TALOS Training Sessions

Joint trajectory controller

PAL ROBOTICS

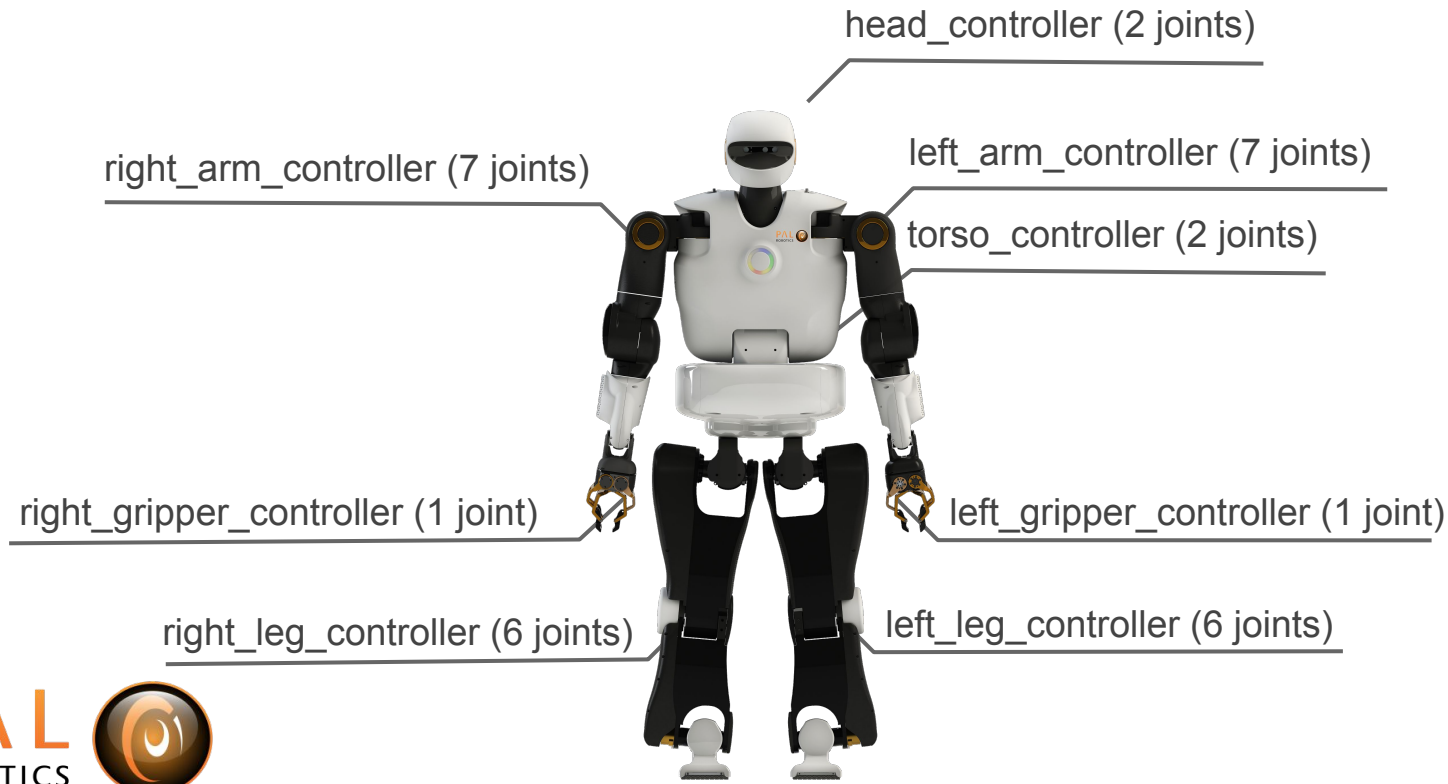
Joint trajectory controller

- Controller for executing **joint-space trajectories** on a group of joints
- Implemented as a **ros_control** controller plugin
- **Walking** is **not** performed by this controller

PAL ROBOTICS

Joint trajectory controller

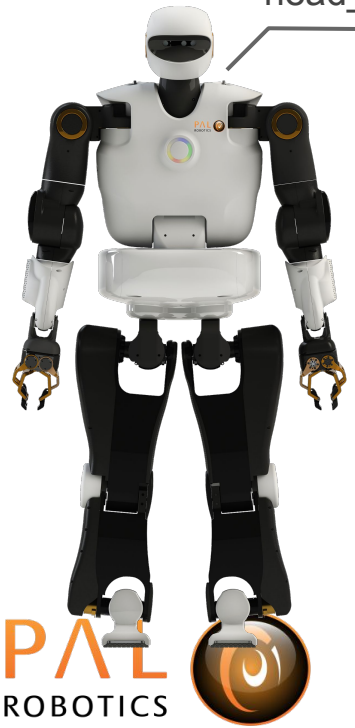
TALOS has eight joint groups



PAL ROBOTICS

Joint trajectory controller

Example controller configuration file (YAML format)




```yaml
head_controller:
  type: "position_controllers/JointTrajectoryController"
  joints:
    - head_1_joint
    - head_2_joint

  constraints:
    goal_time: 0.5
    head_1_joint:
      goal: 0.01
    head_2_joint:
      goal: 0.01
```

PAL ROBOTICS

Joint trajectory controller

Running the controllers

- By default TALOS is in **idle mode**. No controllers are running
- **Load** and **start** the controllers

```
roslaunch talos_controller_configuration full_body_position_controllers.launch
```

- **Verify** the running controllers

```
rosservice call /controller_manager/list_controllers
```

- **Stop** and **unload** the controllers: `Ctrl+c`

Joint trajectory controller

Trajectory is **represented** as a set of **waypoints**:

- time
- position
- velocity (optional, but recommended)
- acceleration (optional)

Trajectory is **generated** by a **spline interpolator** in realtime:

- **Linear:** pos waypoints → pos continuity
- **Cubic:** pos+vel waypoints → vel continuity
- **Quintic:** pos+vel+acc waypoints → acc continuity

Trajectory is **executed** as best as the **joint limits** allow.

Joint trajectory controller

Trajectory representation

The `trajectory_msgs/JointTrajectory` message

```
Header header
string[] joint_names
JointTrajectoryPoint[] points
```

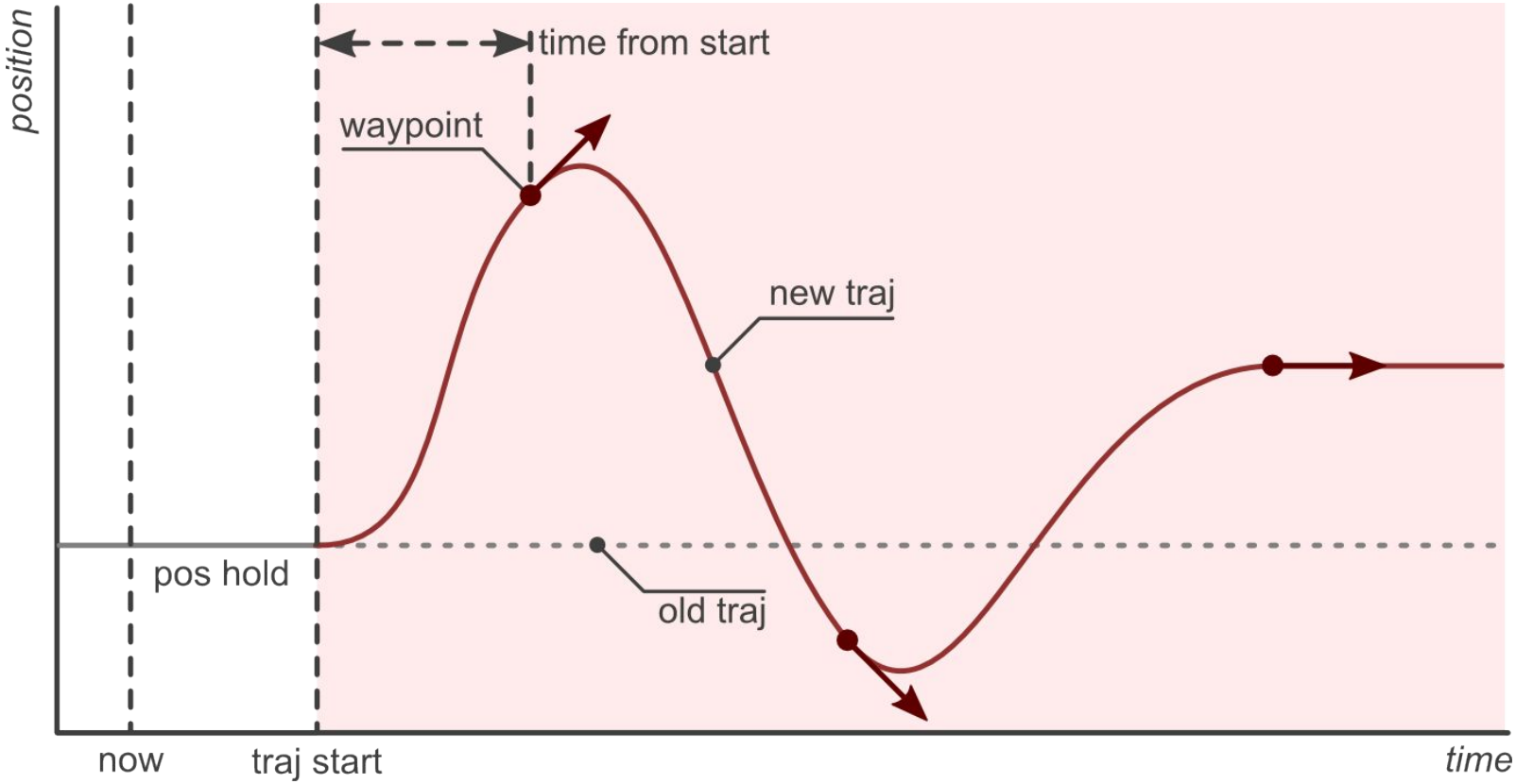
```
float64[] positions
float64[] velocities
float64[] accelerations
duration time_from_start
```

When sending messages to a controller **all joints in the group** must be specified!

PAL ROBOTICS

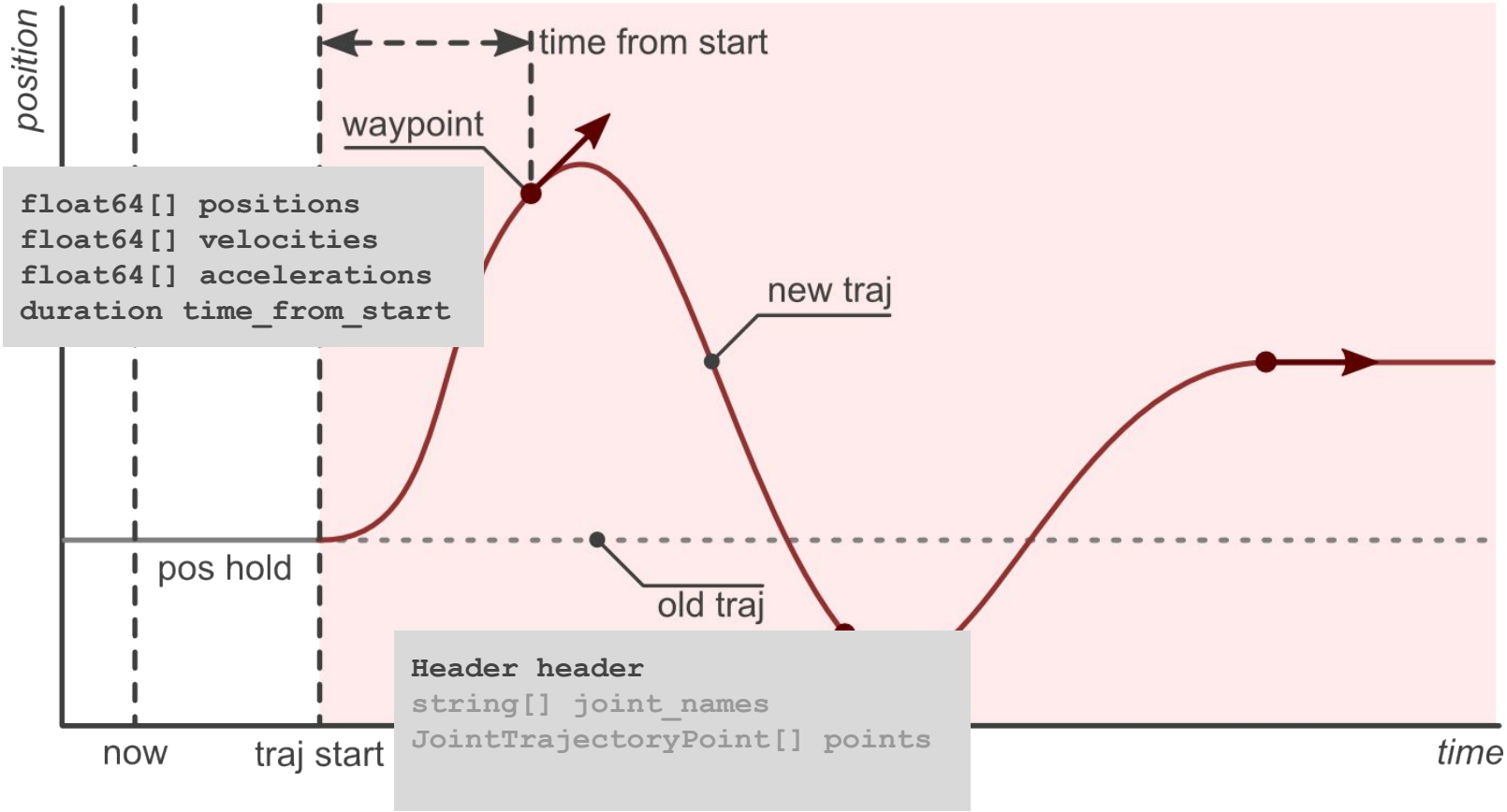
Joint trajectory controller

Trajectory generation and execution



Joint trajectory controller

Trajectory generation and execution



Joint trajectory controller

Sending trajectories

- action interface (prefered)
  - Specifies **trajectory** + (optional) **path and goal tolerances**
  - Allows for **execution monitoring**
  - Goal is **canceled** if tolerances are violated during execution

- topic interface
  - Specifies **trajectory** only
  - No **execution monitoring**, fire and forget
  - Tolerances are **ignored**

PAL ROBOTICS

Joint trajectory controller

Preemption policy

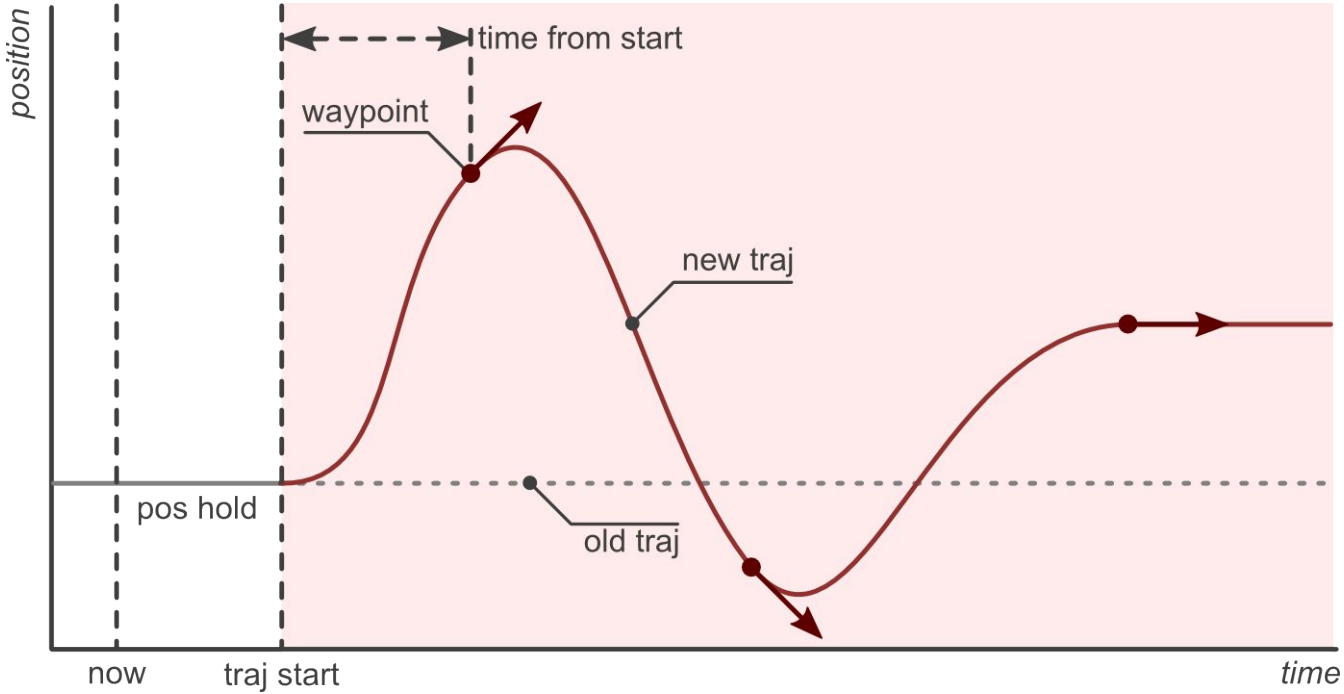
- Only **one action goal** can be active at any moment
- Tolerances are checked **only for the active goal**
- An **empty trajectory** message will stop trajectory execution

Joint trajectory controller

Trajectory replacement

- New trajectory command: doesn't mean discarding the currently running one
- Take useful parts of both and combine them

Effect of trajectory start time: Before new command

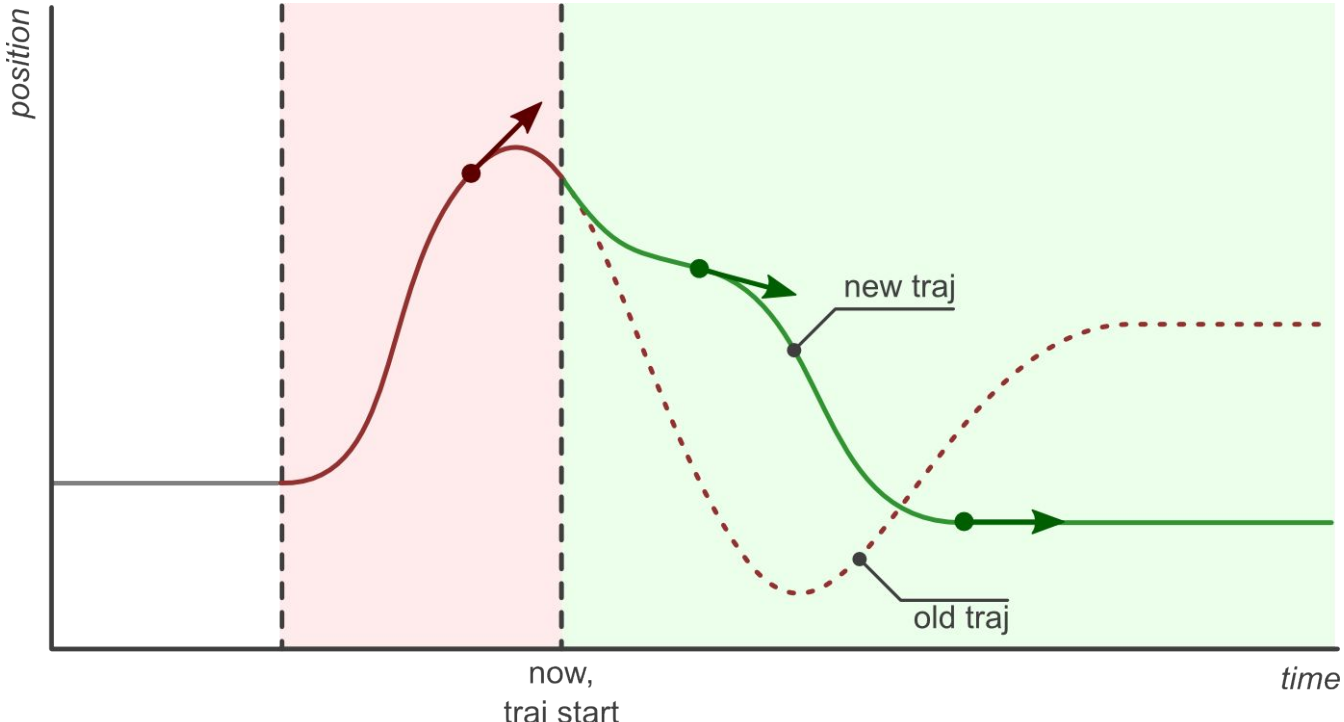


Joint trajectory controller

Trajectory replacement

- New trajectory command: doesn't mean discarding the currently running one
- Take useful parts of both and combine them

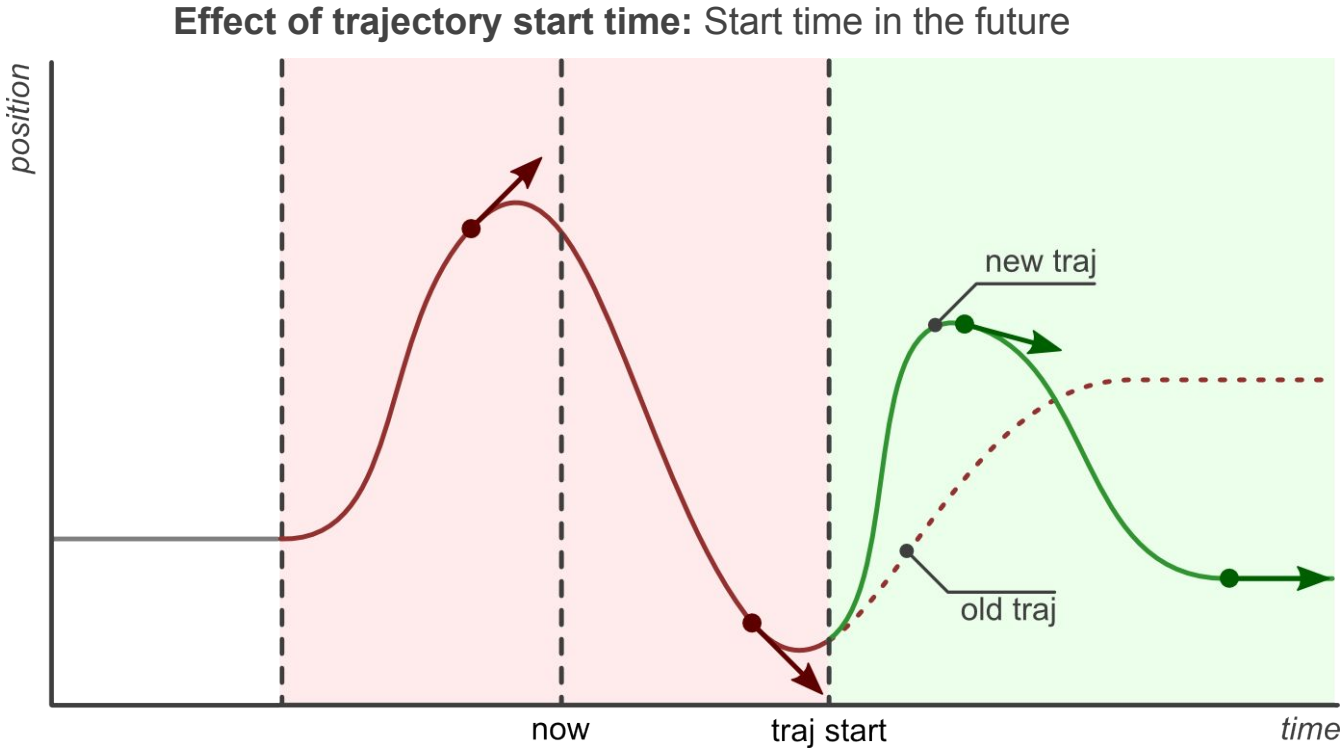
Effect of trajectory start time: Zero start time (start now)



Joint trajectory controller

Trajectory replacement

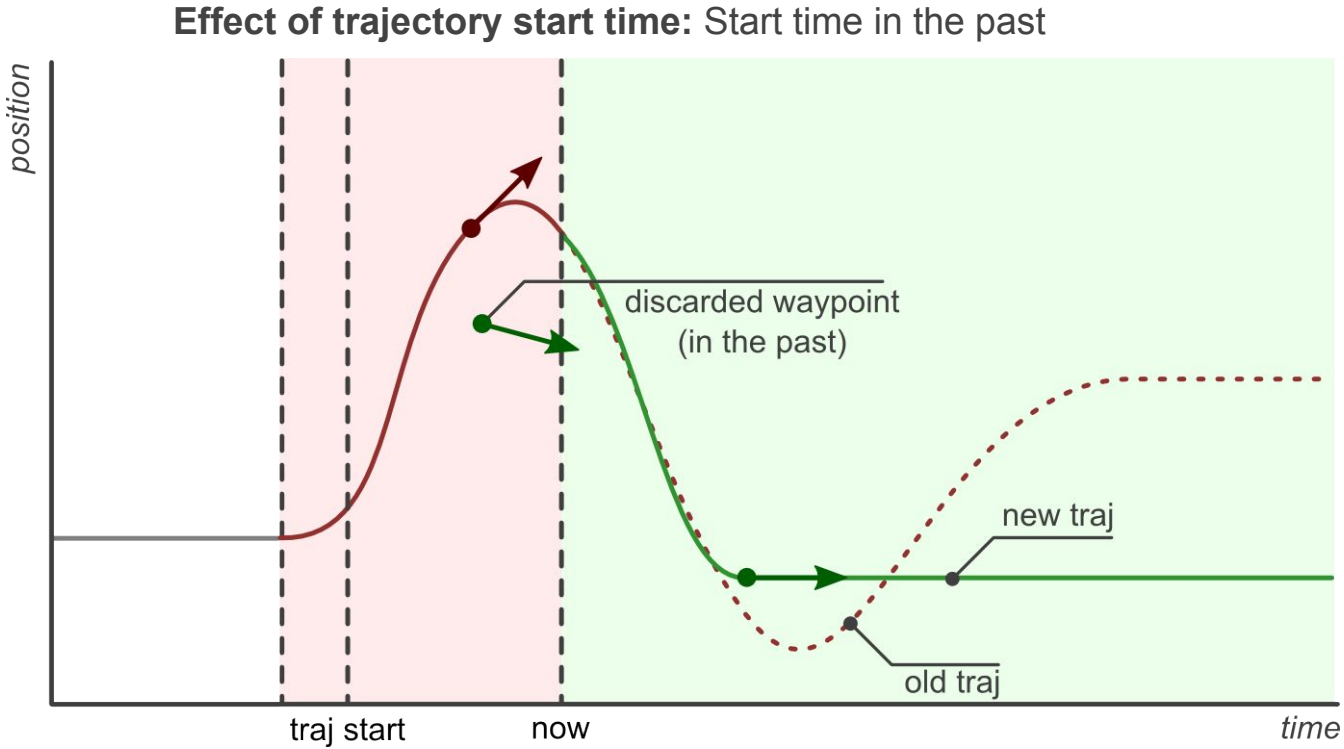
- New trajectory command: doesn't mean discarding the currently running one
- Take useful parts of both and combine them



Joint trajectory controller

Trajectory replacement

- New trajectory command: doesn't mean discarding the currently running one
- Take useful parts of both and combine them



Joint trajectory controller

Demo cheat sheet 1/3:

- In one terminal, **start the controllers**

```
export ROS_MASTER_URI=http://talos-2c:11311
roslaunch talos_controller_configuration full_body_position_controllers.launch
```

- In a second terminal, start an **axclient** pointing to the **head controller**

```
export ROS_MASTER_URI=http://talos-2c:11311
rosrun actionlib axclient.py /head_controller/follow_joint_trajectory
```

- In a third terminal, start an **online visualization** of joint values: desired *pos, vel, acc* and actual *pos* (long command, you can copy/paste it)

```
export ROS_MASTER_URI=http://talos-2c:11311
rqt_plot
/head_controller/state/desired/positions[0],/head_controller/state/actual/positions[0],/head
_controller/state/desired/positions[1],/head_controller/state/actual/positions[1],/head_cont
roller/state/desired/velocities[0],/head_controller/state/desired/velocities[1],/head_contro
ller/state/desired/accelerations[0],/head_controller/state/desired/accelerations[1]
```

Joint trajectory controller

Demo cheat sheet 2/3:

- Copy the following text to the **Goal** textbox of **axclient**, click **SEND_GOAL**

```
trajectory:
  header:
        seq: 0
        stamp:
        secs: 0
        nsecs: 0
        frame_id: ''
  joint_names: ['head_1_joint','head_2_joint']
  points:
        -
        positions: [0.5, 0.2]
        velocities: [0.0, 0.0]
        accelerations: []
        time_from_start:
        secs: 2
        nsecs: 0
        -
        positions: [0.0, 0.0]
        velocities: [0.0, 0.0]
        accelerations: []
        time_from_start:
        secs: 6
        nsecs: 0
path_tolerance: []
goal_tolerance: []
goal_time_tolerance:
  secs: 0.5
  nsecs: 0
```

TALOS T.S.

Joint trajectory controller

Demo cheat sheet 3/3:

- Monitor execution in the **rqt_plot** window