Autonomous and Mobile Robotics

Prof. Giuseppe Oriolo

Whole-Body Motion Planning for Humanoid Robots

(slides prepared by Paolo Ferrari)

DIPARTIMENTO DI INGEGNERIA INFORMATICA AUTOMATICA E GESTIONALE ANTONIO RUBERTI

SAPIENZA UNIVERSITÀ DI ROMA

introduction

- task-constrained motion planning: find feasible, collision-free motions for a humanoid that is assigned a certain task whose execution may require stepping
- it is challenging since humanoids:
    - have a high number of degrees of freedom
    - are not free-flying systems
    - must maintain equilibrium at all times

introduction

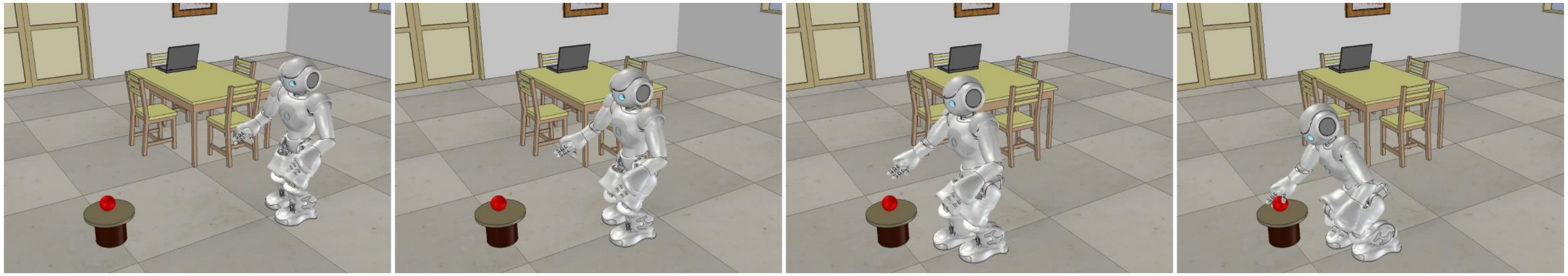
- literature approaches
  - separate locomotion from task execution (e.g., Burget et al., 2013 and Hauser and Ng-Thow-Hing, 2011)
  - compute a collision-free, statically stable path for a free-flying humanoid base, and then approximate it with a dynamically stable walking motion (e.g., Dalibar et al., 2013)
  - achieve acyclic locomotion and task execution through whole-body contact planning (e.g, Bouyarmane and Kheddar, 2012)

- our approach
  - does not separate locomotion from task execution, taking advantage of the whole-body structure of the humanoid
  - walking emerges naturally from the solution of the planning problem

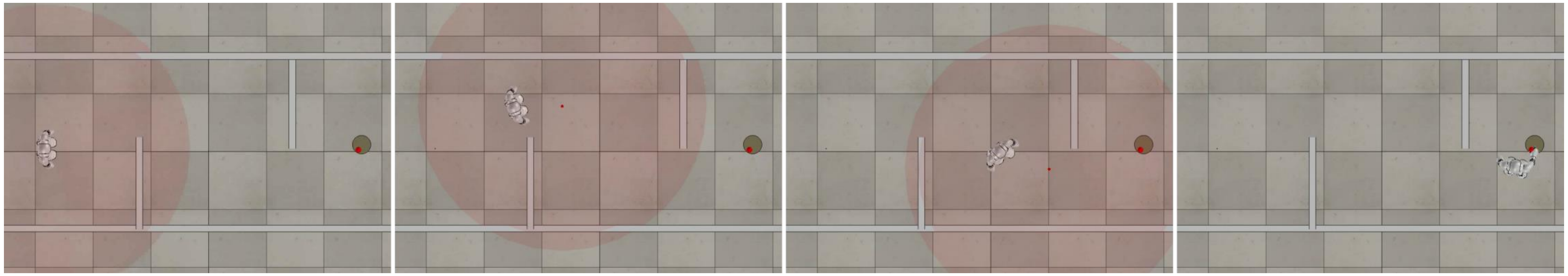
topics

- Planning based on CoM movement primitives

- Planning for loco-manipulation tasks



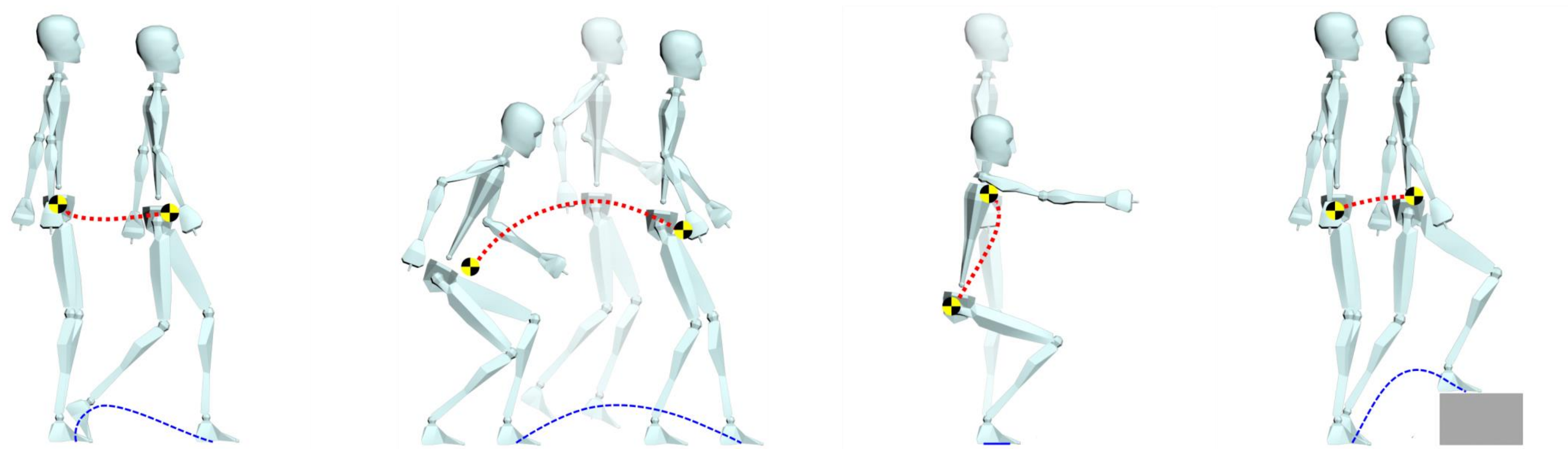
- Anytime planning/replanning



Planning based on CoM movement primitives

CoM movement primitives

- main idea: build a solution by concatenating various feasible whole-body motions that realize CoM movement primitives contained in a precomputed catalogue



- a CoM movement primitive
  - represents an elementary humanoid motion (e.g., walking, jumping)
  - describes the history $\boldsymbol{u}_{\mathrm{CoM}}(t)$ of the CoM pose displacement
  - is characterized by a duration T, a CoM reference trajectory $\boldsymbol{z}^*_{\mathrm{CoM}}(t)$ and a swing foot reference trajectory $\boldsymbol{z}^*_{\mathrm{swg}}(t)$
  - does not specify a whole-body joint motion

humanoid motion model

- robot configuration

$$\boldsymbol{q} = \begin{pmatrix} \boldsymbol{q}_{\mathrm{CoM}} \\ \boldsymbol{q}_{\mathrm{jnt}} \end{pmatrix}$$

$\boldsymbol{q}_{\mathrm{CoM}}$ is the pose of the CoM frame
$\boldsymbol{q}_{\mathrm{jnt}}$ is the n-vector of joint angles

- hybrid motion model

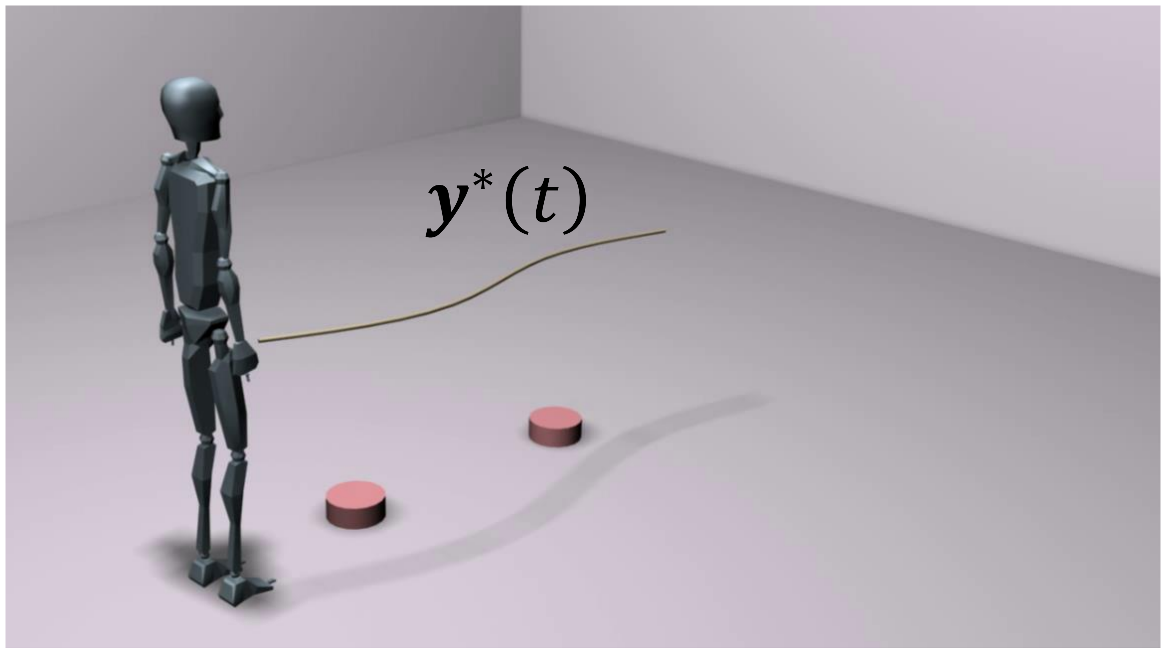
$$\begin{aligned} \boldsymbol{q}_{\mathrm{CoM}}(t) &= \boldsymbol{q}_{\mathrm{CoM}}(t_k) + \boldsymbol{A}\big(\boldsymbol{q}_{\mathrm{CoM}}(t_k)\big)\boldsymbol{u}_{\mathrm{CoM}}(t) \\ \dot{\boldsymbol{q}}_{\mathrm{jnt}}(t) &= \boldsymbol{v}_{\mathrm{jnt}}(t) \end{aligned} \qquad t \in [t_k, t_k + T_k]$$

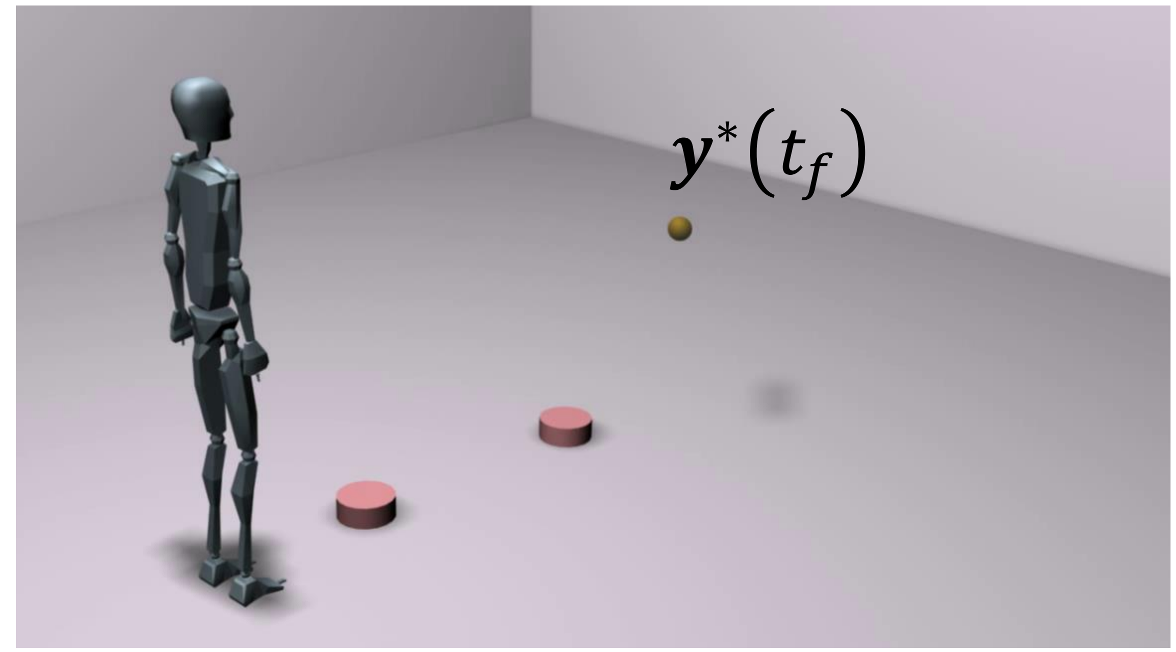
where

$\boldsymbol{q}_{\mathrm{CoM}}(t_k)$ is the pose of the CoM frame at t_k
$\boldsymbol{A}\big(\boldsymbol{q}_{\mathrm{CoM}}(t_k)\big)$ is the transformation matrix between the CoM frame at t_k and the world frame
$\boldsymbol{u}_{\mathrm{CoM}}(t)$ is the CoM displacement at t relative to t_k
$\boldsymbol{v}_{\mathrm{jnt}}(t)$ is the vector of joint velocity commands

task-constrained planning

- task definition: assigned trajectory $\boldsymbol{y}^*(t)$ (or a geometric path $\boldsymbol{y}^*(s)$ or a single point $\boldsymbol{y}^*(t_f)$) for a specific point of the humanoid

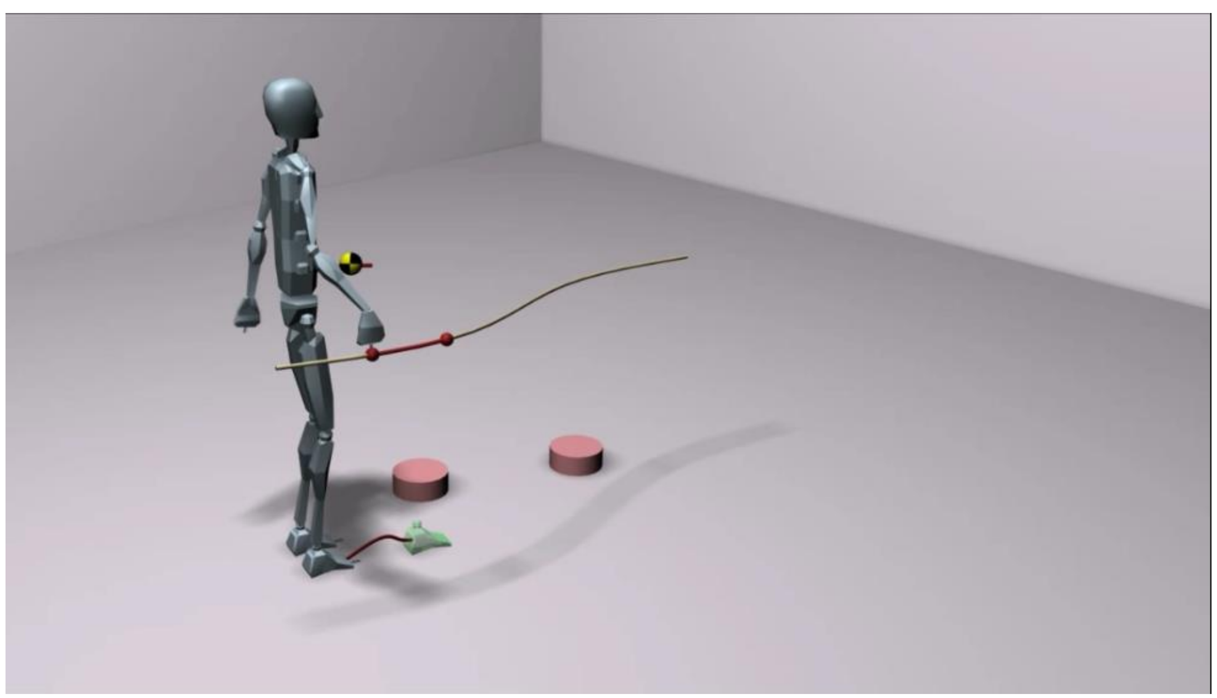


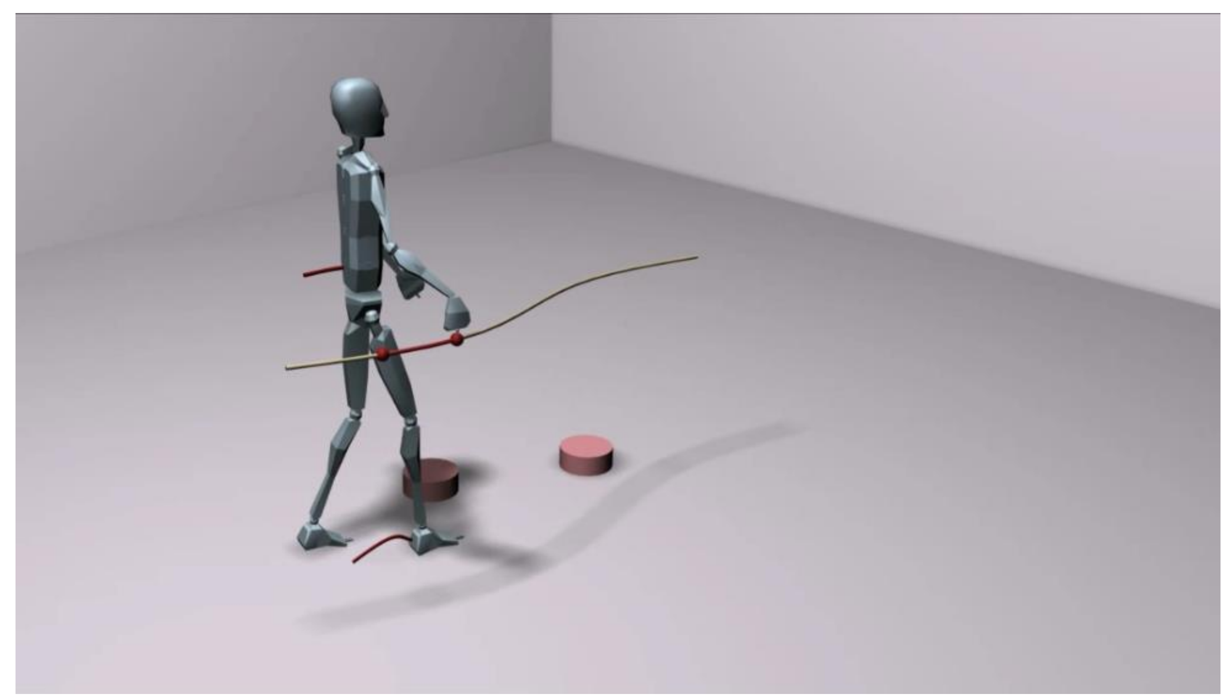


- formal definition of the solution: a whole-body motion in terms of a joint trajectory $\boldsymbol{q}_{\mathrm{jnt}}(t)$ such that
  - the assigned task is exponentially realized
  - collisions with workspace obstacles and self-collisions are avoided
  - position and velocity limits on the robot joints are satisfied
  - the robot is in equilibrium at all times

motion generation

- the planner works iteratively, by repeated calls to a motion generator
- the motion generator is invoked to produce a feasible, collision-free elementary motion that realizes a portion of the assigned task
- it consists of two procedures
  1. CoM movement selection: choose a particular CoM movement from the set of primitives
  2. joint motion generation: compute the corresponding joint motions





CoM movement selection

- it is invoked from the current configuration $\boldsymbol{q}_k$ at time t_k
- a CoM movement is selected by picking one primitive from the catalogue

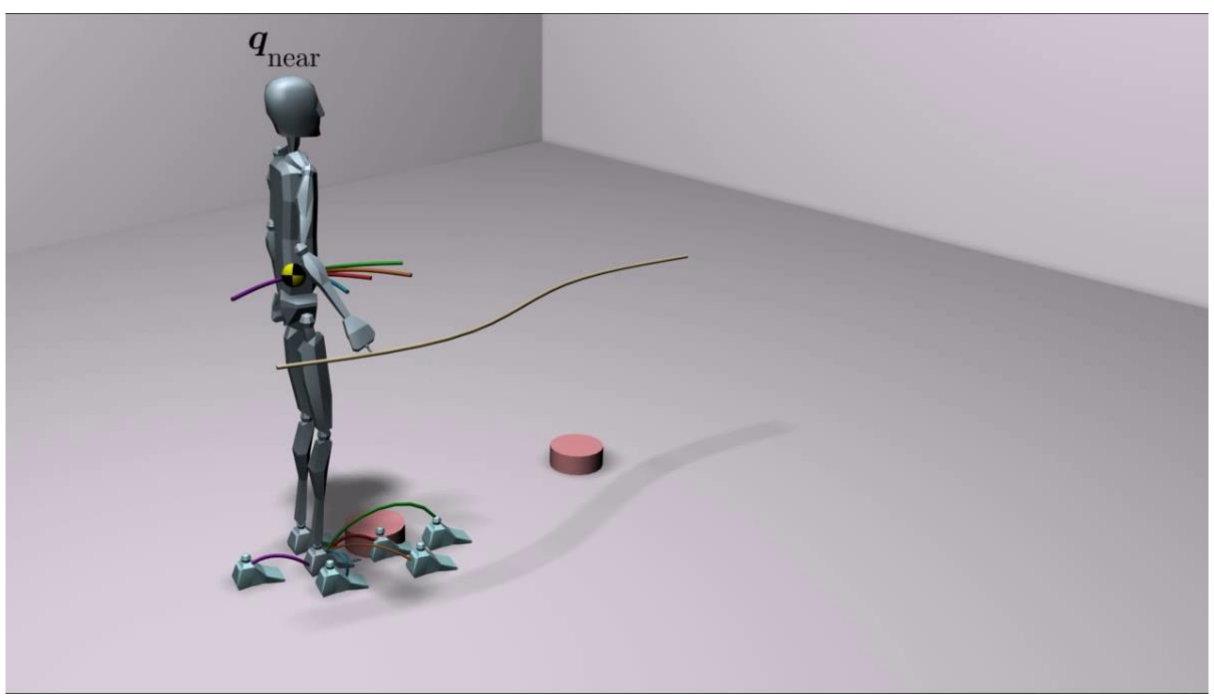
$$U = \{U^S_{\text{CoM}} \cup U^D_{\text{CoM}} \cup \text{free\_CoM}\}$$

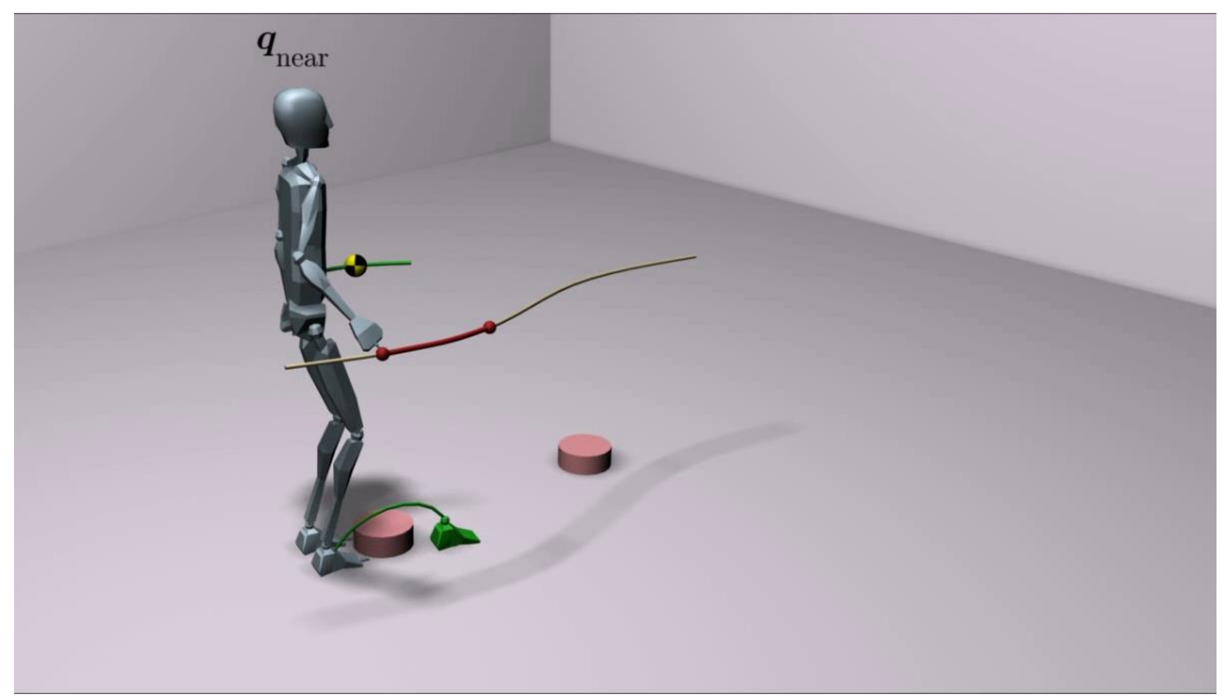
where

U^S_{CoM} is the subset of static steps

U^D_{CoM} is the subset of dynamic steps

free_CoM is a non-stepping, stretchable primitive





joint motion generation

- it produces joint motion from $\boldsymbol{q}_k$ at t_k that realizes the selected CoM primitive and the portion of the assigned task in $[t_k, t_{k+1}]$

$$\boldsymbol{v}_{\text{jnt}} = \boldsymbol{J}_a^{\#}(\boldsymbol{q})(\dot{\boldsymbol{y}}_a^* + \boldsymbol{K}\boldsymbol{e}) + \left(\boldsymbol{I} - \boldsymbol{J}_a^{\#}(\boldsymbol{q})\boldsymbol{J}_a(\boldsymbol{q})\right)\boldsymbol{\omega}$$

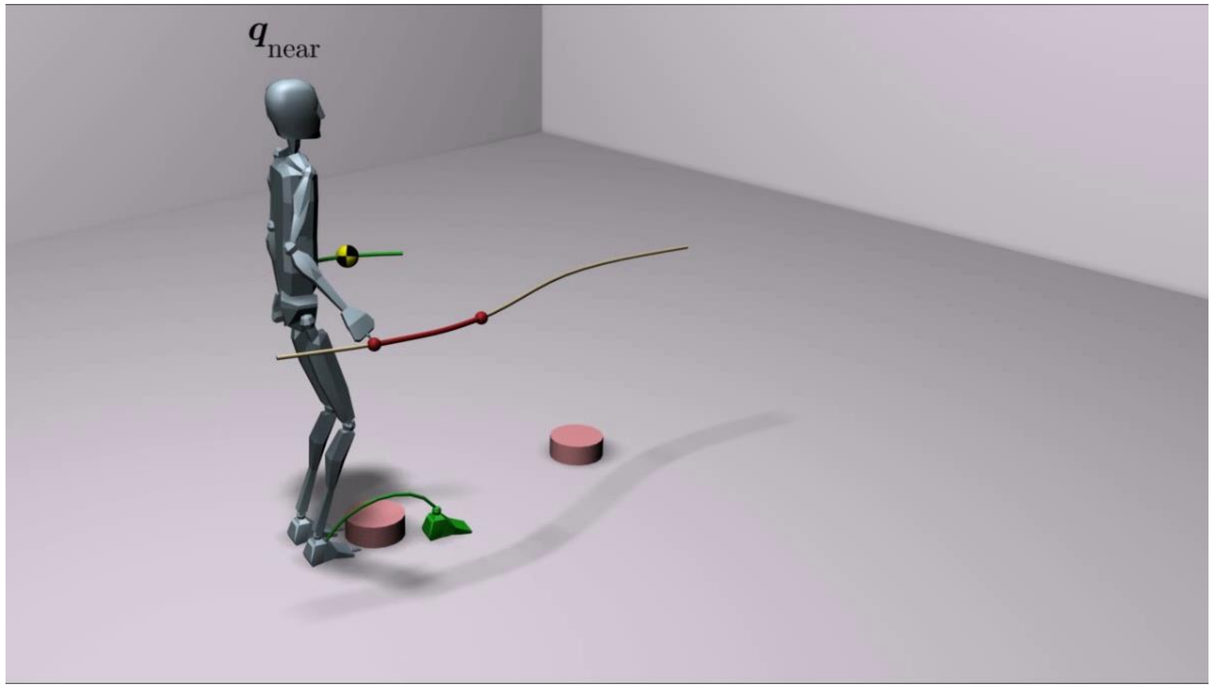
where

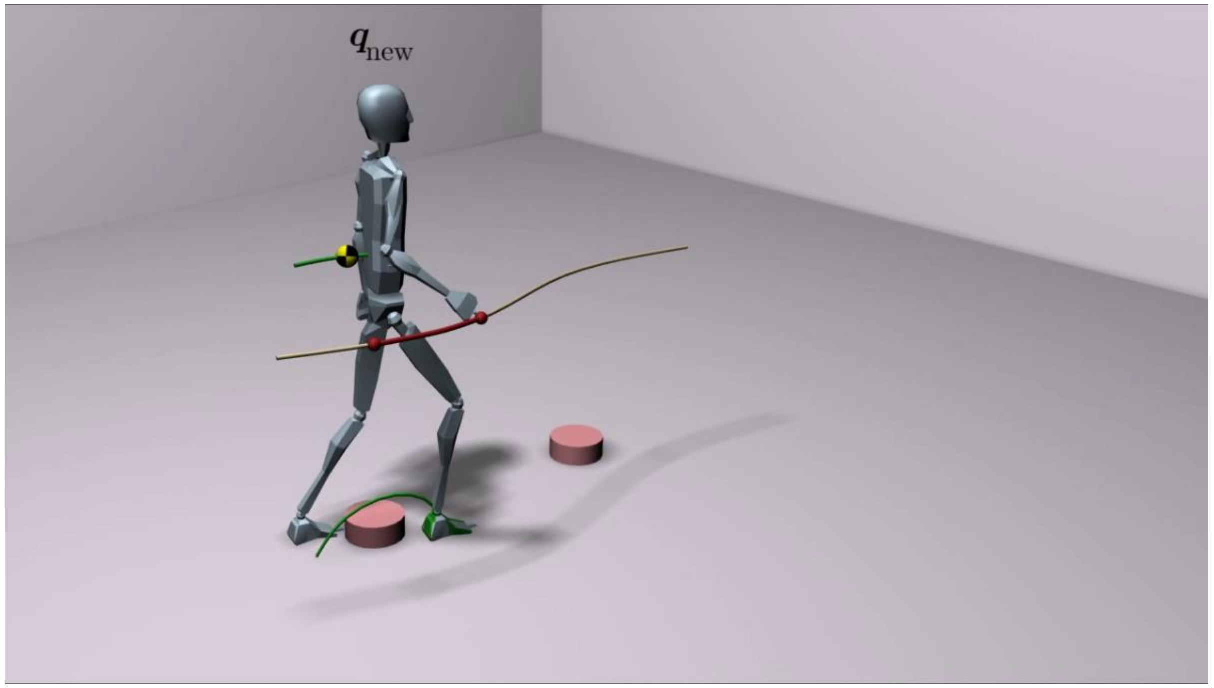
$\boldsymbol{y}_a = \left(\boldsymbol{y}^T \quad \boldsymbol{z}_{\text{CoM}}^T \quad \boldsymbol{z}_{\text{swg}}^T\right)^T$: augmented task vector

$\boldsymbol{J}_a(\boldsymbol{q})$: Jacobian matrix of $\boldsymbol{y}_a$ wrt $\boldsymbol{q}$

$\boldsymbol{e} = \boldsymbol{y}_a^* - \boldsymbol{y}_a$: augmented task error

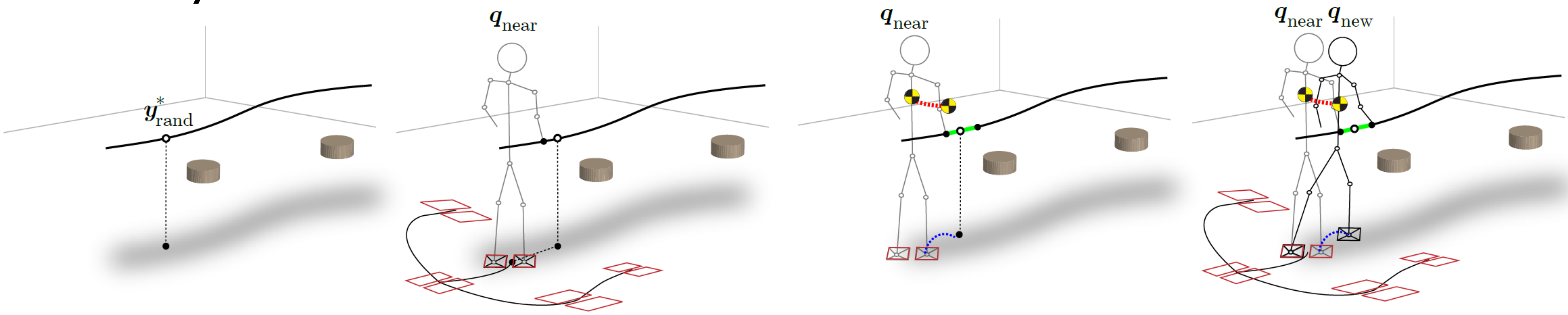
$\boldsymbol{\omega}$: arbitrary n-vector





the planner

iterratively build a tree



1. choose a random sample $\boldsymbol{y}^*_{\mathrm{rand}}$ on the assigned task trajectory
2. extract a configuration $\boldsymbol{q}_{\mathrm{near}}$ with probability proportional to a task compatibility function $\gamma(\cdot, \boldsymbol{y}^*_{\mathrm{rand}})$
3. select a random CoM movement and extract the portion of the task trajectory to be executed
4. generate the joint motion to realize the CoM movement and the portion of the task
5. if the motion is feasible and collision-free, add the final configuration $\boldsymbol{q}_{\mathrm{new}}$ to the tree

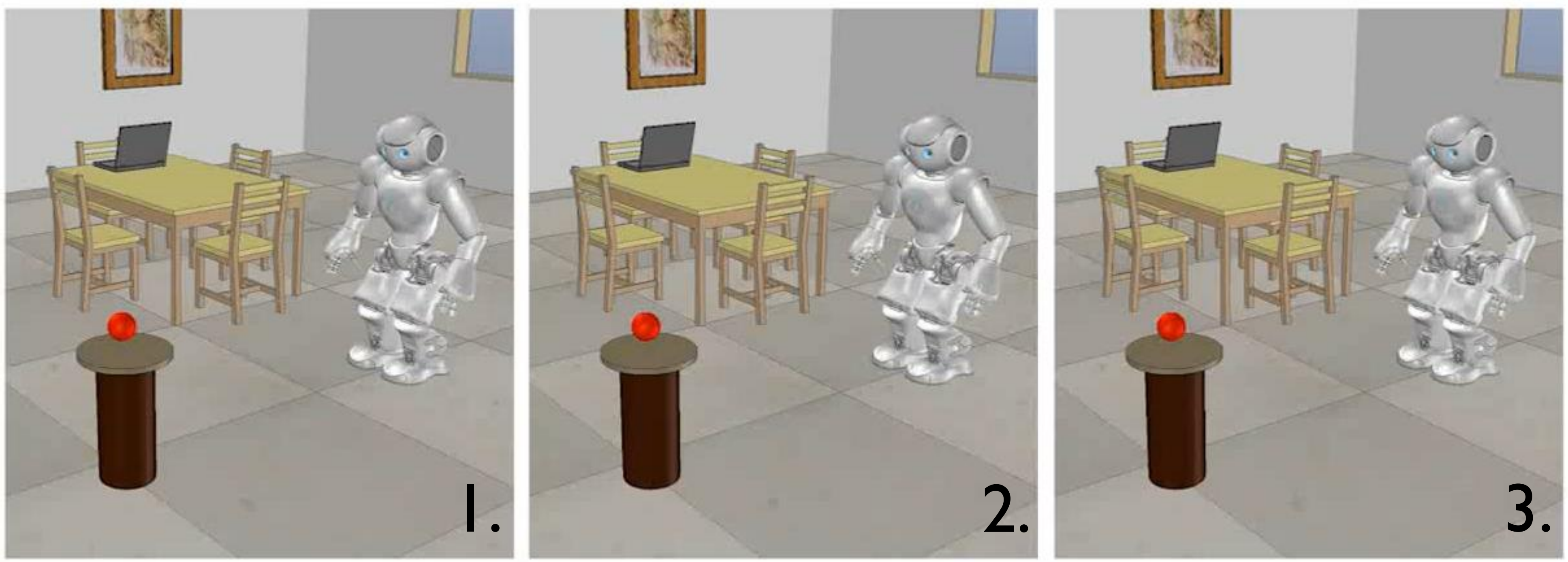
planning experiments: video

https://youtu.be/78xujXc29ig

Planning for loco-manipulation tasks

motivation

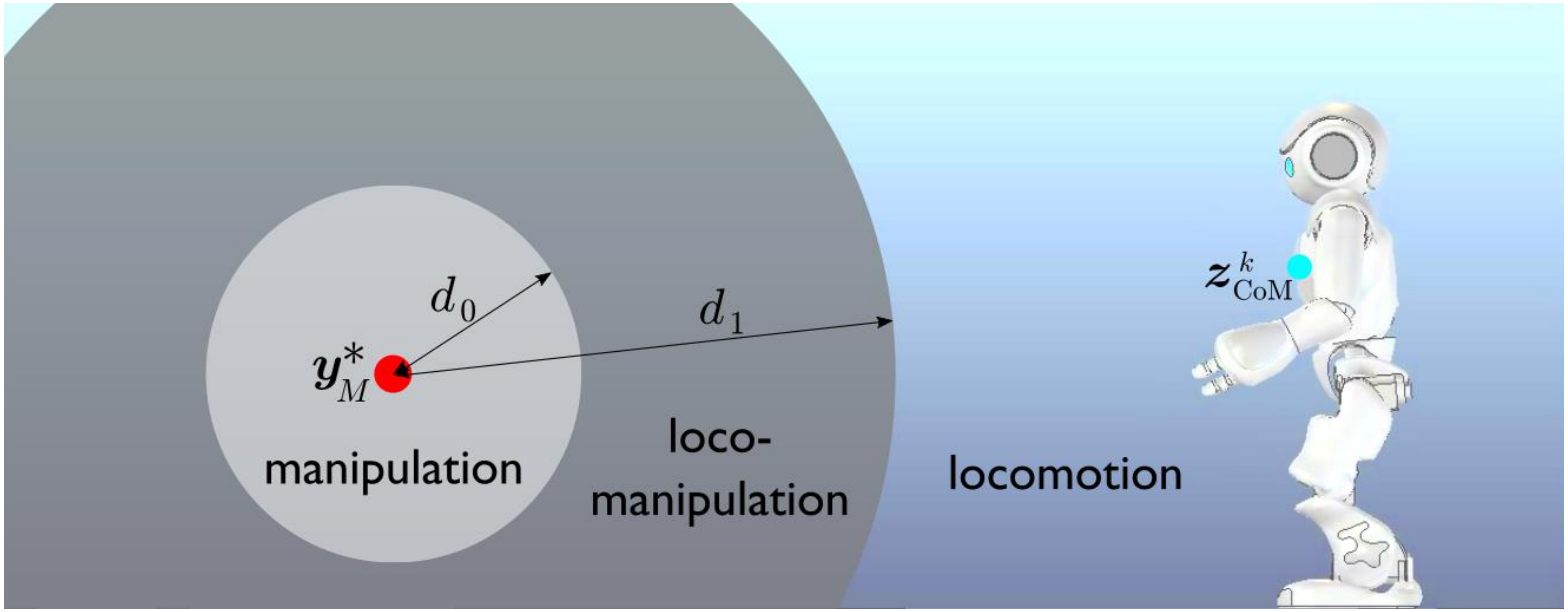
- plan a fluid, natural reaching motion to fulfil a loco-manipulation task
- loco-manipulation task: manipulation task that implicitly requires locomotion, i.e., bring one hand to a certain position (set-point $\boldsymbol{y}_M^*$)



1. manipulation task always active
2. manipulation task activated close to the set-point
3. synchronization of locomotion and manipulation tasks

operation zones

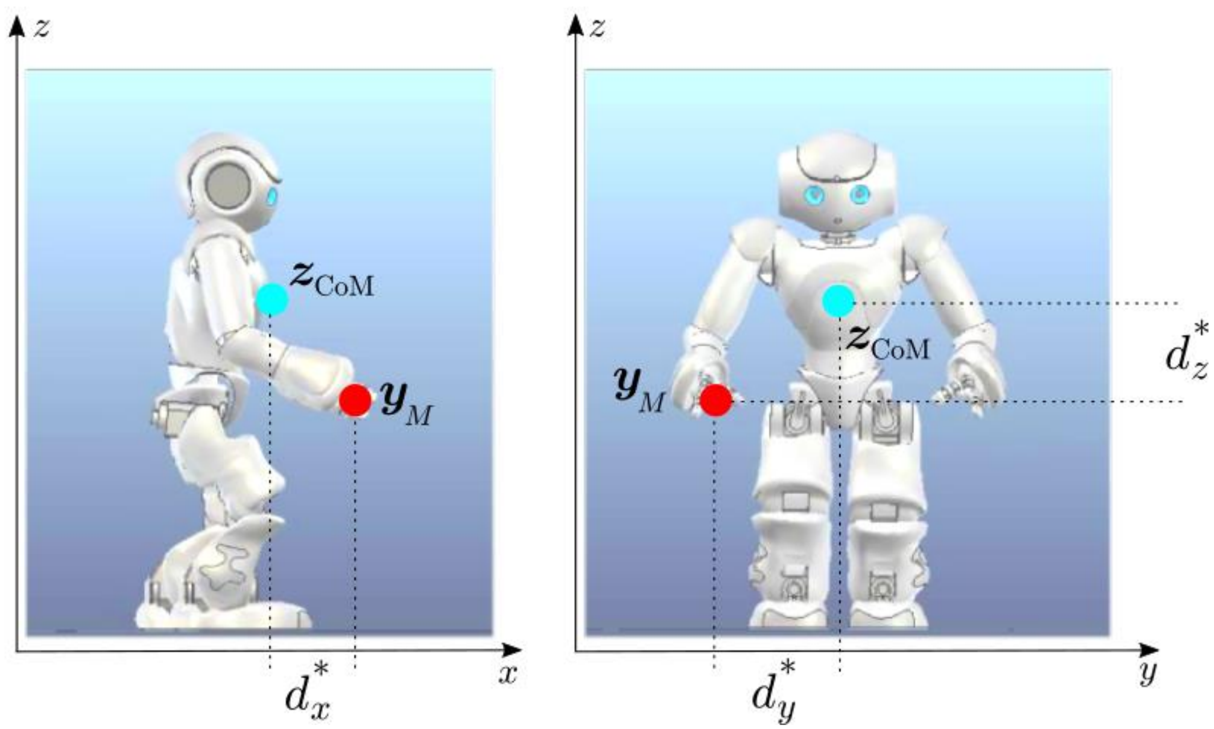
depending on the distance between the CoM and the set-point $\boldsymbol{y}_M^*$

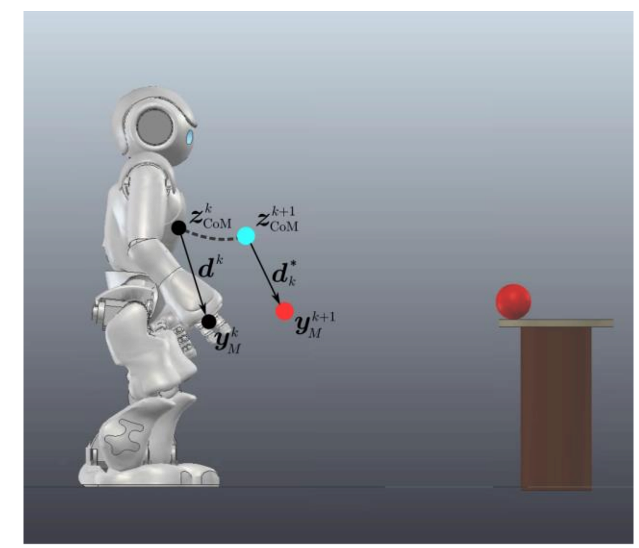


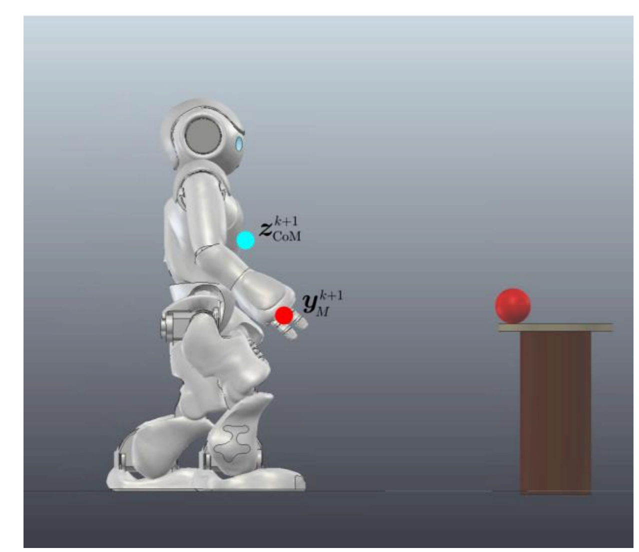
- locomotion zone: only locomotion task is considered
- loco-manipulation zone: both tasks are considered, with a local manipulation task driven by locomotion
- manipulation zone: only manipulation task is considered

synchronizing locomotion and manipulation

- in the loco-manipulation zone, synchronize the motion of the hand with the motion of the CoM
- progressively approach the desired set-point $\boldsymbol{y}_M^*$
- local set-point definition $\boldsymbol{y}_M^{k+1} = \boldsymbol{z}_{\mathrm{CoM}}^{k+1} + \boldsymbol{d}^k + \boldsymbol{K}_d\left(\boldsymbol{d}_k^* - \boldsymbol{d}^k\right)$







$$\boldsymbol{d}_k^* = \alpha \boldsymbol{d}^* + (1-\alpha)\left(\boldsymbol{y}_M^* - \boldsymbol{z}_{\mathrm{CoM}}^k\right)$$

$$\alpha = 1 \Bigg/ \left(1 + exp\left(\beta_1 \frac{\left\|\boldsymbol{z}_{\mathrm{CoM}}^k - \boldsymbol{y}_M^*\right\| - d_0}{d_1 - d_0} + \beta_2\right)\right)$$

the planner

iteratively build a tree

1. extract a vertex $\boldsymbol{v}_{\text{near}}$ with probability proportional to a task compatibility function $\gamma(\cdot, \boldsymbol{y}_M^*)$
2. invoke the motion generator
   - select a CoM movement using the weights in $\boldsymbol{v}_{\text{near}}$ as probabilities
   - identify the operation zone associated to $\boldsymbol{v}_{\text{near}}$
   - compute the locomotion task
   - define the local manipulation set-point $\boldsymbol{y}_M^{k+1}$
   - generate the joint motion through the task-priority approach

     $$\boldsymbol{v}_{\text{jnt}} = \boldsymbol{J}_L^{\#}\dot{\boldsymbol{y}}_L' + \boldsymbol{P}_L(\boldsymbol{J}_M\boldsymbol{P}_L)^{\#}(\dot{\boldsymbol{y}}_M' - \boldsymbol{J}_M\boldsymbol{J}_L'\dot{\boldsymbol{y}}_L') + \boldsymbol{P}_{LM}\boldsymbol{v}_0$$
3. if the motion is feasible and collision-free, add a new vertex $\boldsymbol{v}_{\text{new}}$ to the tree, otherwise update the weights in $\boldsymbol{v}_{\text{near}}$

planning experiments: video

https://youtu.be/KG8NR2aKC0M

Anytime planning/replanning

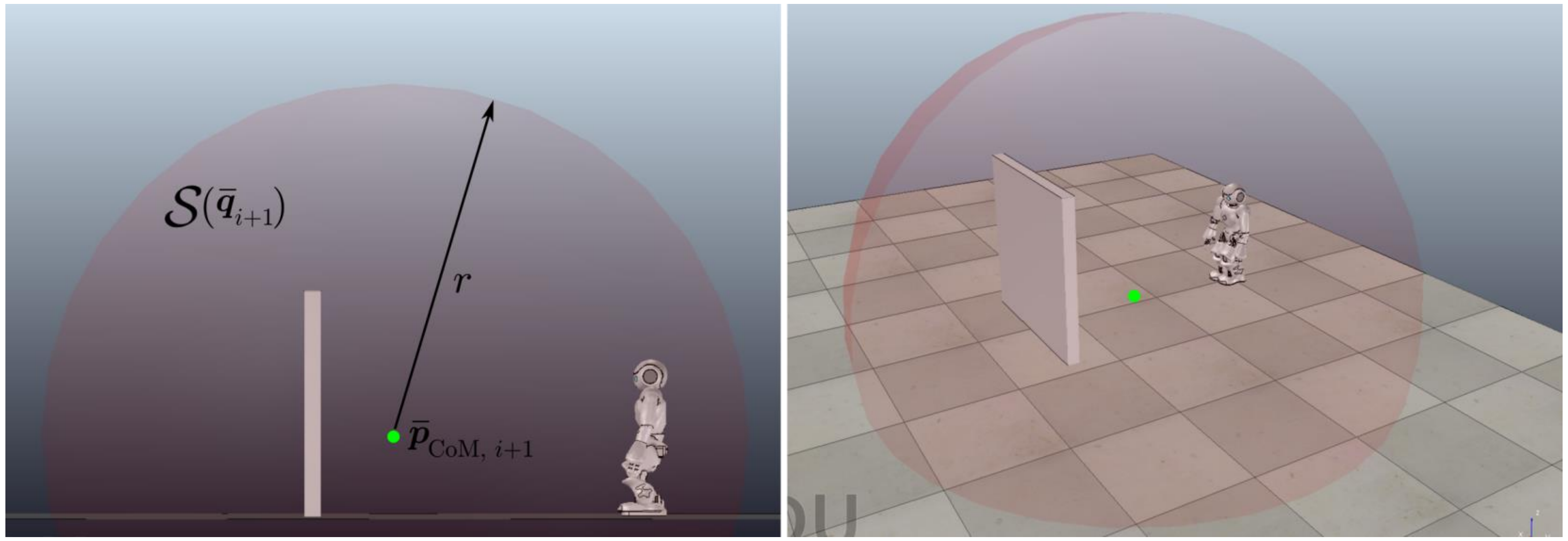
overview

- motivation
  - computing a complete solution for the planning problem can be computationally expensive, requiring high planning times
  - the robot is forced to wait for the solution before being able to start moving

- our approach
  - does not compute (off-line) a complete solution
  - simultaneously performs planning and execution of local plans
  - the complete motion is constituted by a sequence of on-line computed partial solutions
  - naturally extends to the case of sensor-based planning

anytime planning/replanning scheme

- interleave planning and execution phases; i.e., at the i-th generic iteration simultaneously:
  - execute the current local plan
  - generate a novel local plan, within a given time budget, for the next execution phase
- during the i-th planning phase compute a local plan, i.e., a whole-body motion that
  - starts at the final configuration $\bar{q}_{i+1}$ of the simultaneously executed local plan
  - is feasible within a limited planning zone $S(\bar{q}_{i+1})$



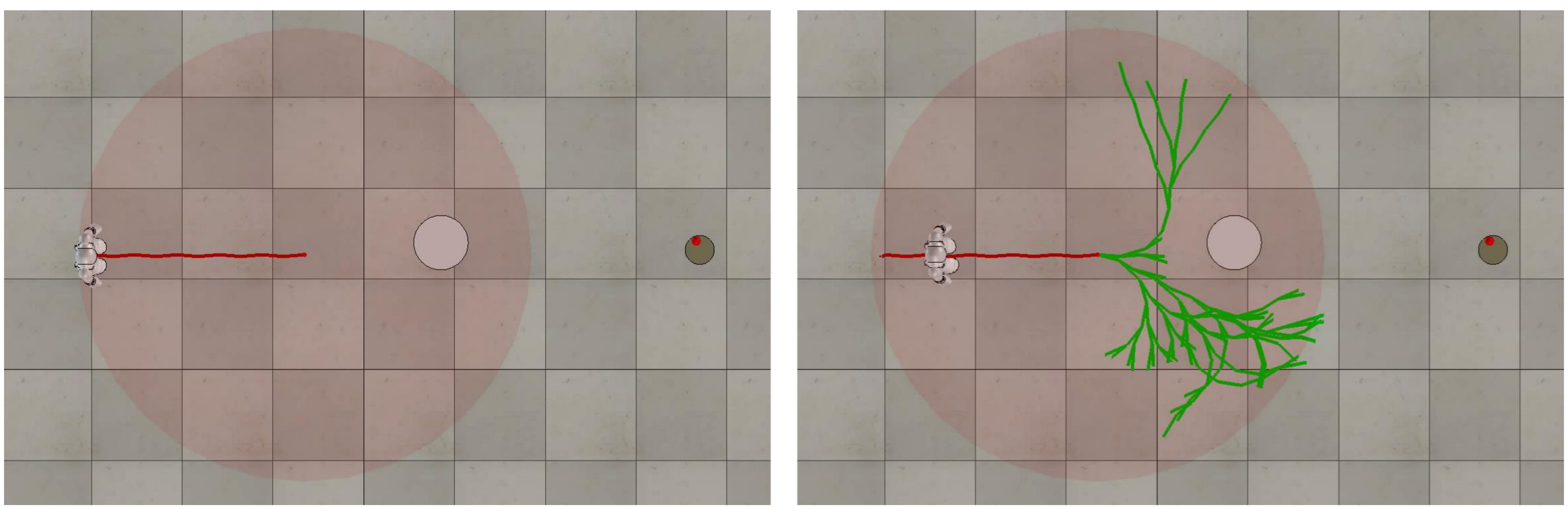
local motion planner

- allowed to run for a given time budget ΔT_P
- randomized CoM movement primitives-based planner
- builds a tree in configuration-time space
- works in two stages
  - lazy stage: quickly expands the tree, checking collisions only at vertexes
  - validation stage: validate the local plan by generating the corresponding whole-body motion and checking collisions both at vertexes and along edges
- time budget ΔT_P is split between the two stages (ΔT_P^L and ΔT_P^V)

lazy stage

iteratively (until ΔT_P^L runs out)

1. choose a random sample $\boldsymbol{y}^*_{\text{rand}}$ in the task space
2. extract a vertex $\boldsymbol{v}_{\text{near}}$ according to a metric $d(\cdot, \boldsymbol{y}^*_{\text{rand}})$
3. select a CoM movement using the weights in $\boldsymbol{v}_{\text{near}}$ as probabilities
4. compute $\boldsymbol{q}^{\text{new}}_{\text{CoM}}$ according to $\boldsymbol{q}^{\text{near}}_{\text{CoM}}$ and the selected CoM movement
5. check collisions on $\boldsymbol{q}^{\text{new}}_{\text{CoM}}$ with obstacles in the planning zone using a simplified occupancy volume for the robot
6. if $\boldsymbol{q}^{\text{new}}_{\text{CoM}}$ is collision-free, add a new vertex $\boldsymbol{v}_{\text{new}}$ to the tree



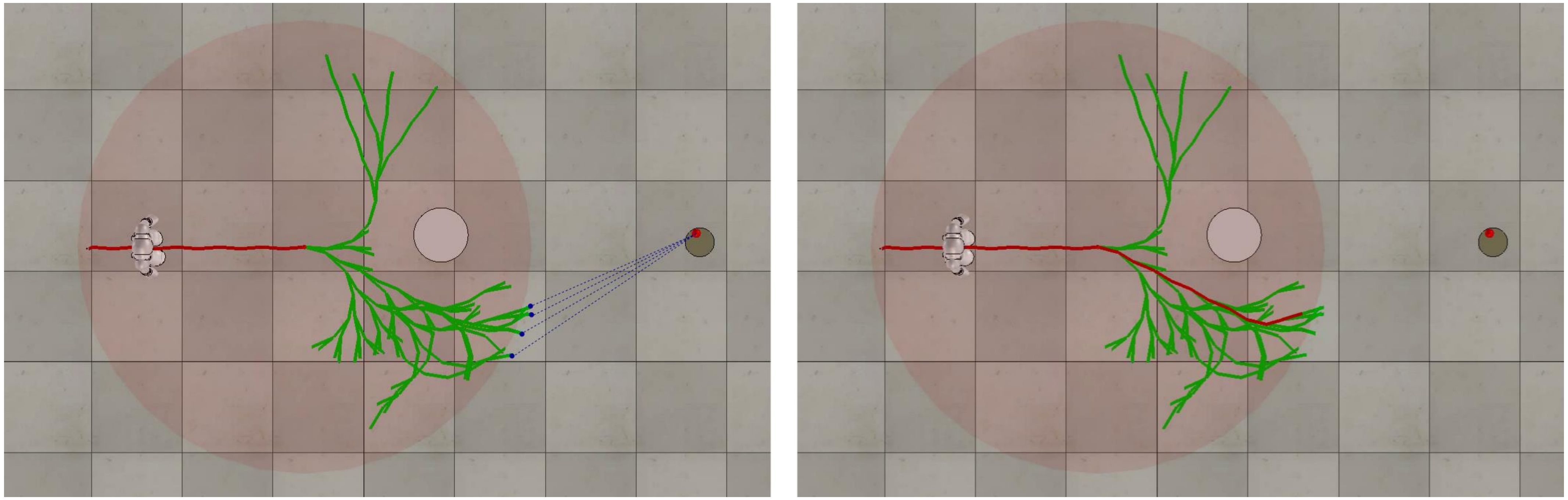
validation stage

candidate local plans: branches of the tree whose ending vertexes are

- outside the planning zone, or
- sufficiently close to $\boldsymbol{y}_M^*$

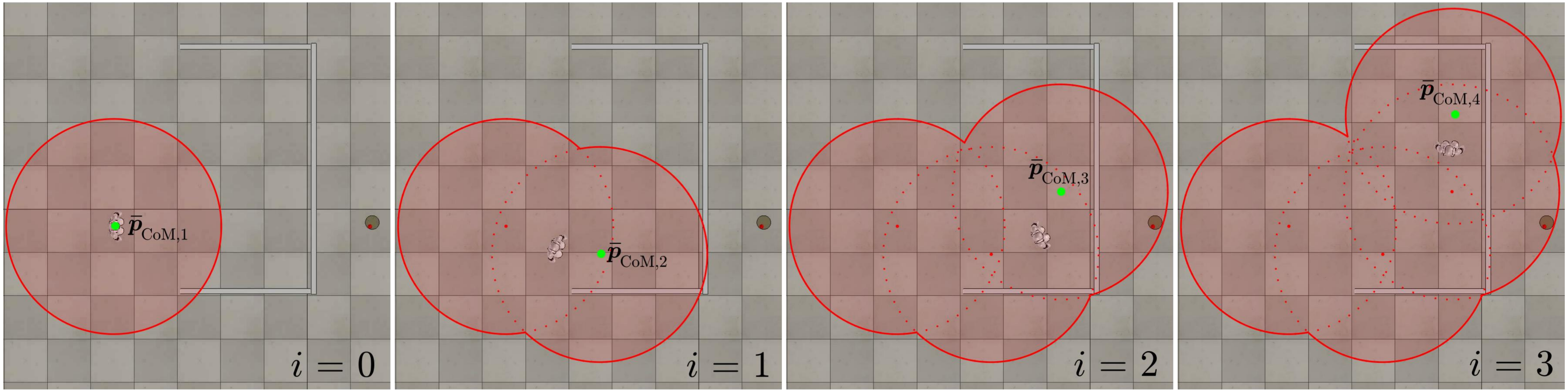
iteratively (until ΔT_P^V runs out)

1. choose the best candidate local plan
2. generate the whole-body motion of all edges along the plan
3. check collisions on both vertexes and edges with obstacles in the planning zone using the actual volume occupancy for the robot

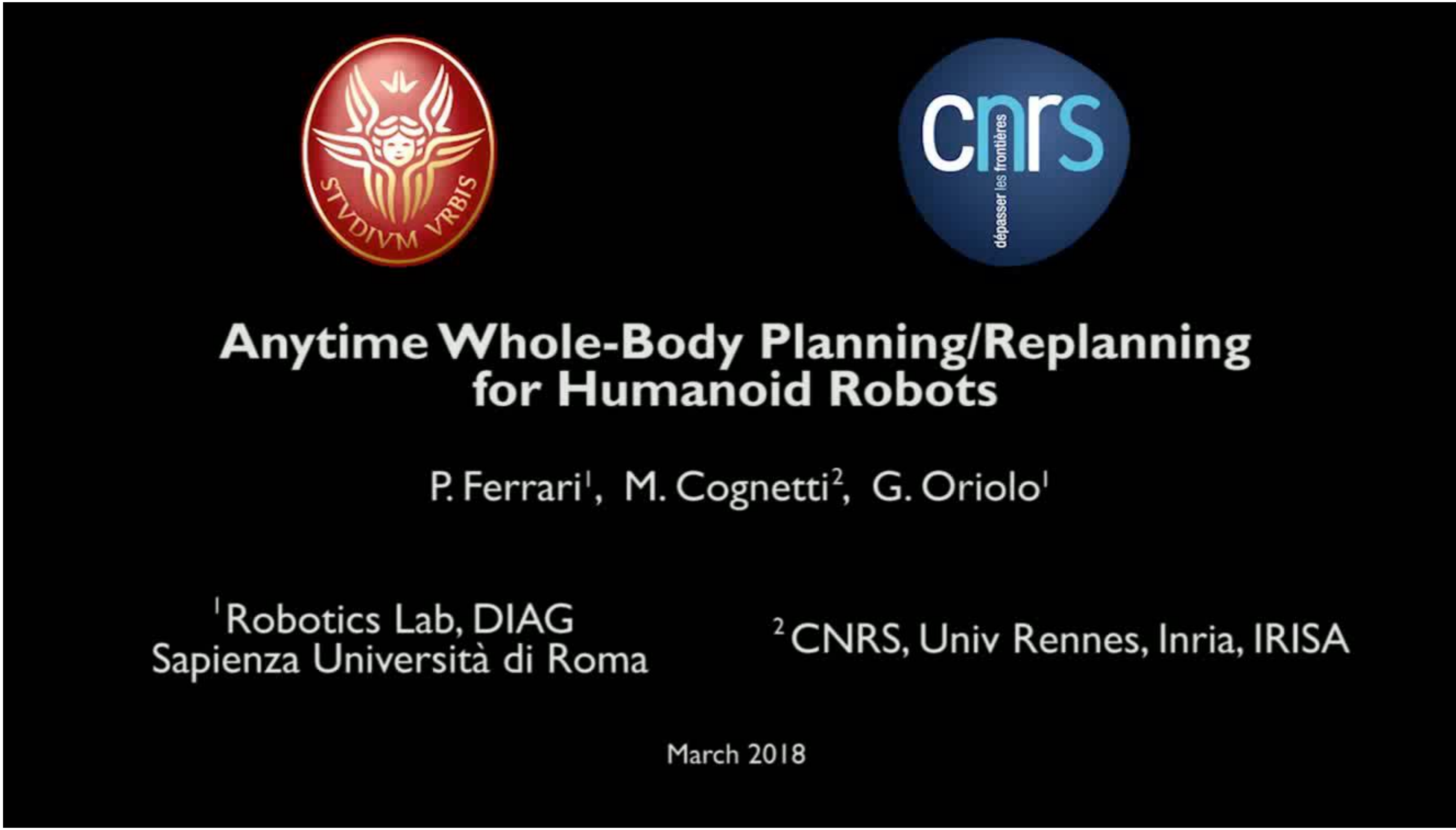


deadlock management

- the fixed-shape planning zone provides only local information about the environment
- in principle, consecutive calls to the local motion planner might generate movements that force the robot to repeatedly traverse the same regions of the task space (deadlock)
- allow the planner to keep memory of previously considered planning zones (current planning zone = union of all previous planning zones)



planning experiments: video



https://youtu.be/ECdDM-usO-k

references

- *Task-Oriented Whole-Body Planning for Humanoids based on Hybrid Motion Generation* – M. Cognetti, P. Mohammadi, G. Oriolo, M. Vendittelli, IROS 2014
- *Whole-Body Motion Planning for Humanoids based on CoM Movement Primitives* – M. Cognetti, P. Mohammadi, G. Oriolo, Humanoids 2015
- *Whole-Body Planning for Humanoids along Deformable Tasks* – M. Cognetti, V. Fioretti, G. Oriolo, ICRA 2016
- *Humanoid Whole-Body Planning for Loco-Manipulation Tasks* – P. Ferrari, M. Cognetti, G. Oriolo, ICRA 2017
- *Anytime Whole-Body Planning/Replanning for Humanoid Robots* – P. Ferrari, M. Cognetti, G. Oriolo, IROS 2018 (submitted)