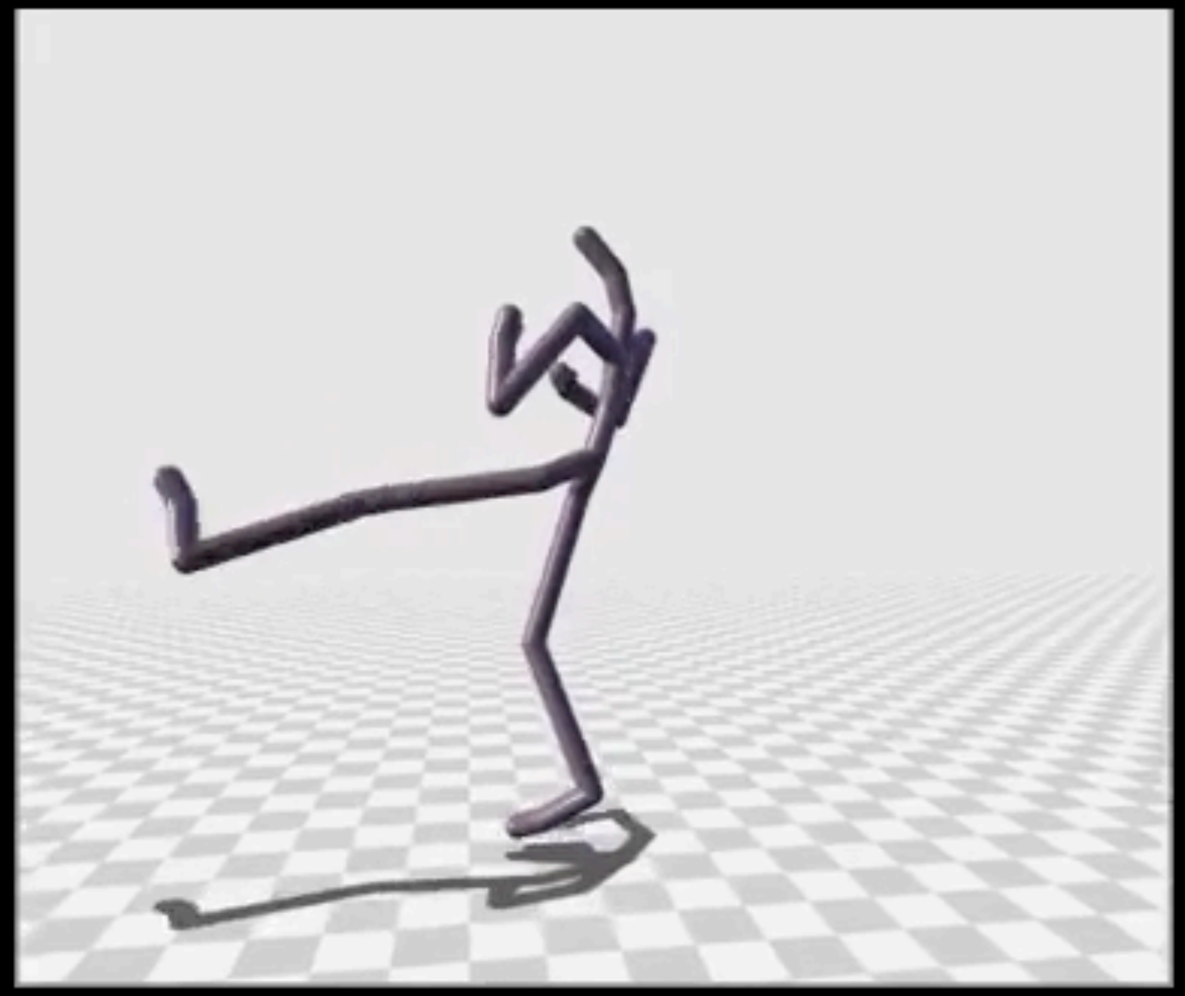
Evaluating Smoothness of Transitions [Kovar+02]



High cost

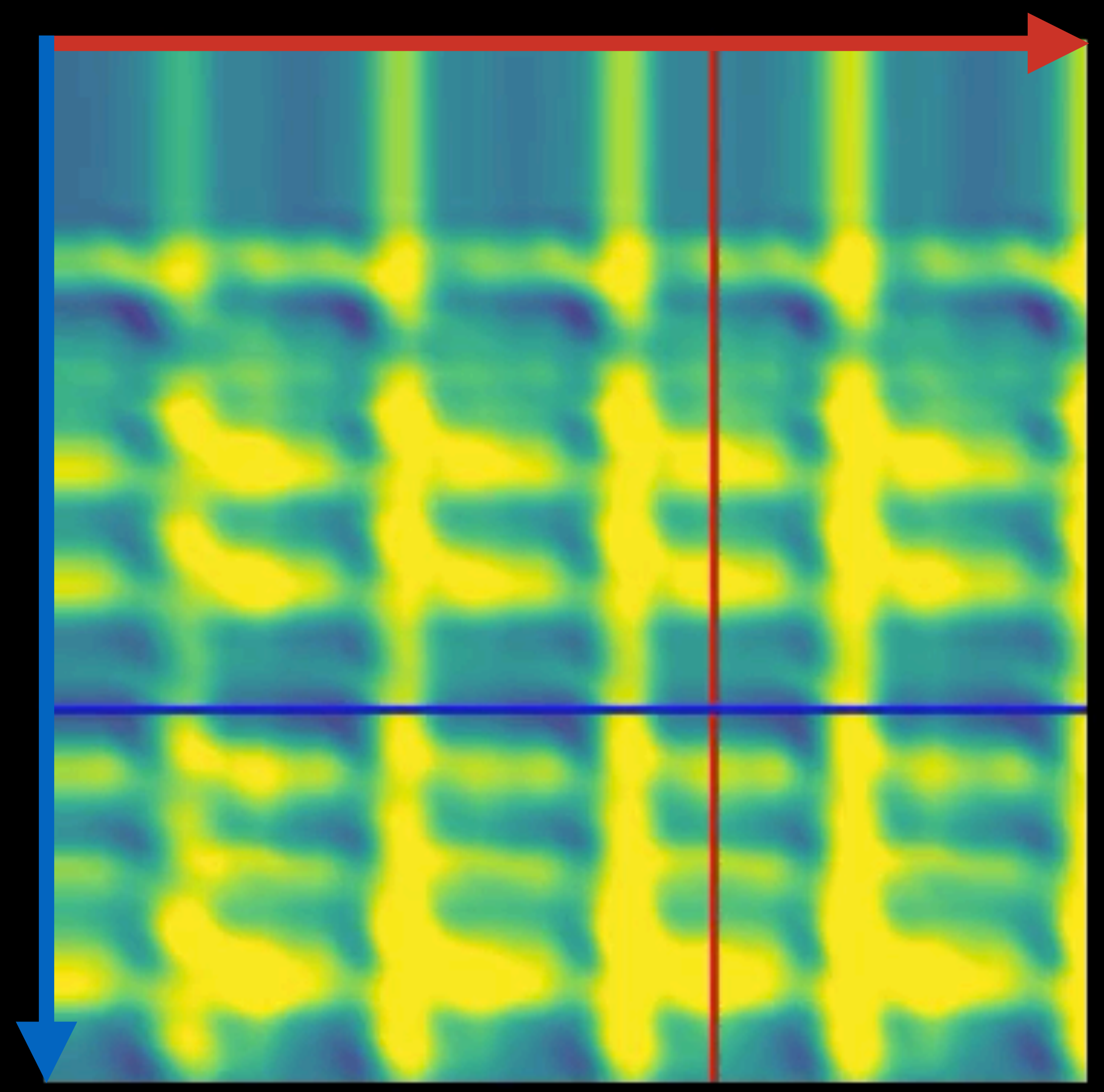




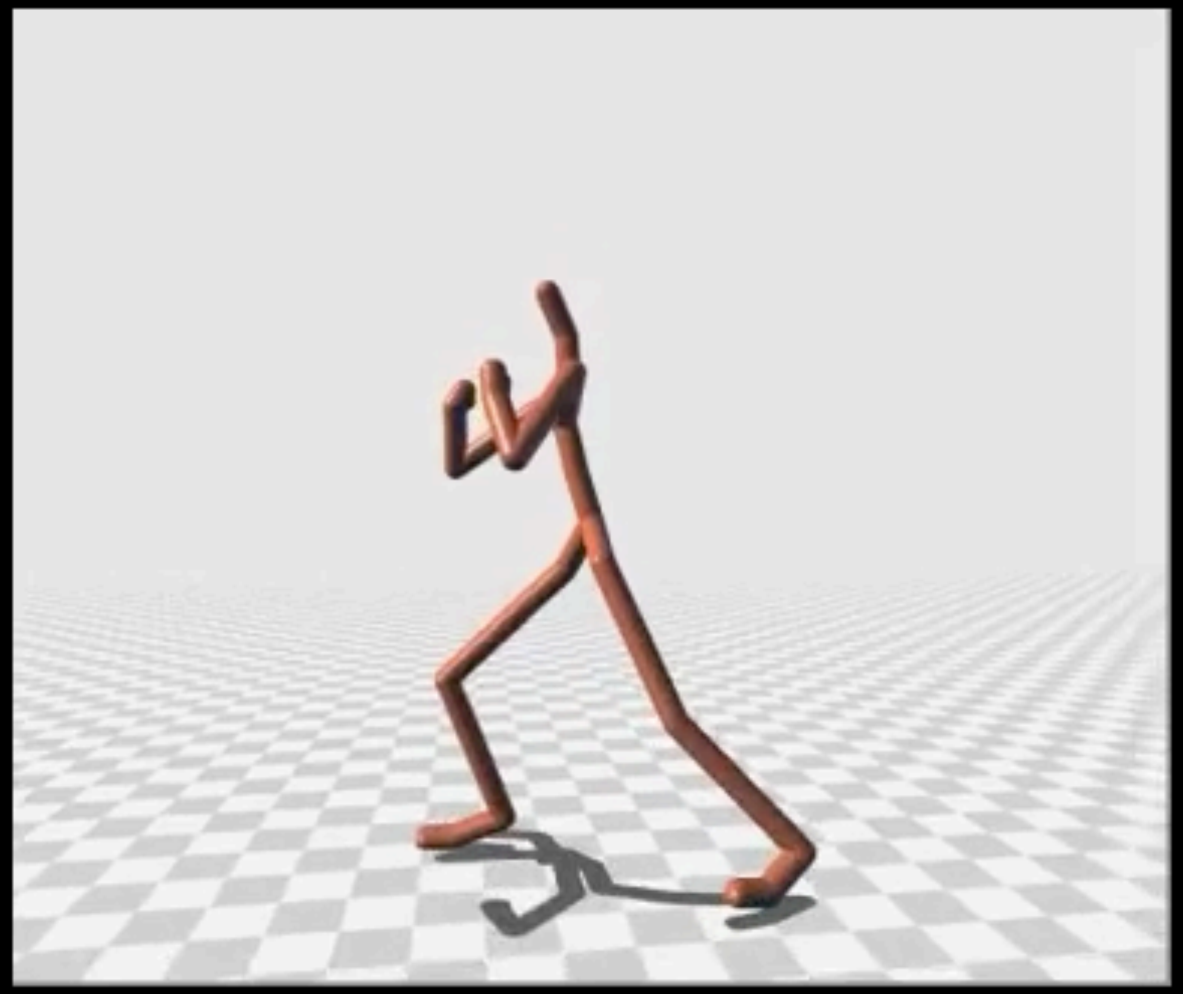


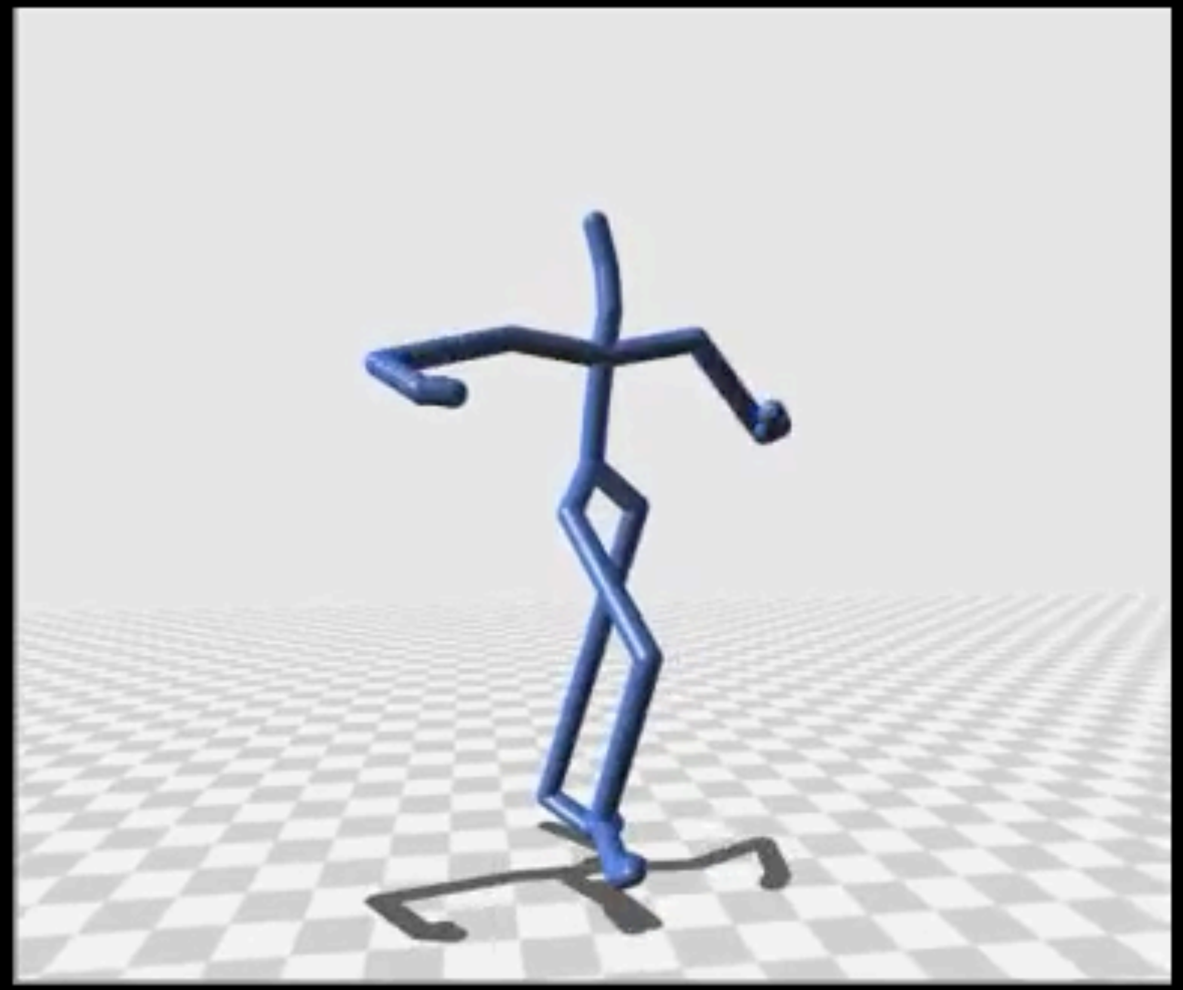
➡ Unnatural transition

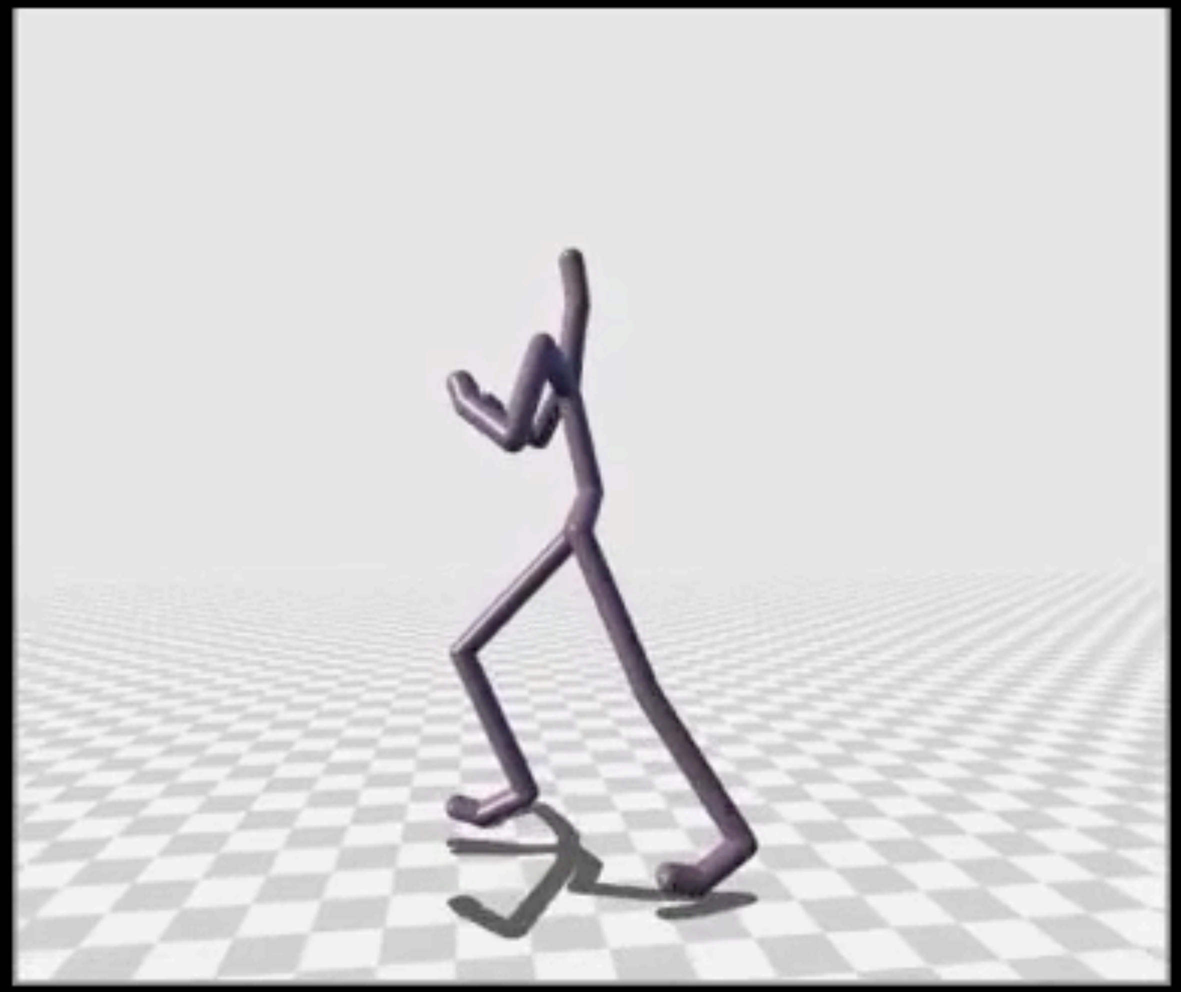
Evaluating Smoothness of Transitions [Kovar+02]



Low cost







➡ Natural transition

2. Solving the Search Problem in Precomputation

Precomputation

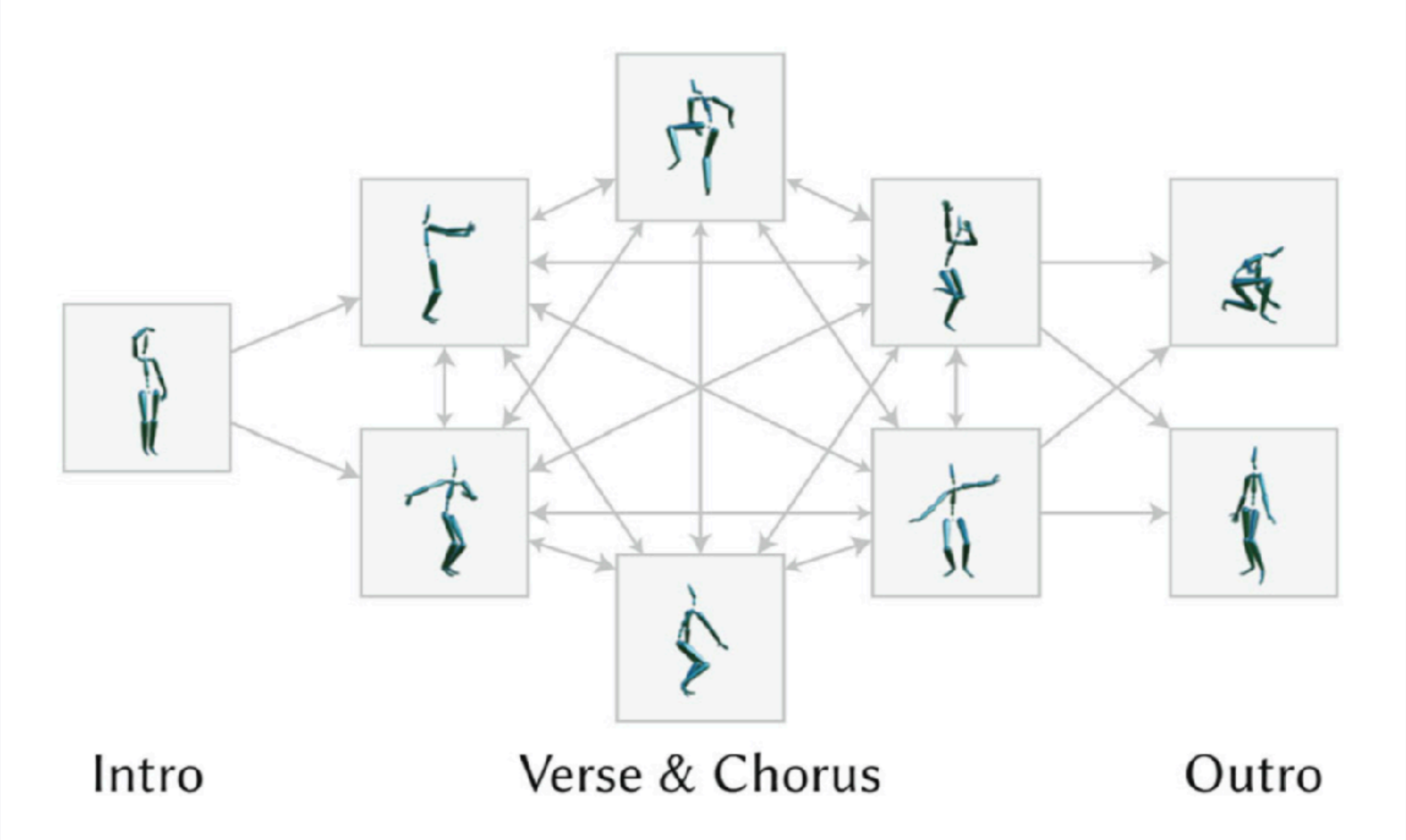
- **Solve the search problem for every possible trigger frame and for every possible state transition**
  - May take a few days depending on the dataset size, but required only once
- **Store optimal transitions in a look-up table**
  - In runtime, retrieve an optimal transition by using the triggered frame as the key

Precomputation

- **Solve the search problem for every possible trigger frame and for every possible state transition**
  - May take a few days depending on the dataset size, but required only once
- **Store optimal transitions in a look-up table**
  - In runtime, retrieve an optimal transition by using the triggered frame as the key

Results

Supposed Scenario: Interactive Dancing

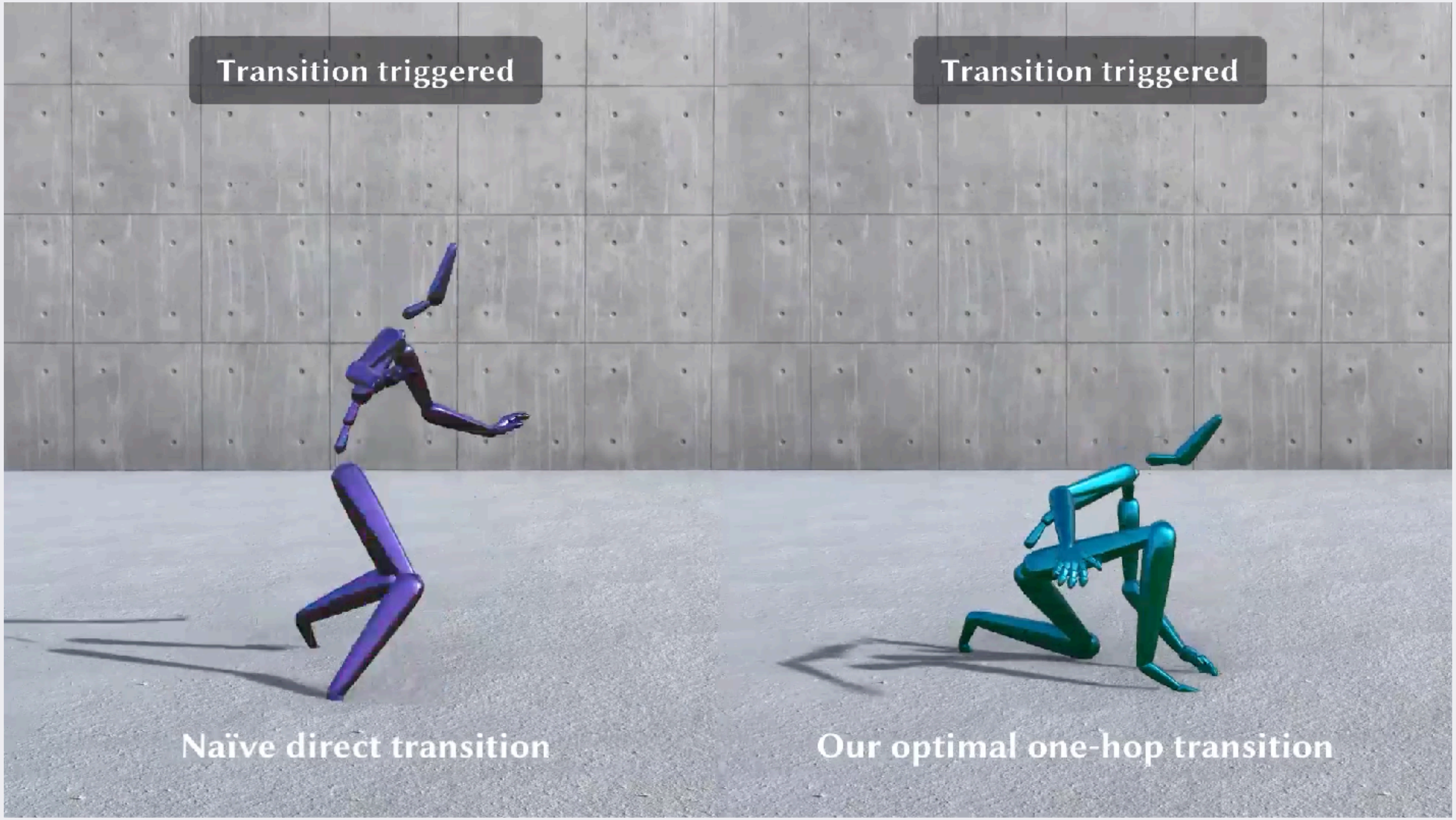


Let a character **dance** using a game pad in an **improvised** manner

Supposed Scenario: Interactive Dancing

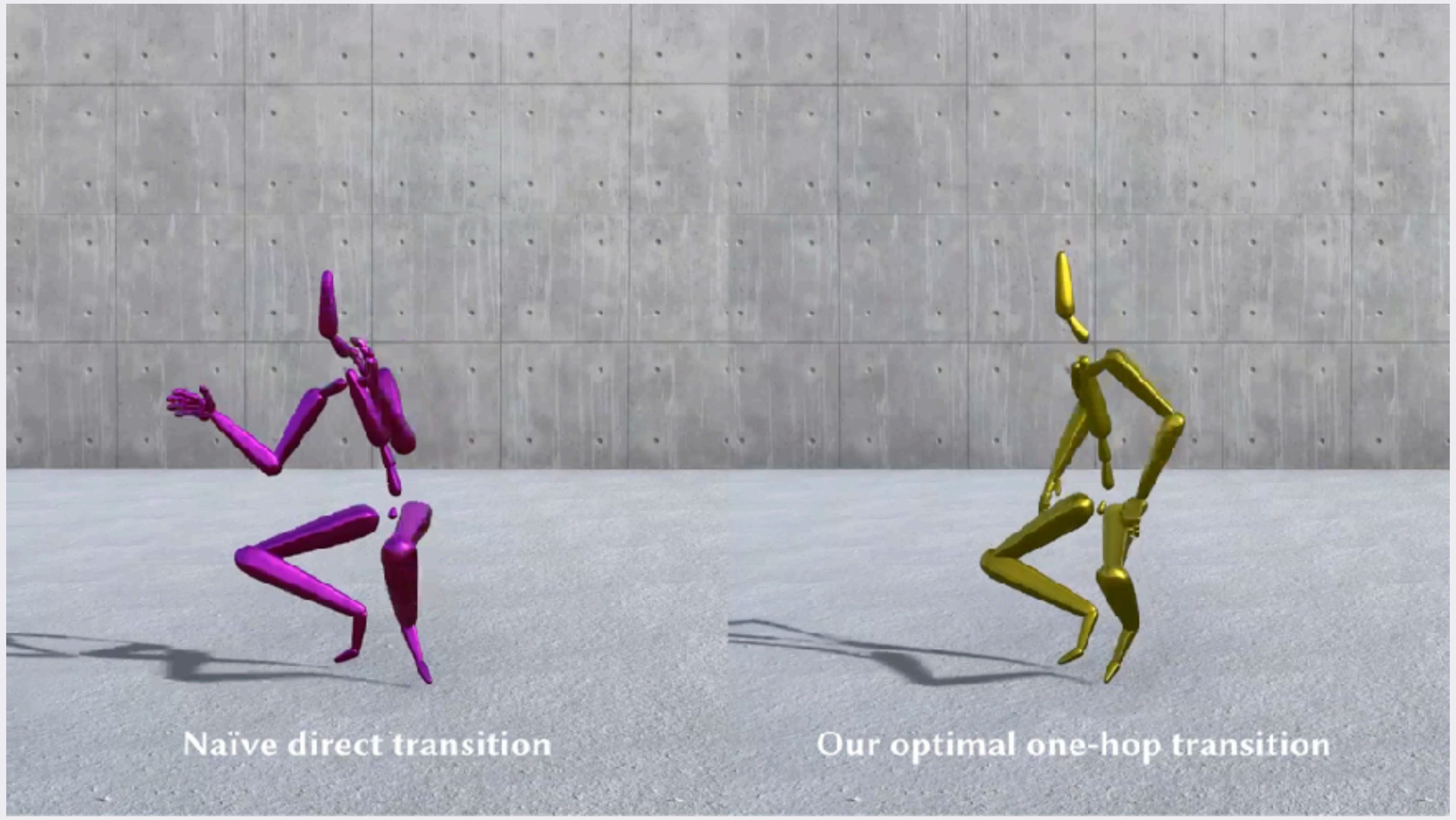
- **Run in Unity**
  - It is straightforward to implement our method using Unity's PlayableAPI
- **Our dance motion dataset**
  - Around 20 motion clips for hop candidates
  - Each motion clip has around 20 sec
- **Use 30 frames (= 0.5 sec) for individual transitions**
  - I.e., src-to-hop (0.5 sec) and hop-to-dst (0.5 sec)
- **No postprocessing applied**
  - E.g., footskating removal (e.g., [Ikemoto+06])

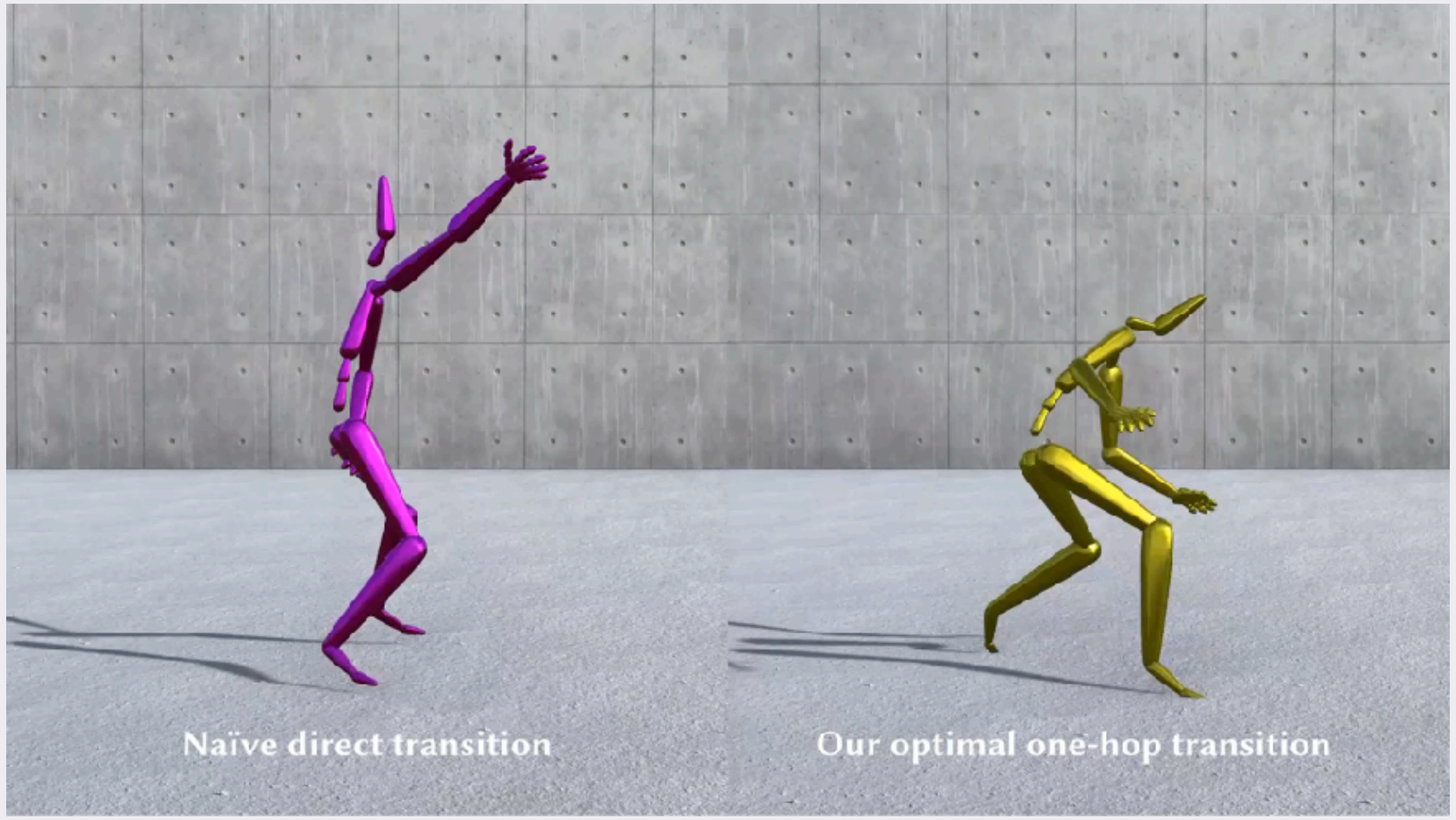
Result: “Ghost Force” Disappears by a Standing-Up Motion



Other Results

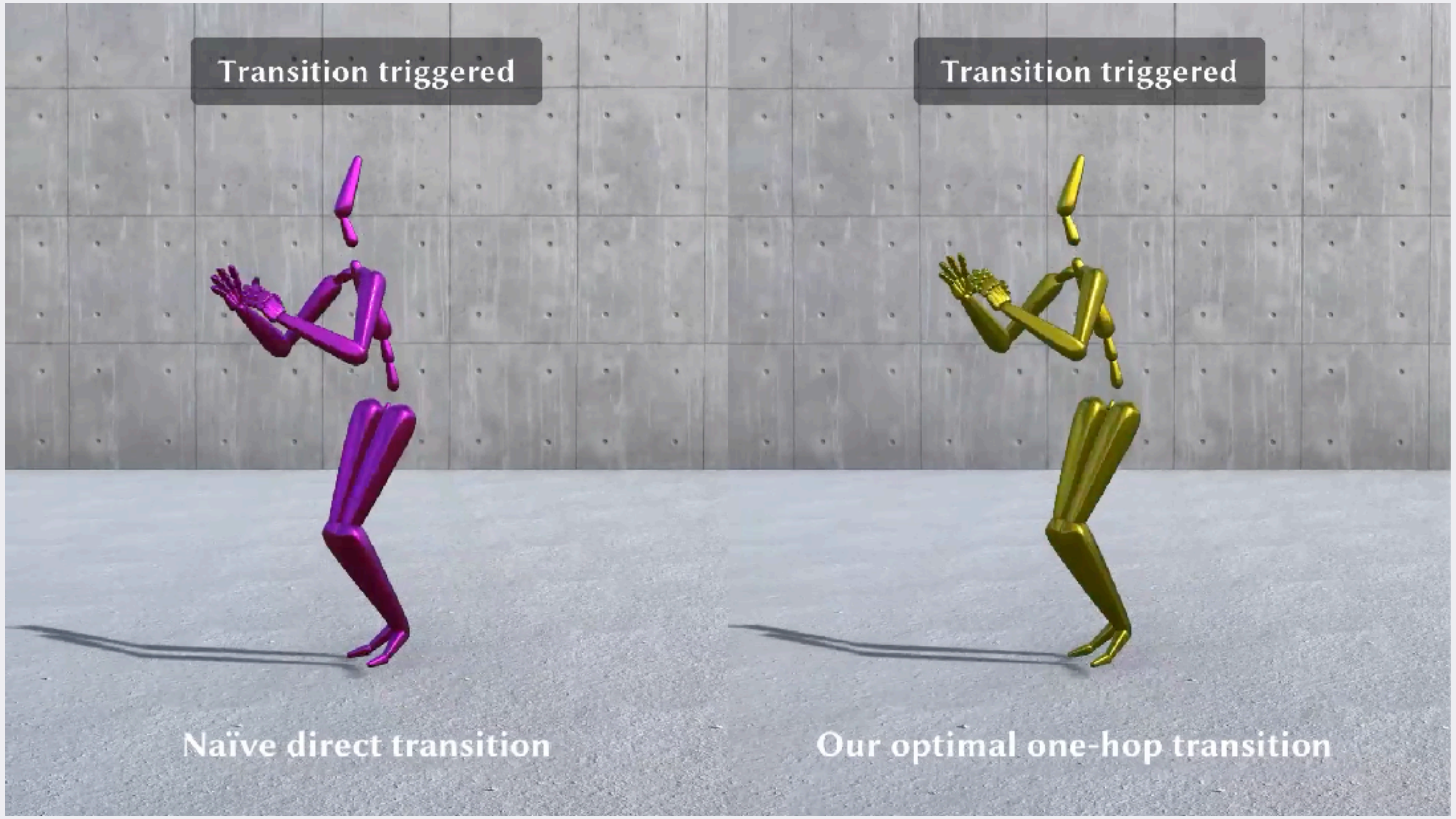
https://koyama.xyz/project/one-hop_motion_transition/





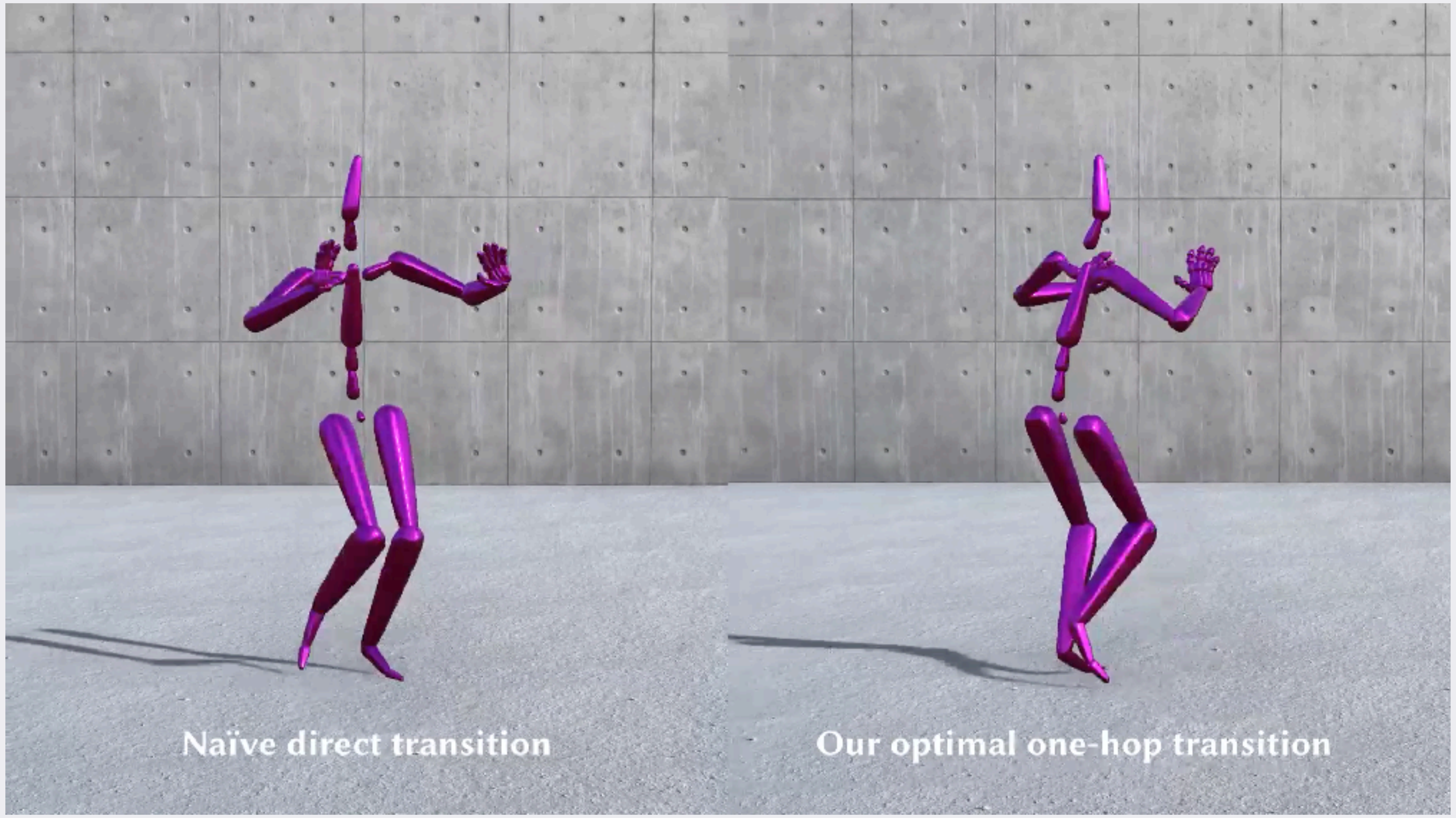
Discussions

The Direct Transition is Included in the Search Space



When the direct transition is already good, our solver is likely to select the one-hop transition that is equivalent to the direct transition

Future Work: Positional & Orientational Constraints

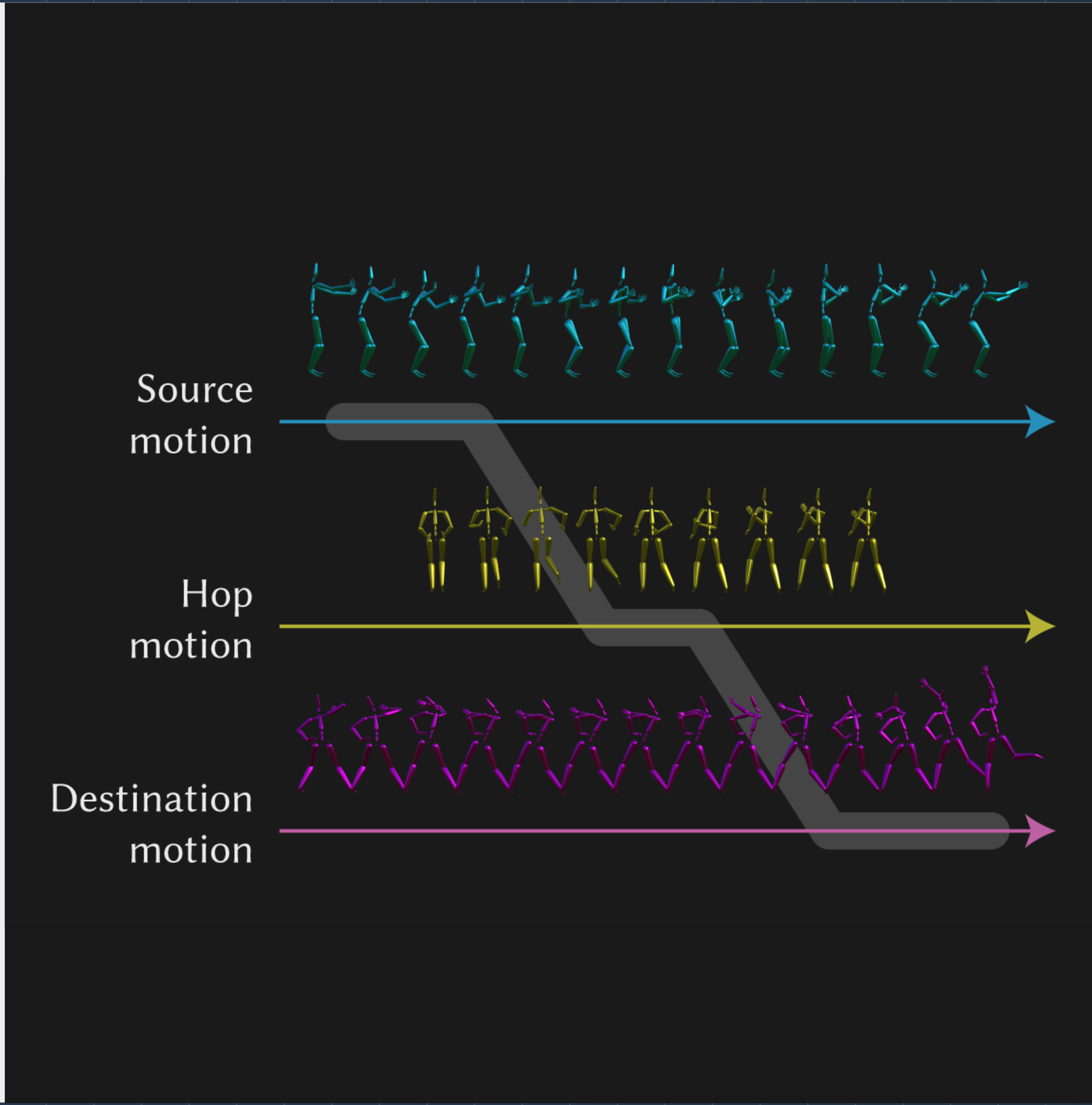


- **Current behavior**
  - Depending on the inserted hop motion, the **position & orientation** may be changed
- **Future work**
  - Add constraints in the search

Summary

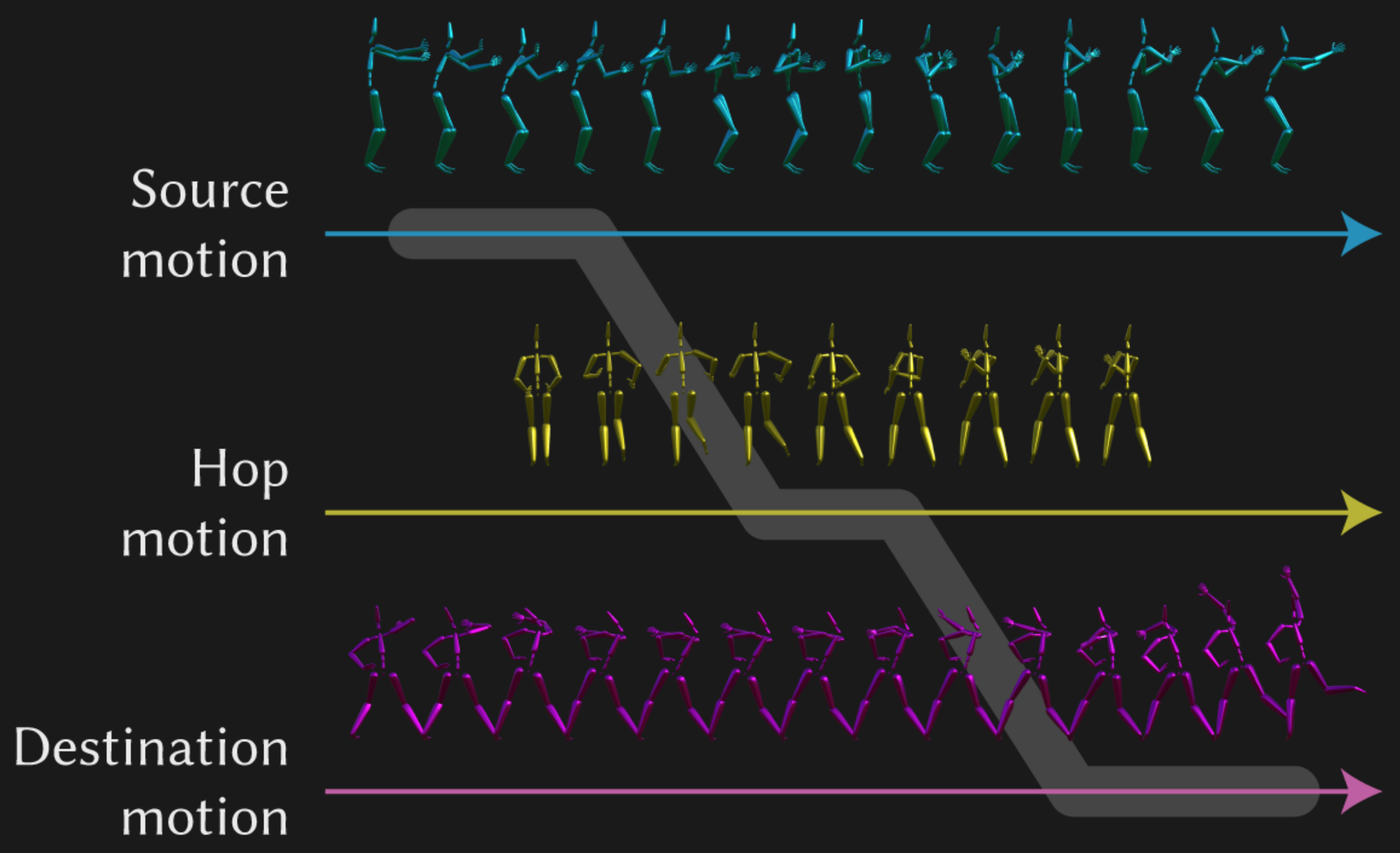
Summary

- **Optimal one-hop motion transition**
  - It inserts a hop motion between the source and destination motions in an optimal way
- **Consider both smoothness and responsiveness**
  - Try to maximize both in balance
- **Can be precomputed**
  - Run in real-time apps



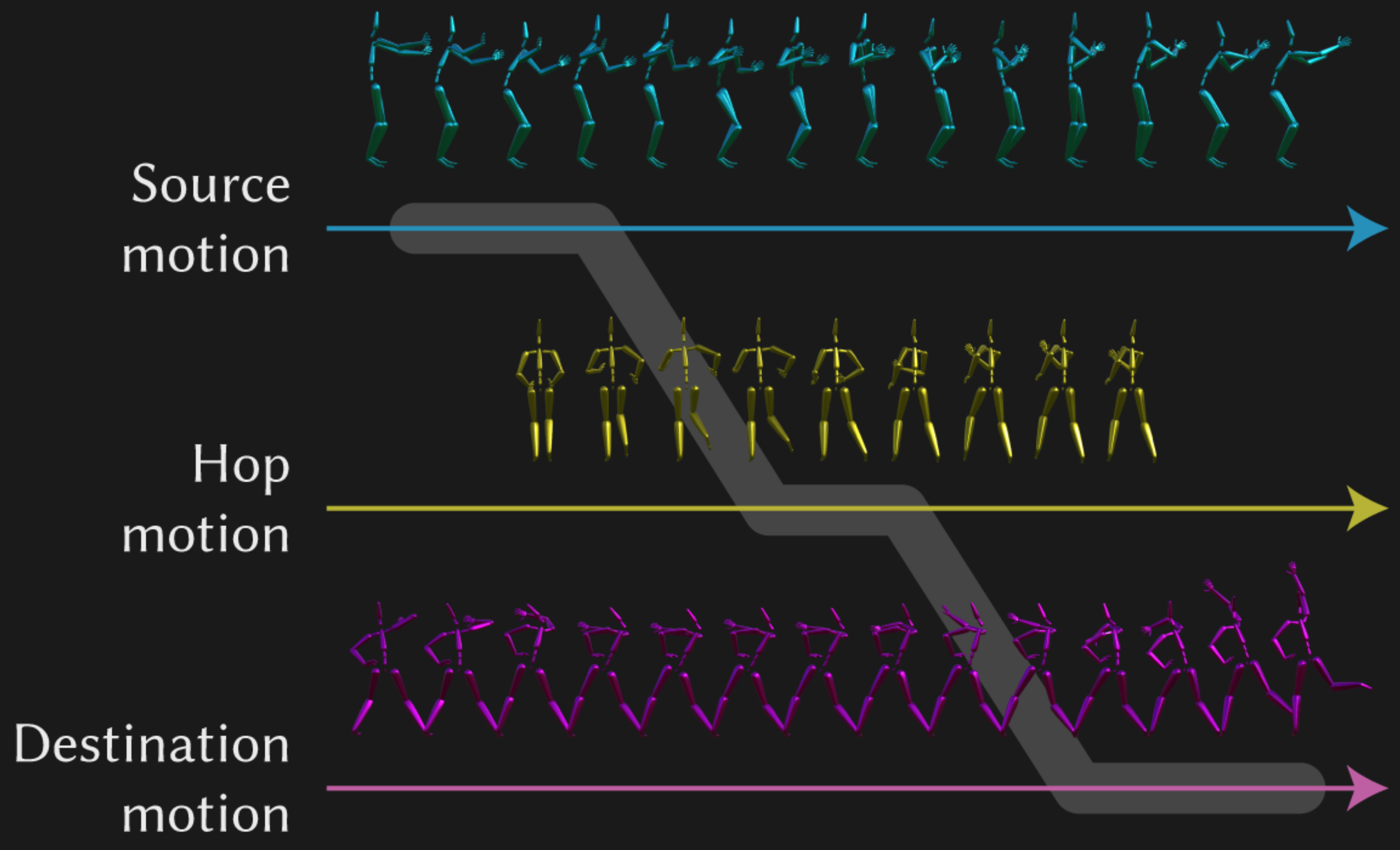
Summary

- **Optimal one-hop motion transition**
  - It inserts a hop motion between the source and destination motions in an optimal way
- **Consider both smoothness and responsiveness**
  - Try to maximize both in balance
- **Can be precomputed**
  - Run in real-time apps



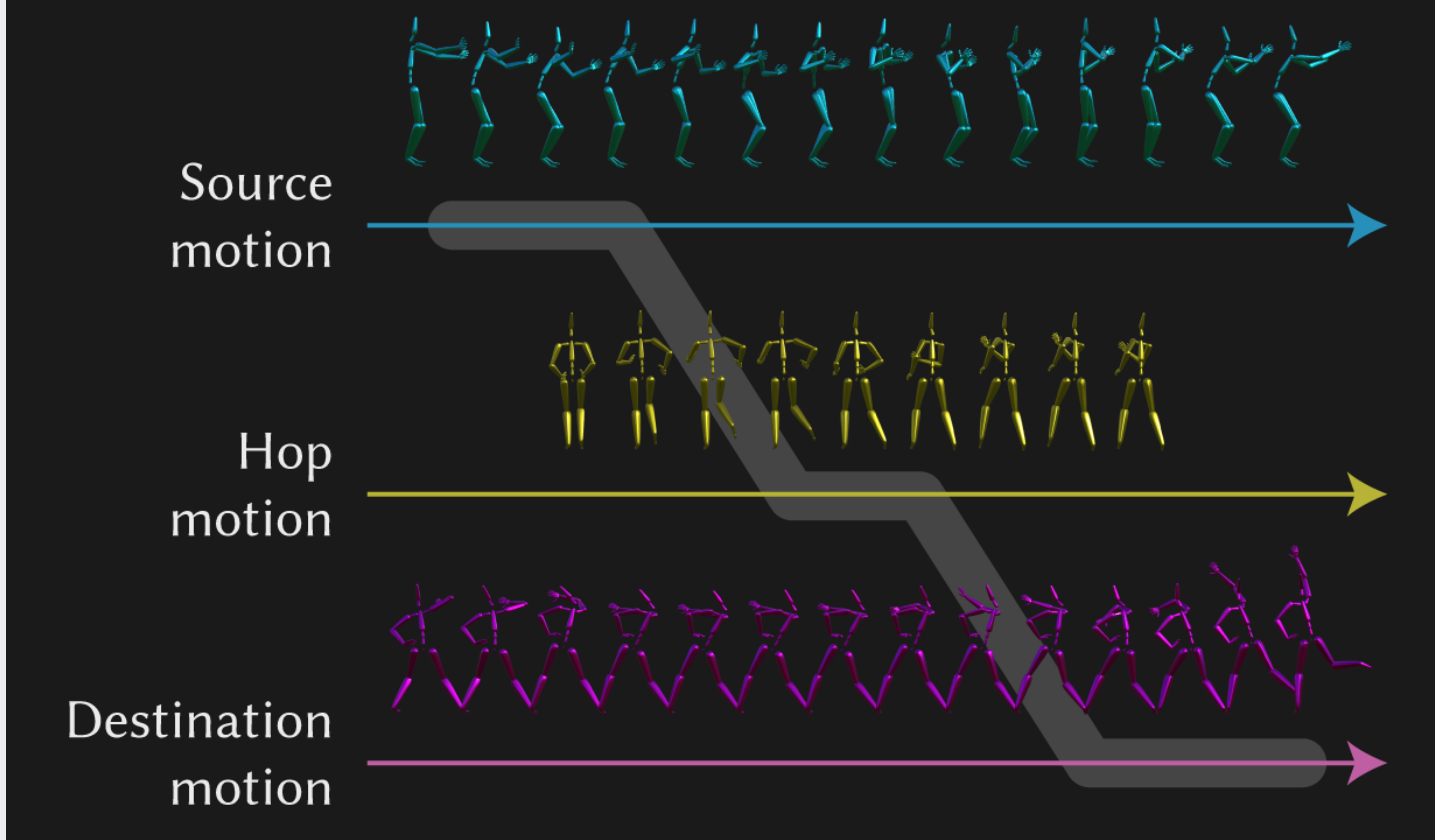
Summary

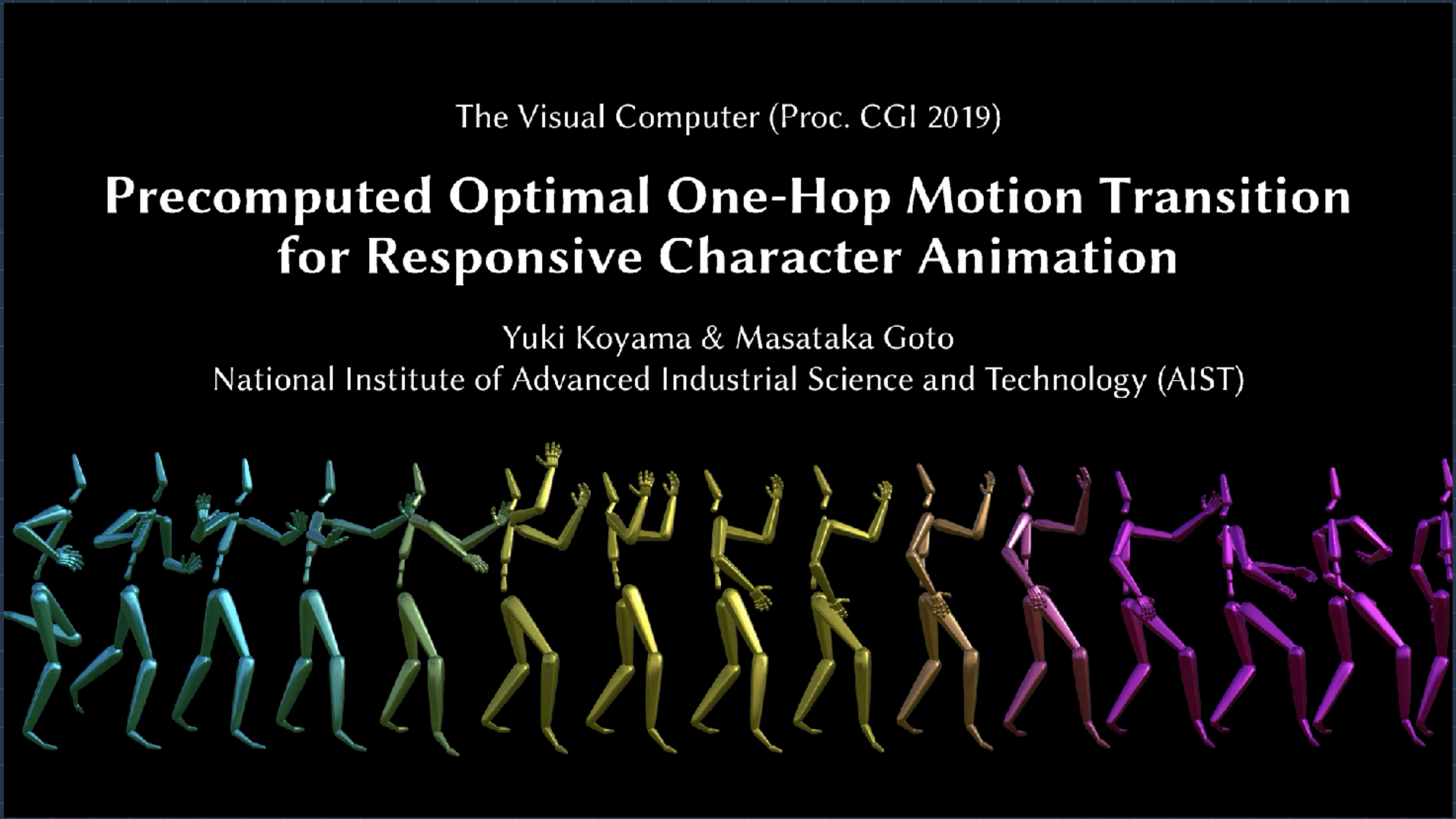
- **Optimal one-hop motion transition**
  - It inserts a hop motion between the source and destination motions in an optimal way
- **Consider both smoothness and responsiveness**
  - Try to maximize both in balance
- **Can be precomputed**
  - Run in real-time apps



Summary

- **Optimal one-hop motion transition**
  - It inserts a hop motion between the source and destination motions in an optimal way
- **Consider both smoothness and responsiveness**
  - Try to maximize both in balance
- **Can be precomputed**
  - Run in real-time apps





The paper, videos, and slides are available at

https://koyama.xyz/project/one-hop_motion_transition/

Appendix

Comparison: Different Responsiveness Weights

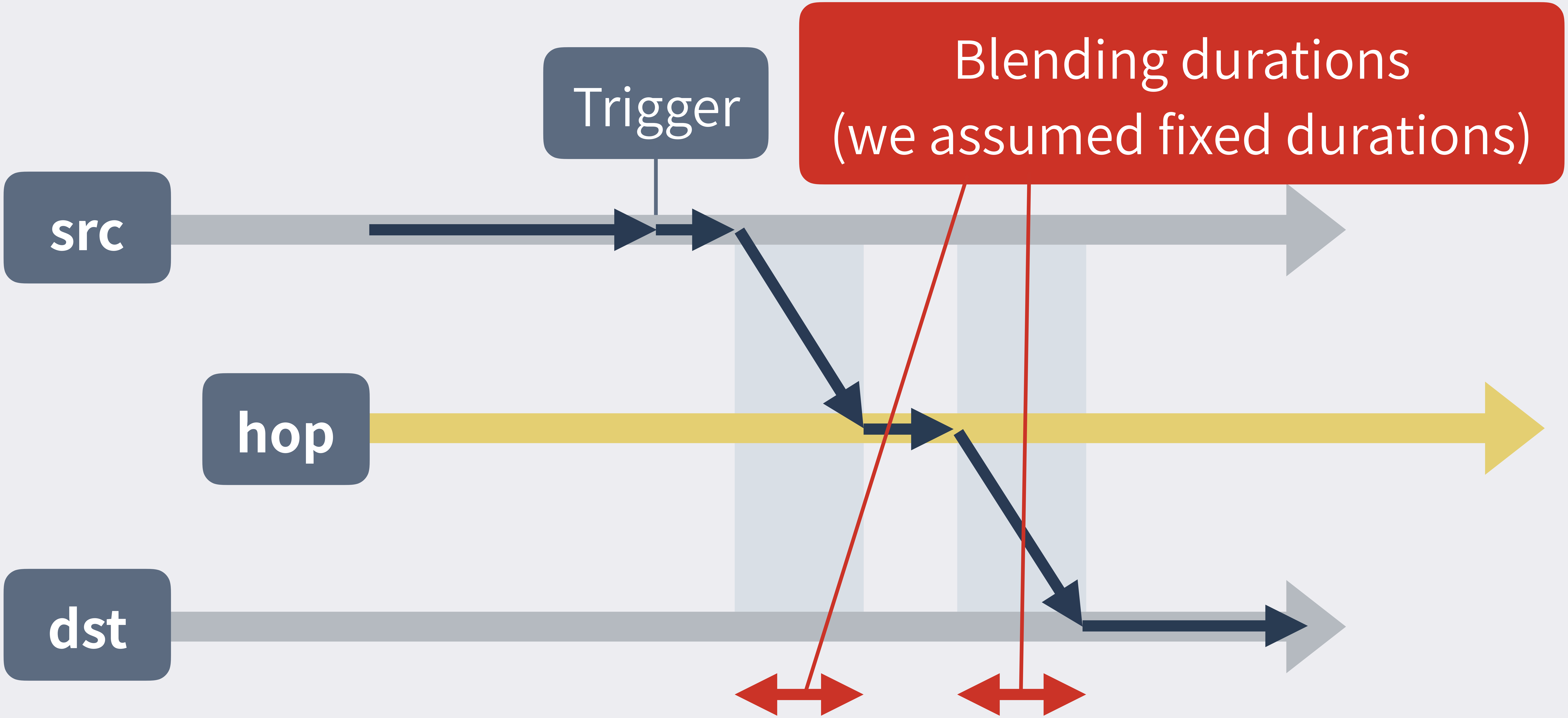
Naïve direct transition

Our one-hop transition
[large weight]
more responsive, less smooth

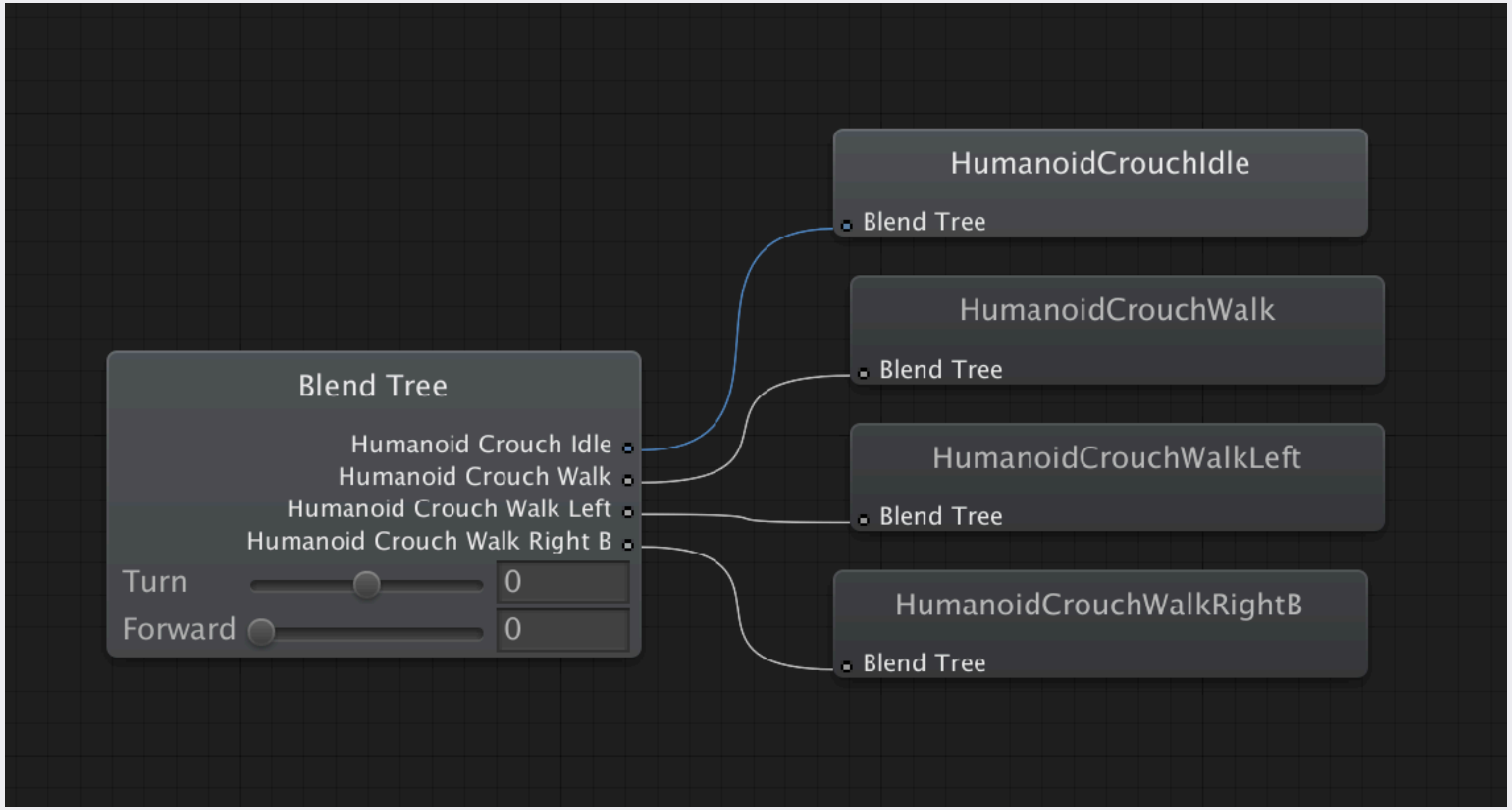
Our one-hop transition
[default weight]

Our one-hop transition
[small weight]
less responsive, more smooth

Future Work: Optimizing Blending Durations (c.f., [Wang+08])



Future Work: Blend Tree



Blend Tree:
Each state can also be associated a dynamic mixture of motion clips

Challenge:
How can we precompute all the possible blends and store the results?