Hierarchical Visuomotor Control of Humanoids

Josh Merel, Arun Ahuja, Vu Pham, Saran Tunyasuvunakool, Siqi Liu, Dhruva Tirumala, Nicolas Heess, Greg Wayne

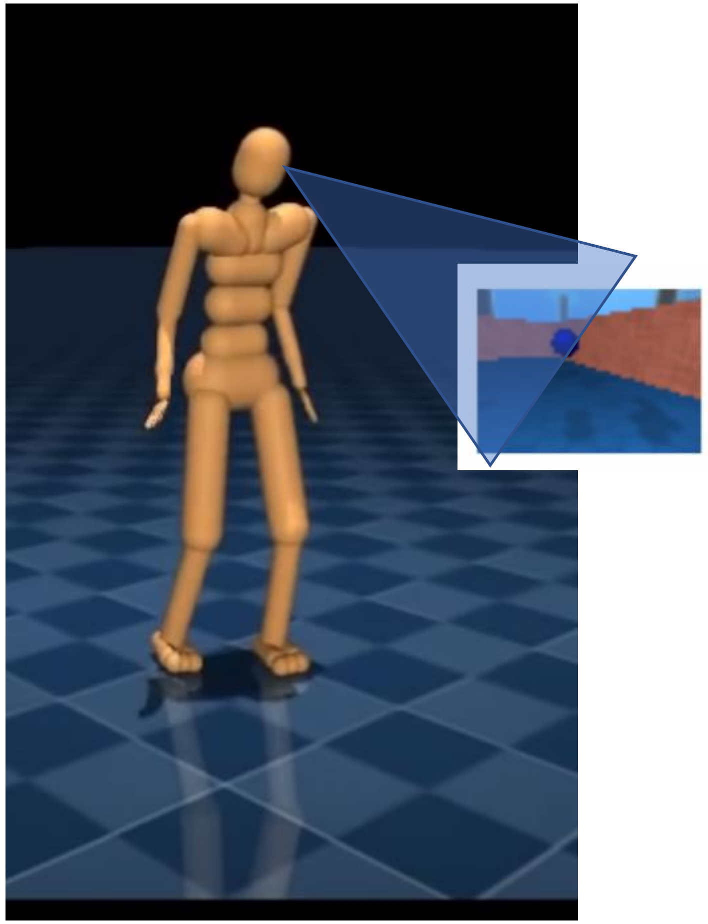
DeepMind London, UK

ICLR 2019 conference paper

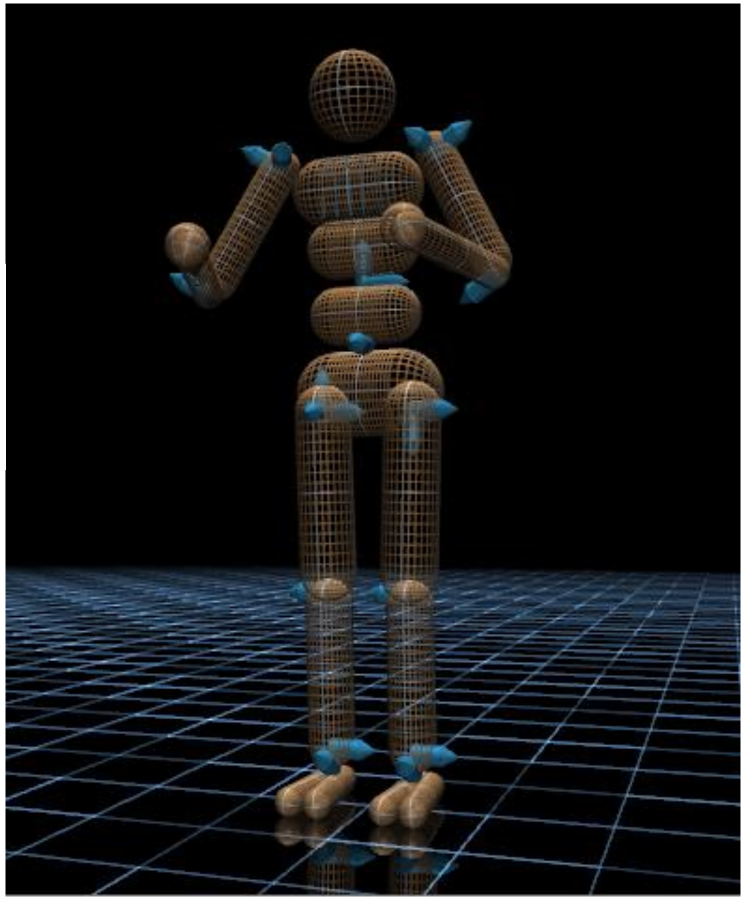
Contents

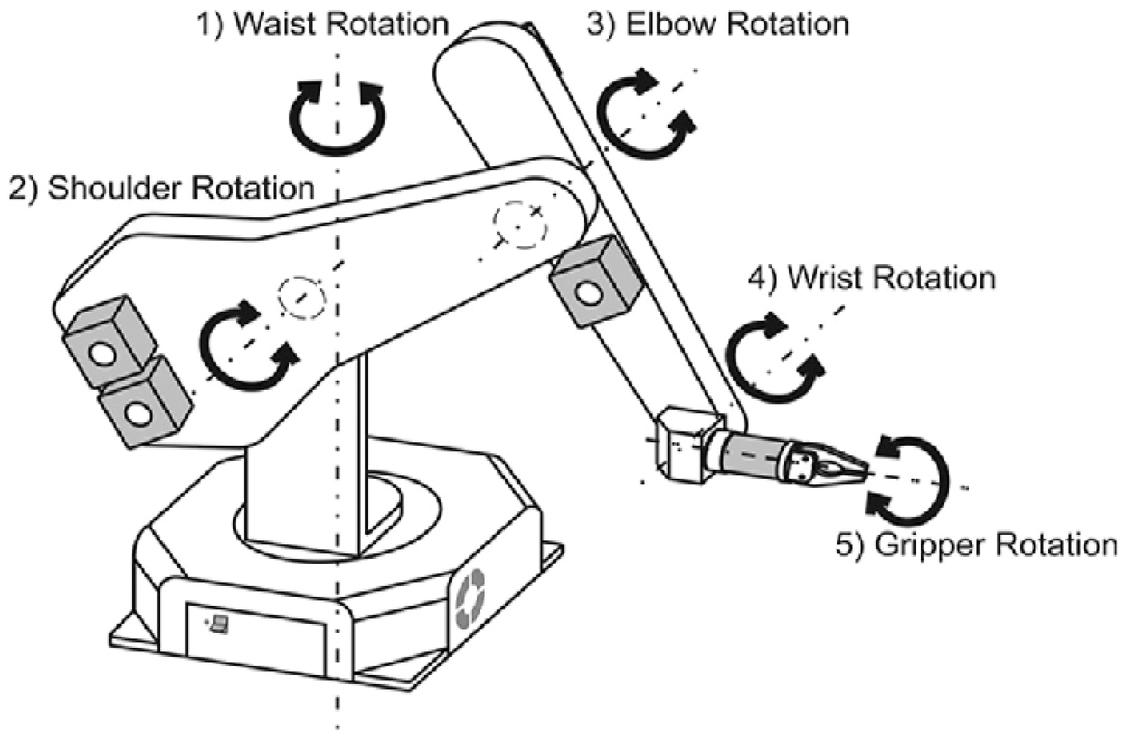
Goal

- We aim to build complex humanoid agents that integrate perception, motor control, and memory.



(56 degree of freedom)





Five degrees of freedom robot arm model

where $S\theta_i = \sin\theta_i$, $C\theta_i = \cos\theta_i$, $S\alpha_i = \sin\alpha_i$, $C\alpha_i = \cos\alpha_i$, $S_{ijk} = \sin(\theta_i + \theta_j + \theta_k)$, $C_{ijk} = \sin(\theta_i + \theta_j + \theta_k)$

The overall transformation matrix, ${}^{0}T_{5} = {}^{0}T_{1} * {}^{1}T_{2} * {}^{2}T_{3} * {}^{3}T_{4} * {}^{4}T_{5}$

$$ {}^{0}T_{1} = \begin{bmatrix} c_1 & 0 & s_1 & a_1c_1 \\ s_1 & 0 & -c_1 & a_1s_1 \\ 0 & 1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad {}^{1}T_{2} = \begin{bmatrix} c_2 & -s_2 & 0 & a_2c_2 \\ s_2 & c_2 & 0 & a_2s_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad {}^{2}T_{3} = \begin{bmatrix} c_3 & -s_3 & 0 & a_3c_3 \\ s_3 & c_3 & 0 & a_3s_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^{3}T_{4} = \begin{bmatrix} c_4 & 0 & s_4 & 0 \\ s_4 & 0 & -c_4 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad {}^{4}T_{5} = \begin{bmatrix} c_5 & -s_5 & 0 & 0 \\ s_5 & c_5 & 0 & 0 \\ 0 & 0 & 1 & d_5 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

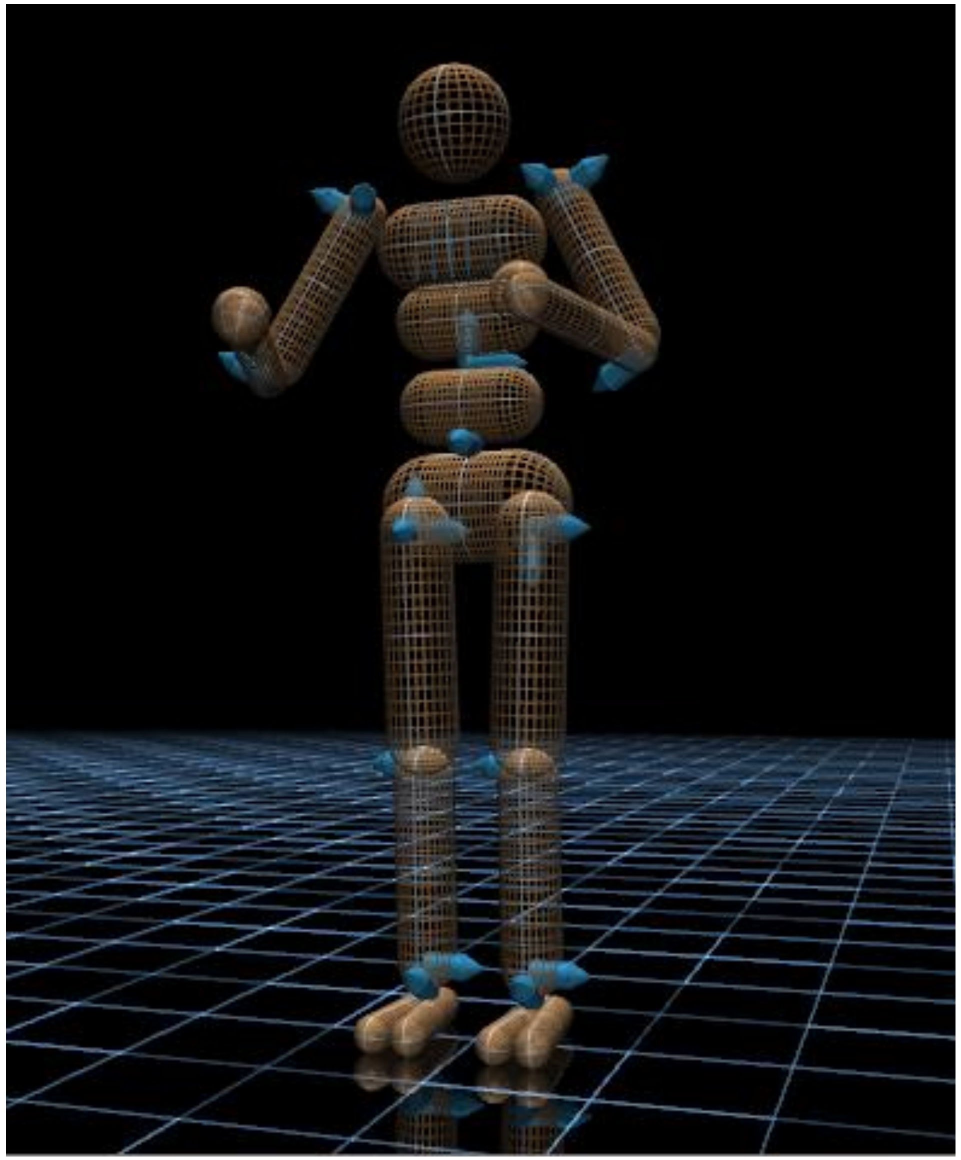
$$ {}^{0}T_{5} = \begin{bmatrix} c_1c_{234}c_5 + s_1s_5 & c_1c_{234}s_5 + s_1c_5 & c_1s_{234} & c_1(d_5s_{234} + a_3c_{23} + a_2c_2 + a_1) \\ s_1c_{234}c_5 - c_1s_5 & -s_1c_{234}s_5 - c_1c_5 & s_1s_{234} & s_1(d_5c_{234} + a_3s_{23} + a_2c_2 + a_1) \\ s_{234}c_5 & -s_{234}s_5 & -c_{234} & (-d_5c_{234} + a_3s_{23} + a_2s_2 + d_1) \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

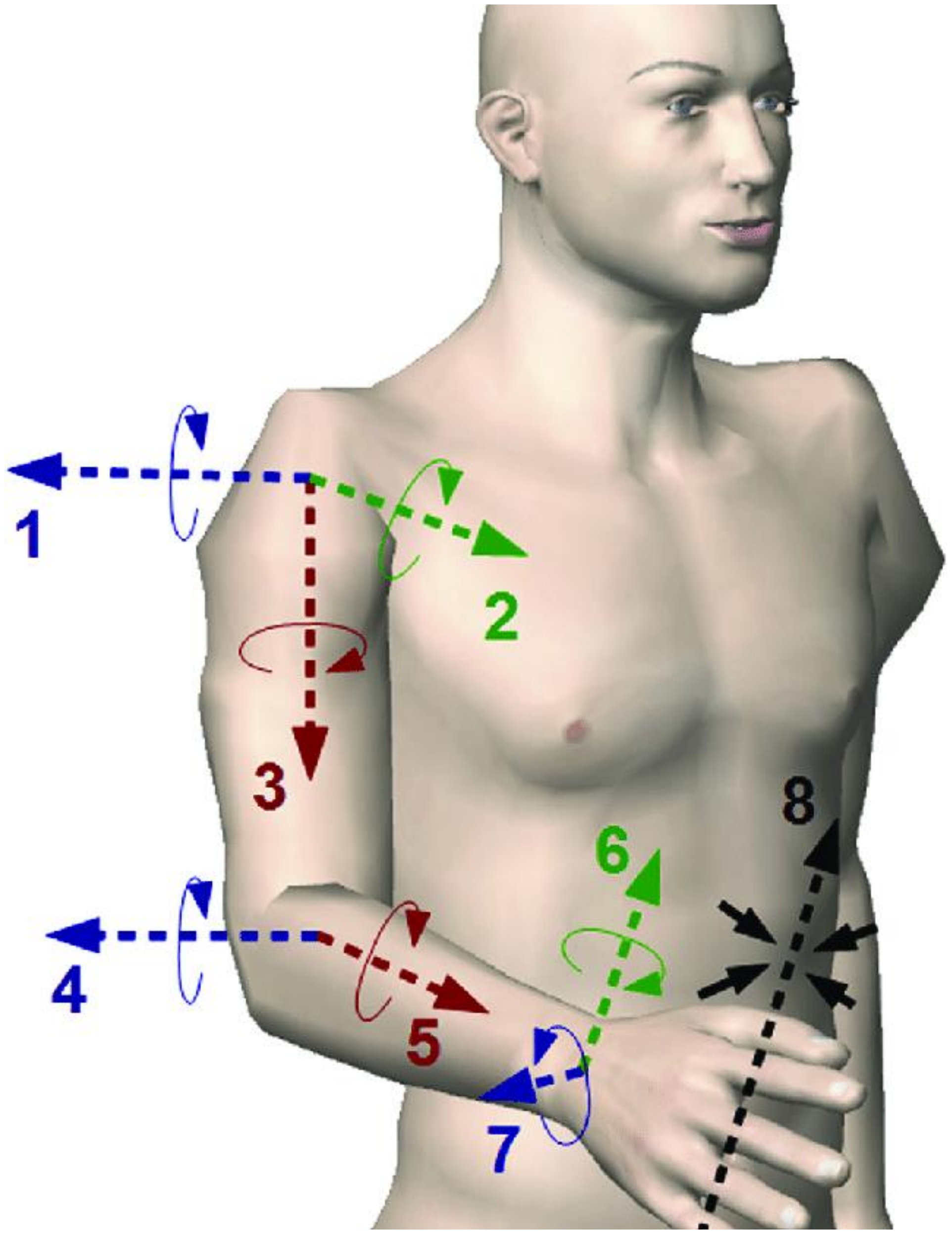
$$ {}^{0}T_{5} = T_e = \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

Where, Te is end-effector transformation matrix.

This kinematic model can also be expressed by 12 equations as:

nx = c1*c234*c5+s1*s5
ny = s1*c234*c5-c1*s5
nz = s234*c5
ox = c1*c234*s5+s1*c5
oy = -s1*c234*s5-c1*c5
oz = -s234*s5
ax = c1*s234
ay = s1*s234
az = -c234
px = c1*(s234*d5+a3*c23+a2*c2+a1)
py = s1*(s234*d5+a3*c23+a2*c2+a1)
pz = -c234*d5+a3*s23+a2*s2+d1




1
2
3
4
5
6
7
8

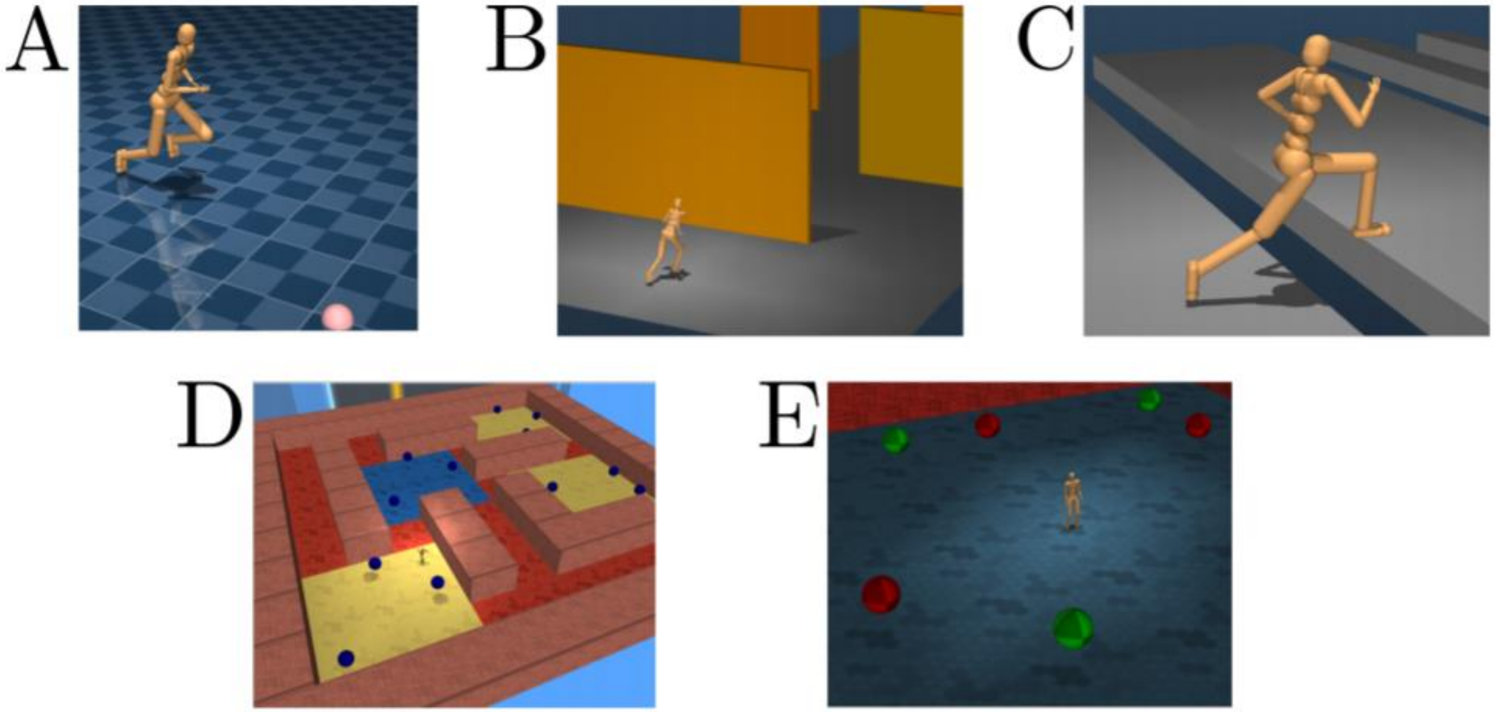


Figure 5: A. Go-to-target: in this task, the agent moves on an open plane to a target provided in egocentric coordinates. B. Walls: The agent runs forward while avoiding solid walls using vision. C. Gaps: The agent runs forward and must jump between platforms to advance. D. Forage: Using vision, the agent roams in a procedurally-generated maze to collect balls, which provide sparse rewards. E. Heterogeneous Forage: The agent must probe and remember rewards that are randomly assigned to the balls in each episode.

Approach

1. Tracking motion capture clips (Low level policy)
2. Training High level policy

1. Tracking motion capture clips

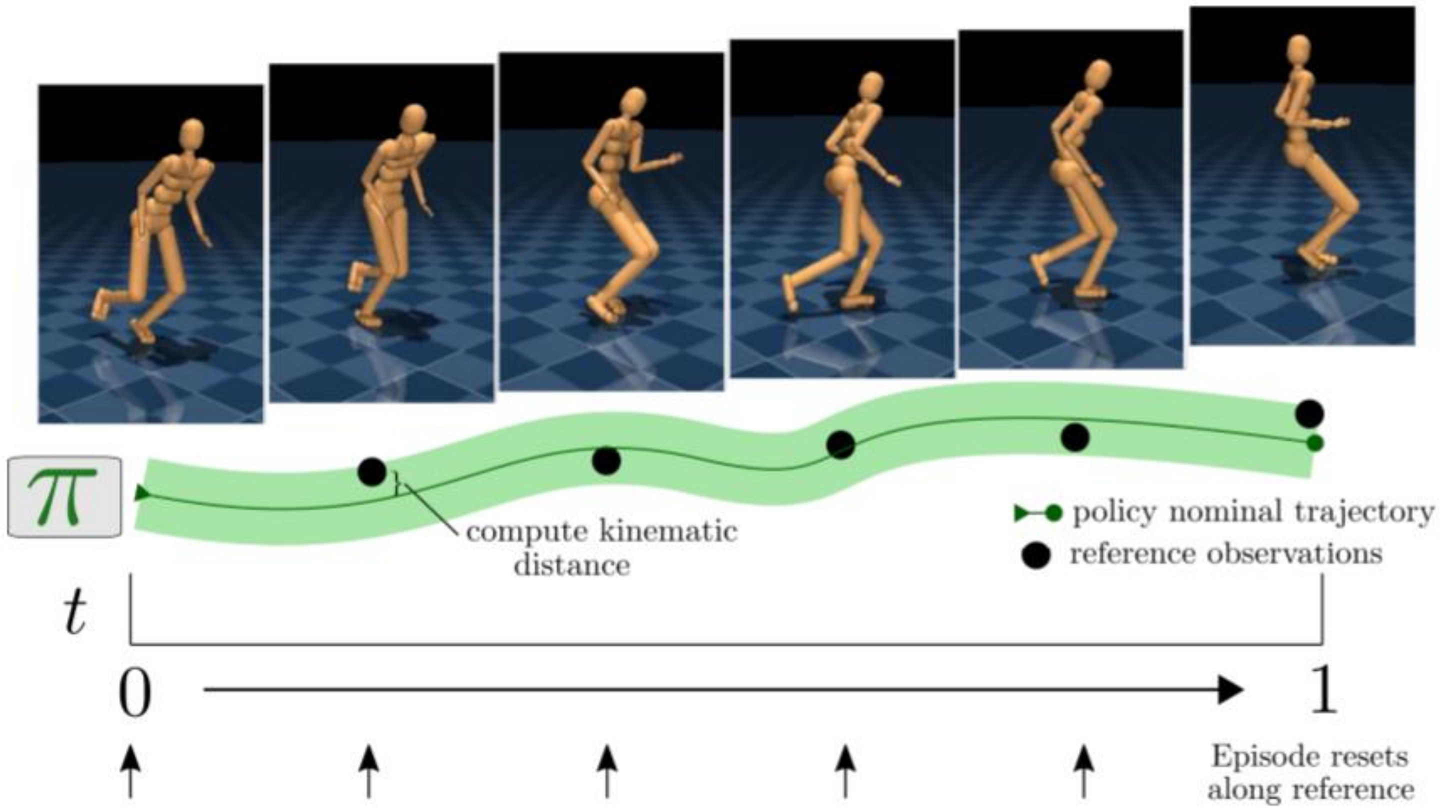
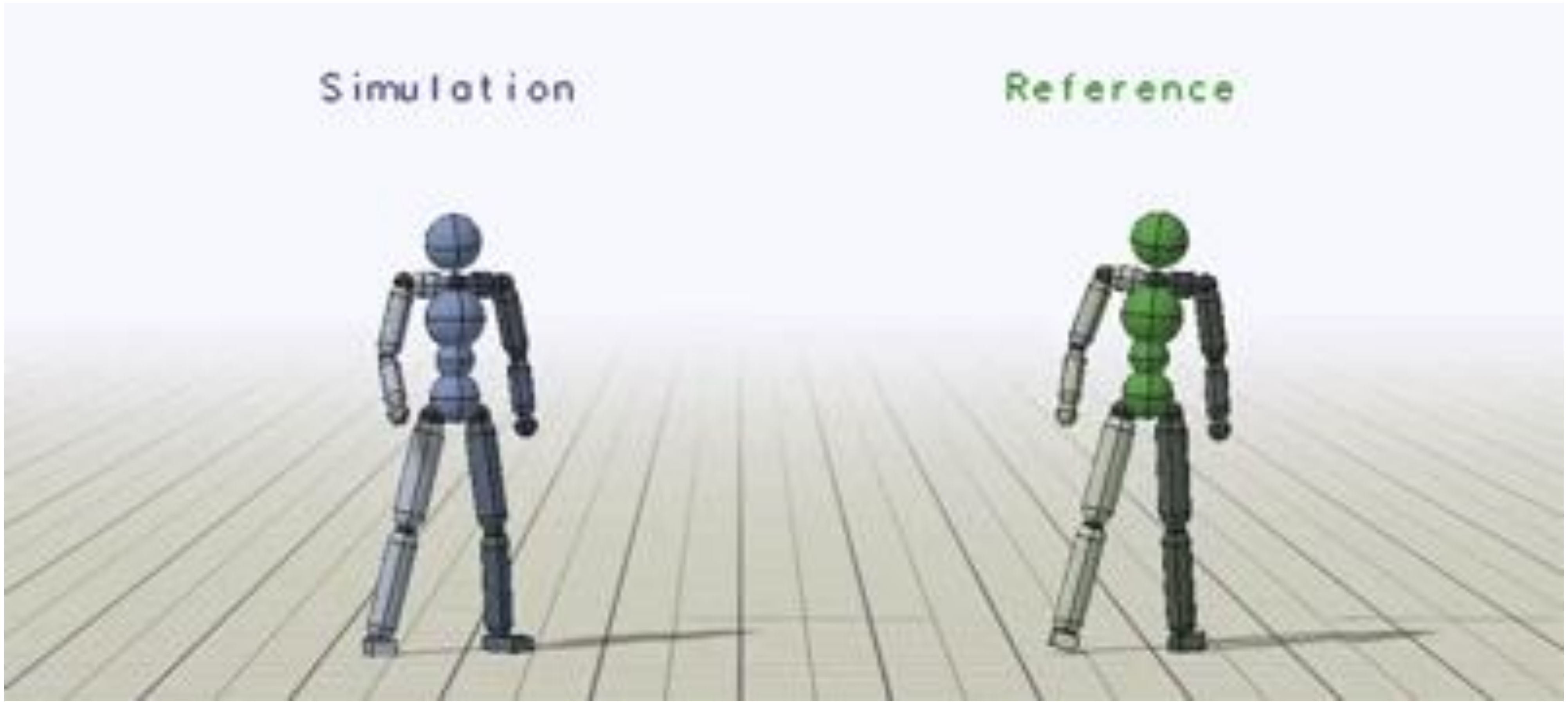
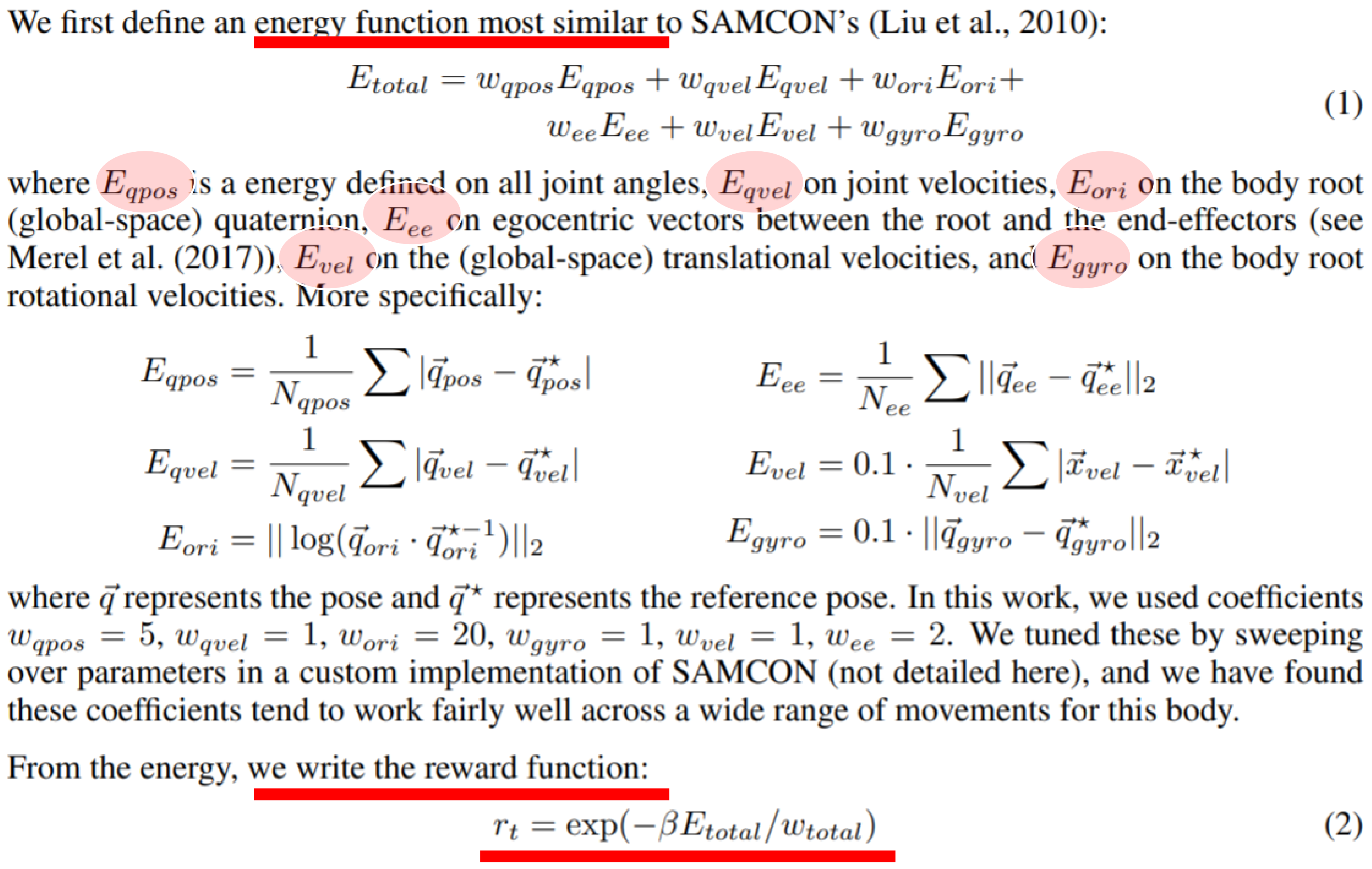

π
compute kinematic distance
policy nominal trajectory
reference observations
t
0
1
Episode resets along reference

Figure 1: Illustration of tracking-based RL training. Training iteratively refines a policy to robustly track the reference trajectory as well as physically feasible.

1. Tracking motion capture clips



1. Tracking motion capture clips – Reward function



We first define an energy function most similar to SAMCON's (Liu et al., 2010):

$$E_{total} = w_{qpos} E_{qpos} + w_{qvel} E_{qvel} + w_{ori} E_{ori} + w_{ee} E_{ee} + w_{vel} E_{vel} + w_{gyro} E_{gyro} \tag{1}$$

where E_{qpos} is a energy defined on all joint angles, E_{qvel} on joint velocities, E_{ori} on the body root (global-space) quaternion, E_{ee} on egocentric vectors between the root and the end-effectors (see Merel et al. (2017)), E_{vel} on the (global-space) translational velocities, and E_{gyro} on the body root rotational velocities. More specifically:

$$E_{qpos} = \frac{1}{N_{qpos}} \sum |\vec{q}_{pos} - \vec{q}^{\star}_{pos}| \qquad E_{ee} = \frac{1}{N_{ee}} \sum ||\vec{q}_{ee} - \vec{q}^{\star}_{ee}||_2$$

$$E_{qvel} = \frac{1}{N_{qvel}} \sum |\vec{q}_{vel} - \vec{q}^{\star}_{vel}| \qquad E_{vel} = 0.1 \cdot \frac{1}{N_{vel}} \sum |\vec{x}_{vel} - \vec{x}^{\star}_{vel}|$$

$$E_{ori} = || \log(\vec{q}_{ori} \cdot \vec{q}^{\star -1}_{ori}) ||_2 \qquad E_{gyro} = 0.1 \cdot ||\vec{q}_{gyro} - \vec{q}^{\star}_{gyro}||_2$$

where $\vec{q}$ represents the pose and $\vec{q}^{\star}$ represents the reference pose. In this work, we used coefficients $w_{qpos} = 5$, $w_{qvel} = 1$, $w_{ori} = 20$, $w_{gyro} = 1$, $w_{vel} = 1$, $w_{ee} = 2$. We tuned these by sweeping over parameters in a custom implementation of SAMCON (not detailed here), and we have found these coefficients tend to work fairly well across a wide range of movements for this body.

From the energy, we write the reward function:

$$r_t = \exp(-\beta E_{total} / w_{total}) \tag{2}$$

1. Tracking motion capture clips – State and Action

State.

joint angles (q_pos) and velocities (q_vel), root-to-end-effector vectors (q_ee), root frame velocimeter (q_veloc), rotational velocity (q_gyro), root-frame accelerometers (q_accel), and 3D orientation relative to the z-axis (rz: functionally a gravity sensor).

Action

(56 DOFs)

Each joint is given an actuation range in [−1, 1], and this is mapped to the angular range of that joint.

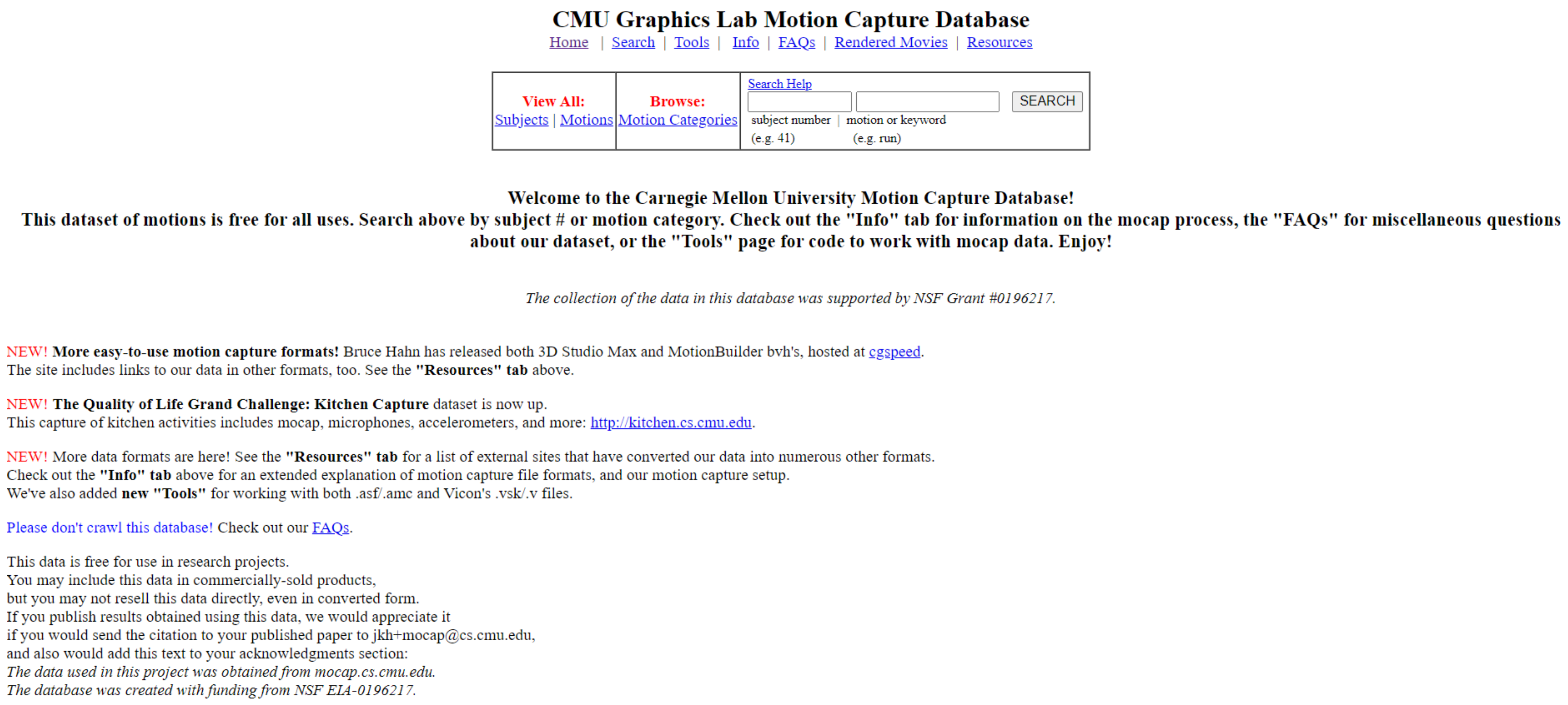
1. Tracking motion capture clips – Train

1. Pre-train policy to produce target pose by supervised learning.
2. The Q-function was learned off-policy using TD-learning using importance-weighted Retrace (Munos et al., 2016), and the actor was learned off-policy using SVG(0) (Heess et al., 2015).

$$\max_{\pi_\theta} \sum_{\tau=1\ldots T} \mathbb{E}_{a\sim\pi(a|s_\tau)}[Q_{target}(s_\tau, a)] - \eta D_{KL}[\pi_\theta(a|s_\tau)||\pi_{target}(a|s_\tau)]$$

1. Tracking motion capture clips – Train

Reference data



CMU Graphics Lab Motion Capture Database

Home | Search | Tools | Info | FAQs | Rendered Movies | Resources

View All: Subjects | Motions
Browse: Motion Categories
Search Help
subject number (e.g. 41)
motion or keyword (e.g. run)
SEARCH

Welcome to the Carnegie Mellon University Motion Capture Database!
This dataset of motions is free for all uses. Search above by subject # or motion category. Check out the "Info" tab for information on the mocap process, the "FAQs" for miscellaneous questions about our dataset, or the "Tools" page for code to work with mocap data. Enjoy!

The collection of the data in this database was supported by NSF Grant #0196217.

NEW! **More easy-to-use motion capture formats!** Bruce Hahn has released both 3D Studio Max and MotionBuilder bvh's, hosted at cgspeed.
The site includes links to our data in other formats, too. See the **"Resources" tab** above.

NEW! **The Quality of Life Grand Challenge: Kitchen Capture** dataset is now up.
This capture of kitchen activities includes mocap, microphones, accelerometers, and more: http://kitchen.cs.cmu.edu.

NEW! More data formats are here! See the **"Resources" tab** for a list of external sites that have converted our data into numerous other formats.
Check out the **"Info" tab** above for an extended explanation of motion capture file formats, and our motion capture setup.
We've also added **new "Tools"** for working with both .asf/.amc and Vicon's .vsk/.v files.

Please don't crawl this database! Check out our FAQs.

This data is free for use in research projects.
You may include this data in commercially-sold products,
but you may not resell this data directly, even in converted form.
If you publish results obtained using this data, we would appreciate it
if you would send the citation to your published paper to jkh+mocap@cs.cmu.edu,
and also would add this text to your acknowledgments section:
The data used in this project was obtained from mocap.cs.cmu.edu.
The database was created with funding from NSF EIA-0196217.

2. Training High level policy

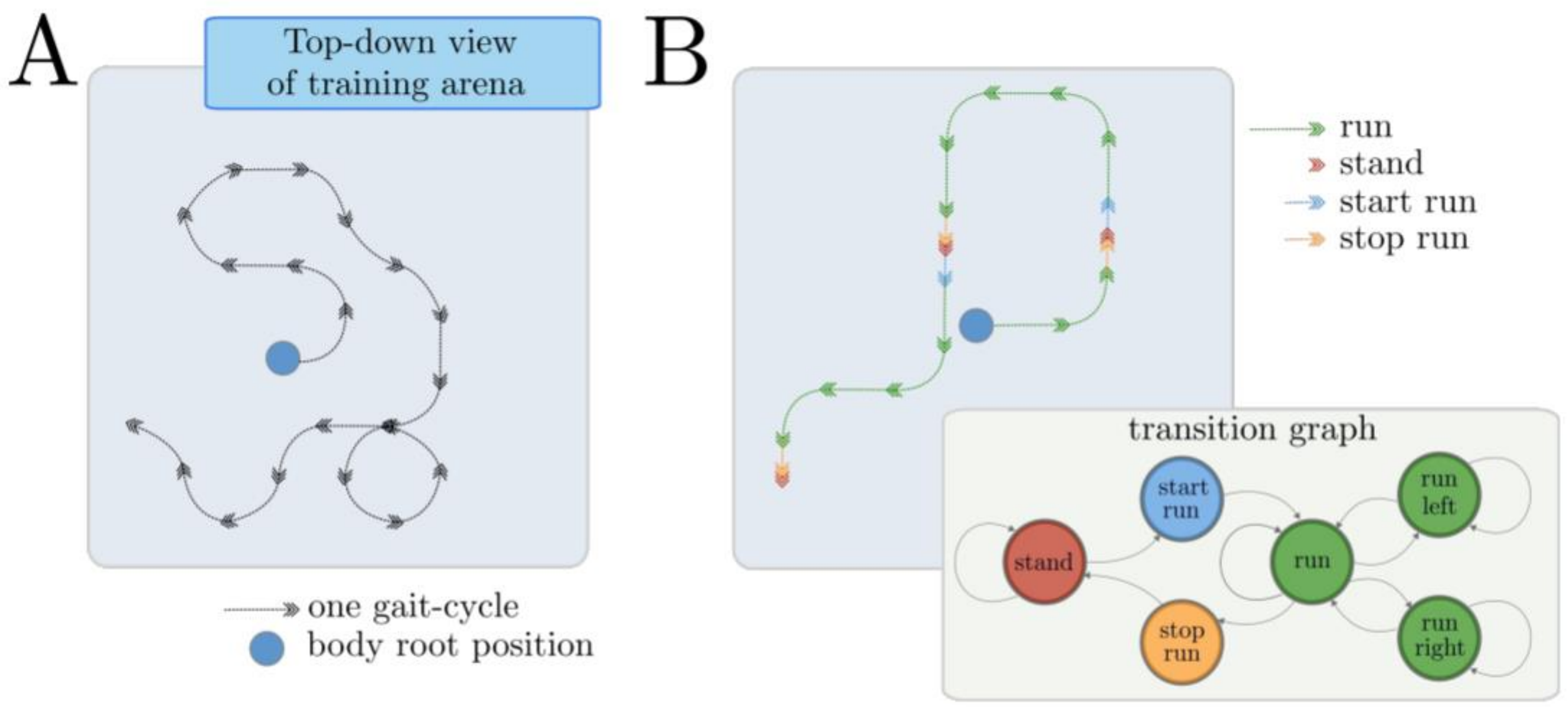


Figure 2: Training settings for explicit training of transition-capable controllers. Panel A depicts a cartoon of a training episode for a steerable controller in which the turning radius of a each gait-cycle is selected randomly. Panel B depicts training a policy under an explicit, hand-designed transition graph for k options.

2. Training High level policy

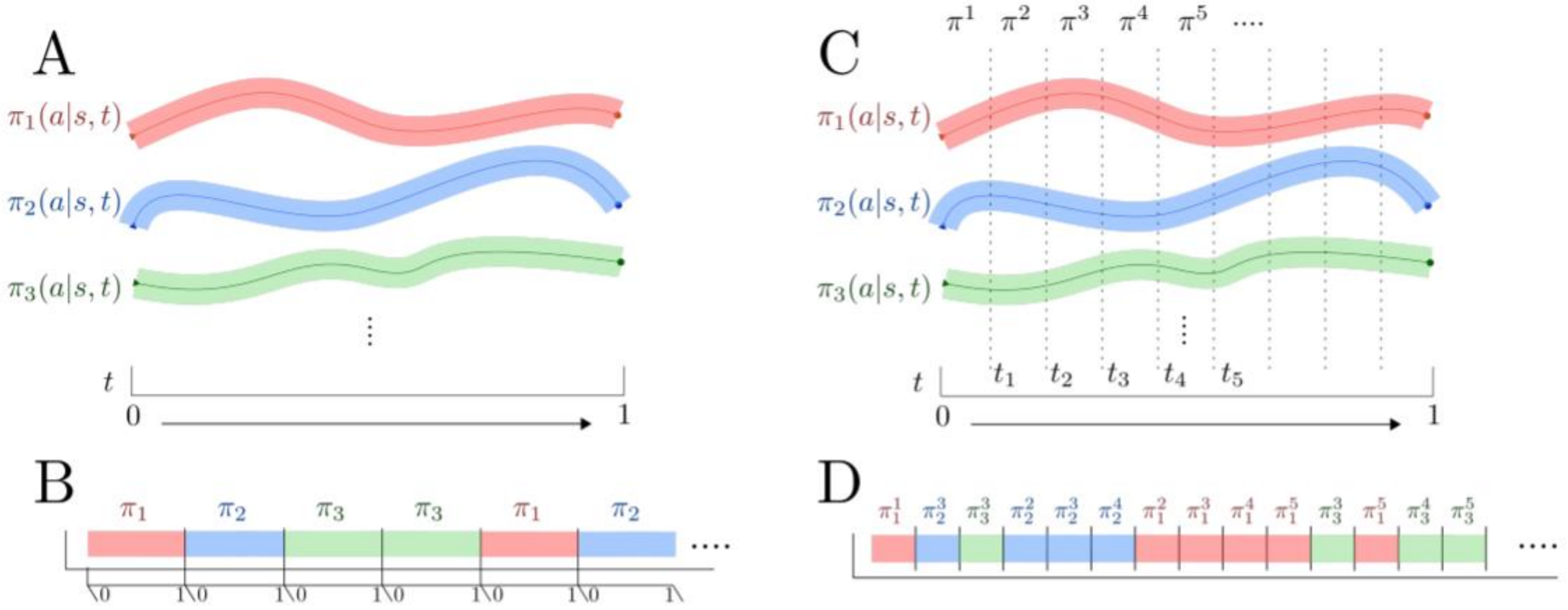

A
$\pi_1(a|s,t)$
$\pi_2(a|s,t)$
$\pi_3(a|s,t)$
t
0
1
B
π_1 π_2 π_3 π_3 π_1 π_2
C
π^1 π^2 π^3 π^4 π^5 ....
t_1 t_2 t_3 t_4 t_5
D
π_1^1 π_2^3 π_3^3 π_2^2 π_2^3 π_2^4 π_1^2 π_1^3 π_1^4 π_1^5 π_3^3 π_1^5 π_3^4 π_3^5

Figure 3: Cold-switching among a set of behaviors (A) only at end of clips to form a trajectory composed of sequentially activation of the policies (B). Alternatively, policies are fragmented at a pre-specified set of times, cutting the policy into sub-policies (C), which serve as *control fragments*, enabling sequencing at a higher frequency (D).

2. Training High level policy

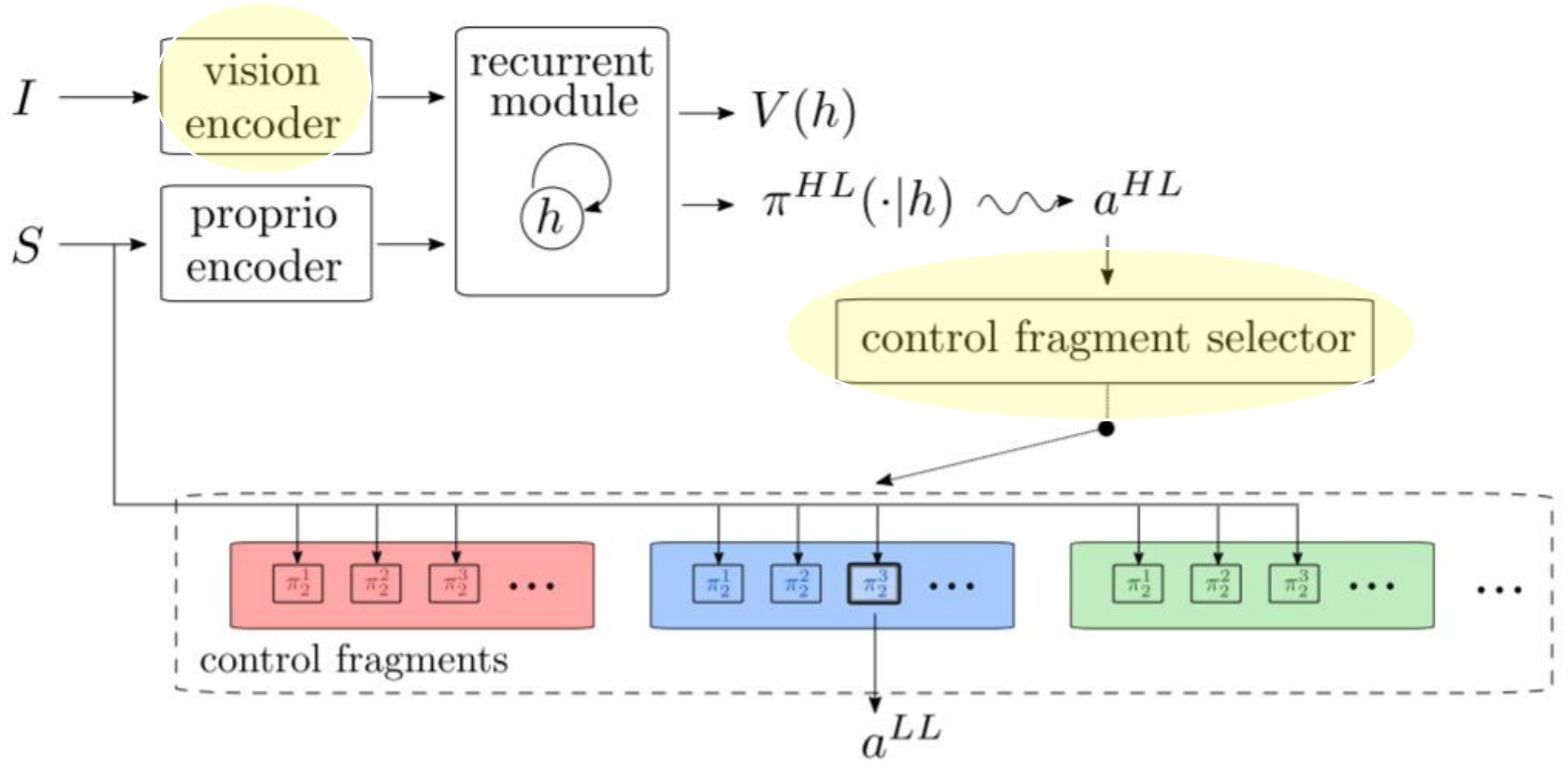


Figure 4: Schematic of the architecture: a high-level controller (HL) selects among multiple low-level (LL) control fragments, which are policies with proprioception. Switching from one control fragment to another occurs every k time steps.

- Gaps test: 12 single-clip policies, 359 fragments (0.09s)
- To train these discrete controllers, we fit the state-value baseline V -function using V-Trace and update the policy according to the method in Espeholt et al. (2018). - IMPALA

2. Training High level policy

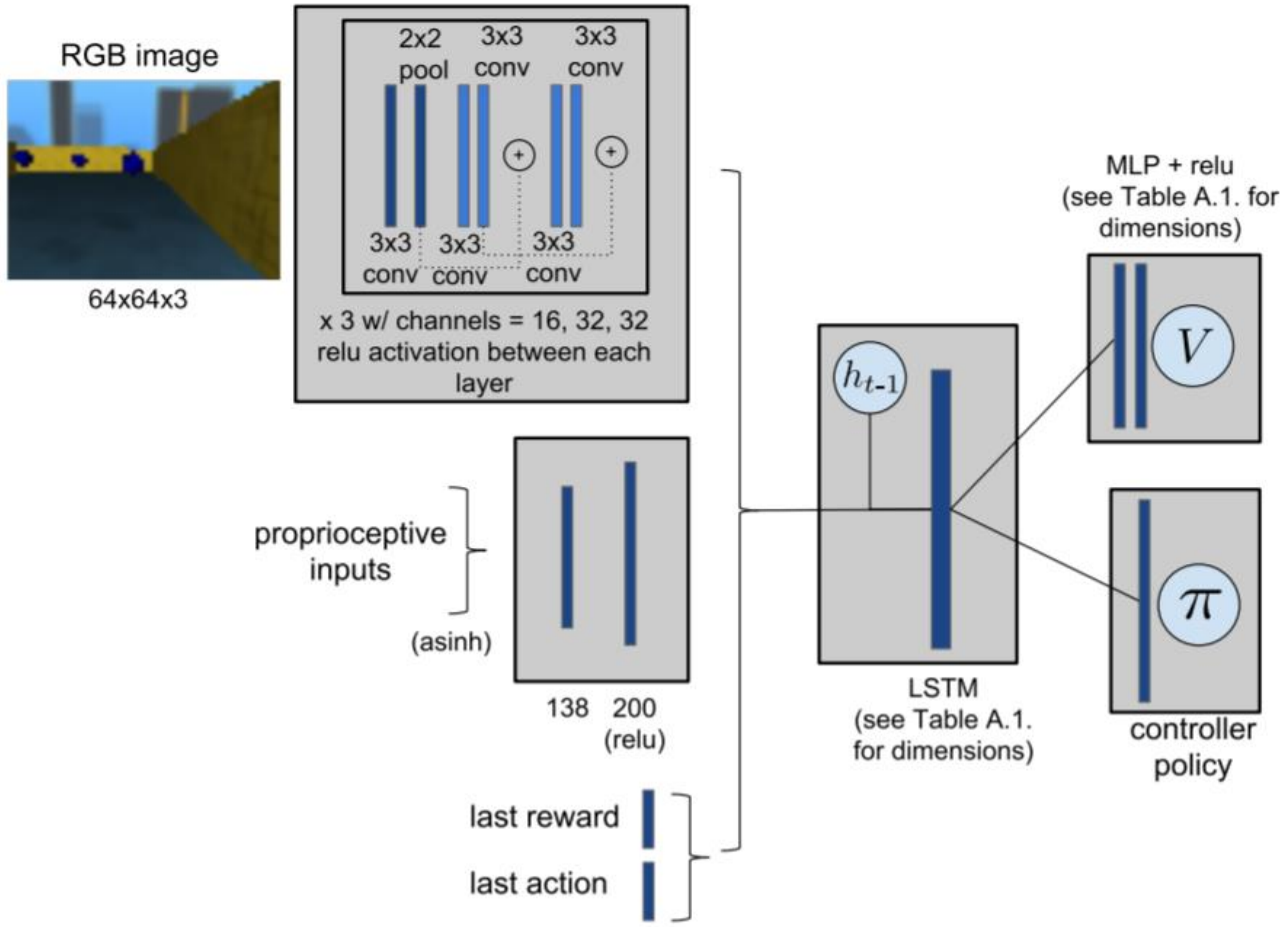


Figure A.1: Complete diagram of high-level agent architecture with encoders.

3. Result

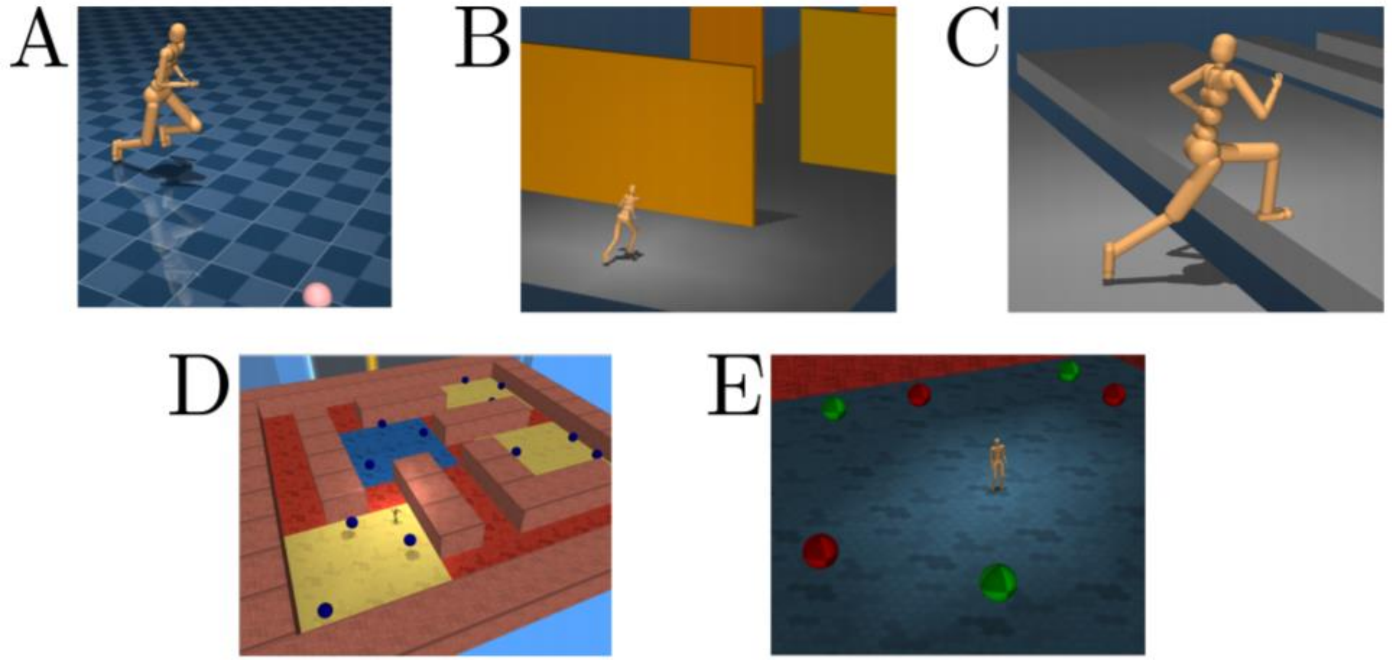
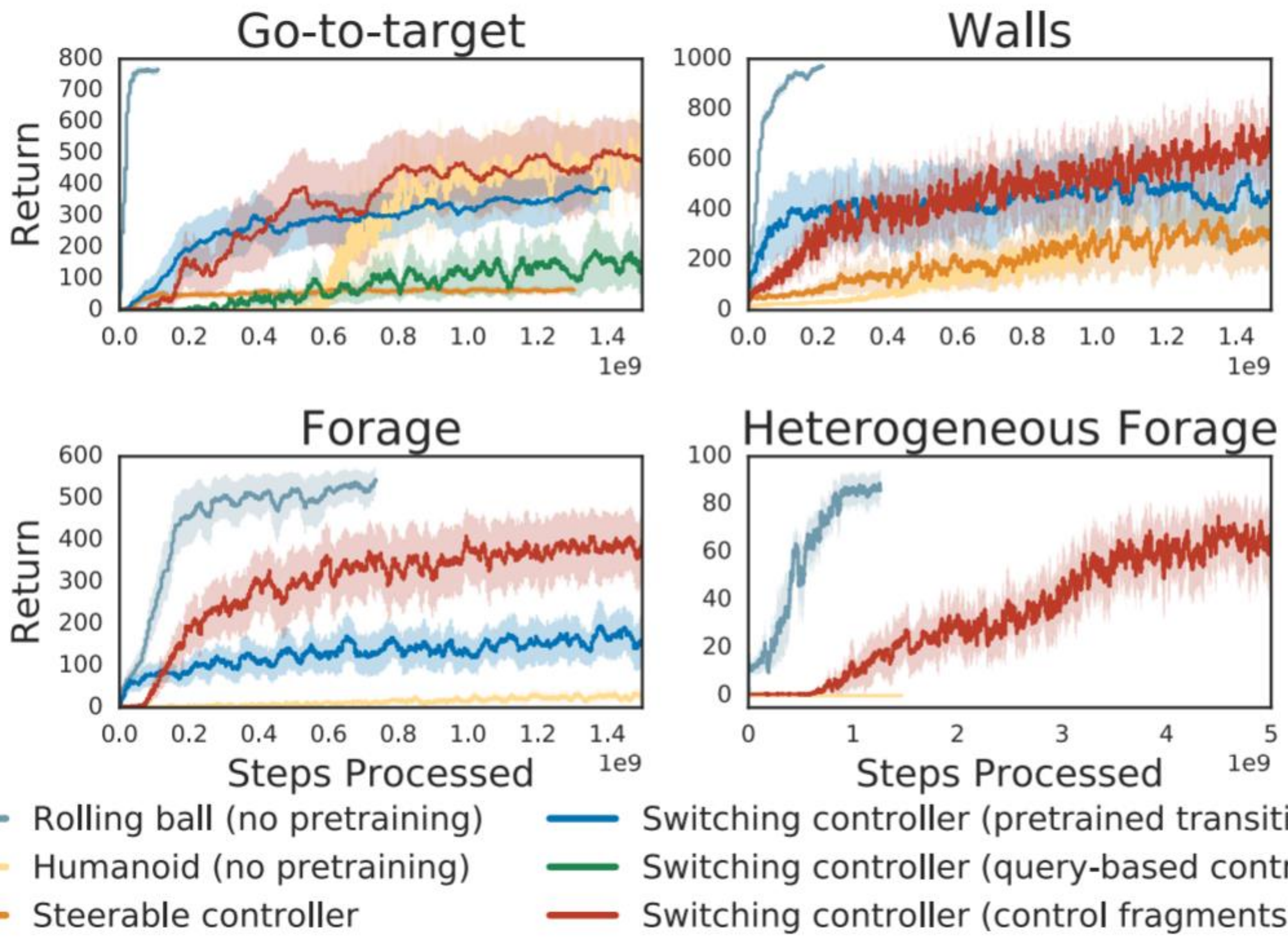


Figure 5: A. Go-to-target: in this task, the agent moves on an open plane to a target provided in egocentric coordinates. B. Walls: The agent runs forward while avoiding solid walls using vision. C. Gaps: The agent runs forward and must jump between platforms to advance. D. Forage: Using vision, the agent roams in a procedurally-generated maze to collect balls, which provide sparse rewards. E. Heterogeneous Forage: The agent must probe and remember rewards that are randomly assigned to the balls in each episode.

3. Result


Go-to-target
Walls
Forage
Heterogeneous Forage
Return
Steps Processed
1e9
Rolling ball (no pretraining)
Humanoid (no pretraining)
Steerable controller
Switching controller (pretrained transitions)
Switching controller (query-based control fragments)
Switching controller (control fragments)

3. Result

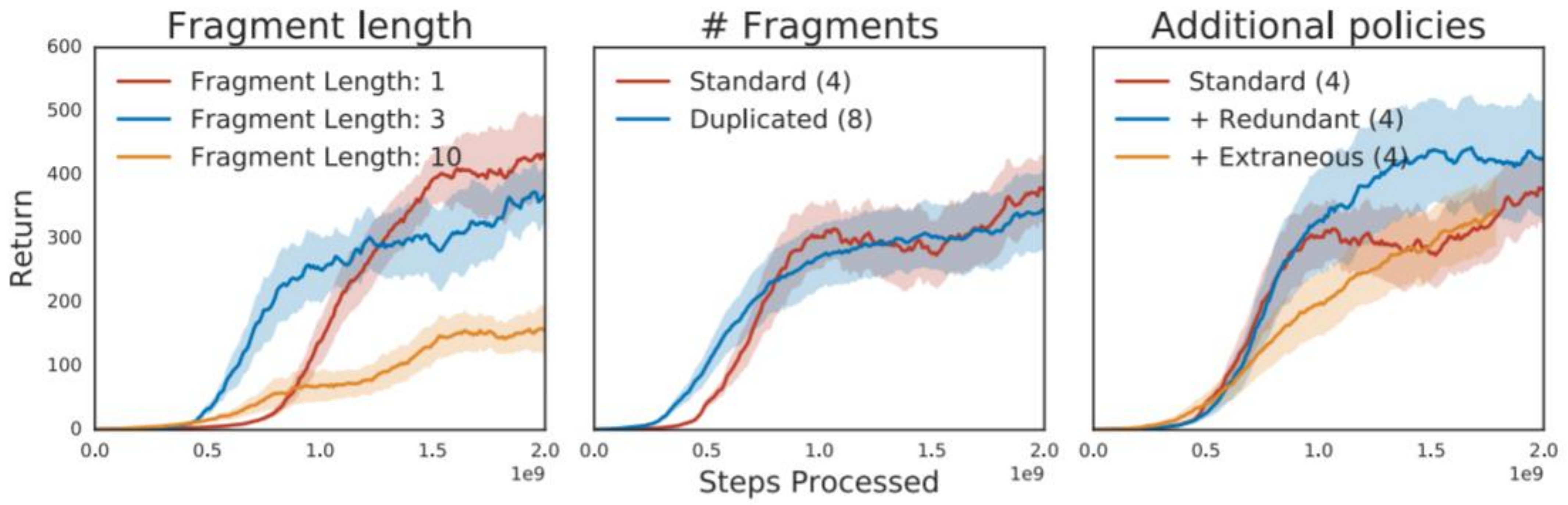
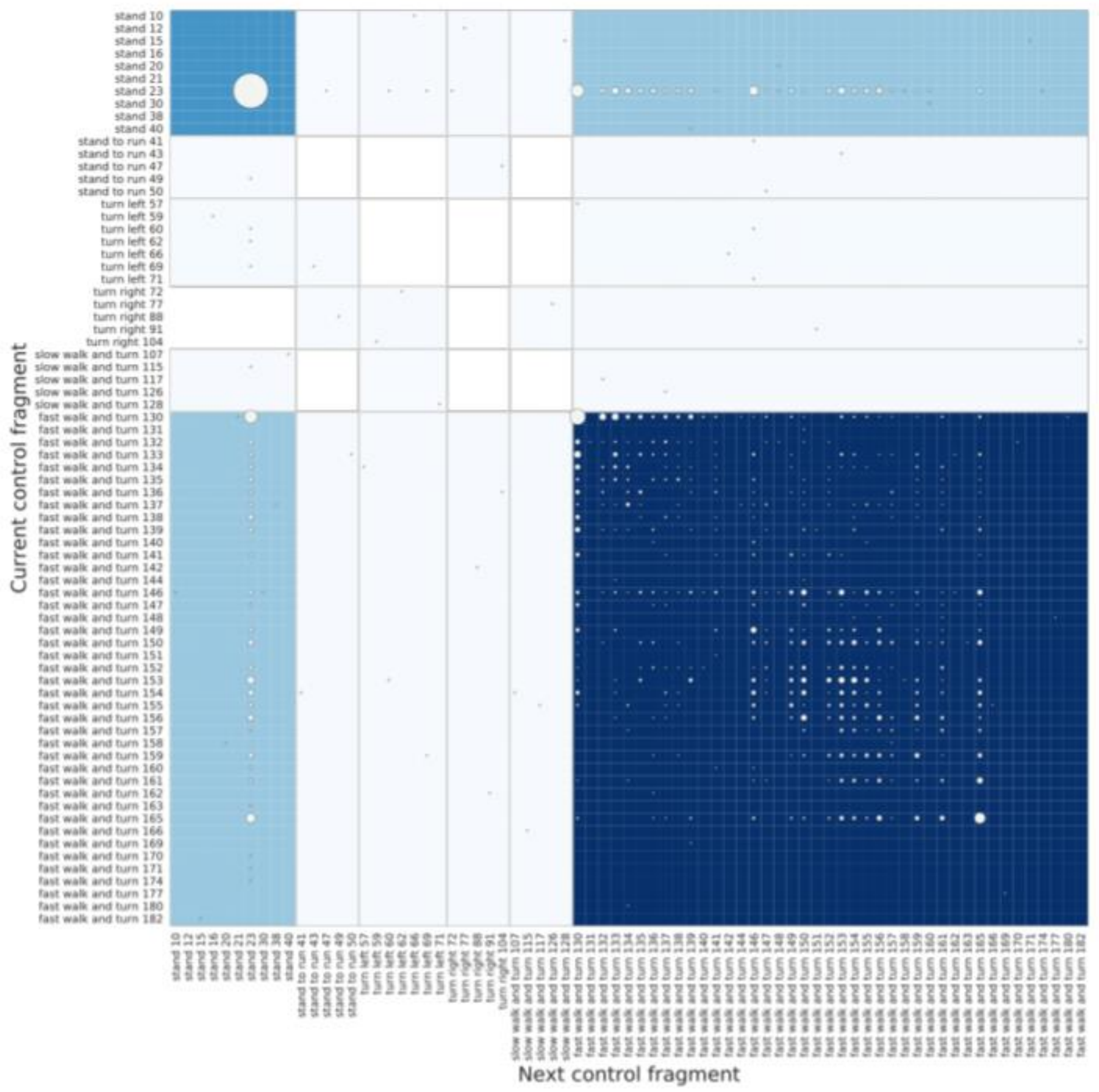


Figure A.4: A. Training was most stable with shorter fragment length (1 / .03 sec or 3 / .09 sec. B. Increasing the number of fragments, by duplicating the original set, did not hurt training performance. C. Additional functionally redundant policies (i.e. 4 vs 8 behaviors, cut into many control fragments) improved training speed, while additional extraneous policies were easily ignored.

3. Result

C Transition analyses of an example policy



4. Summary: Main contributions

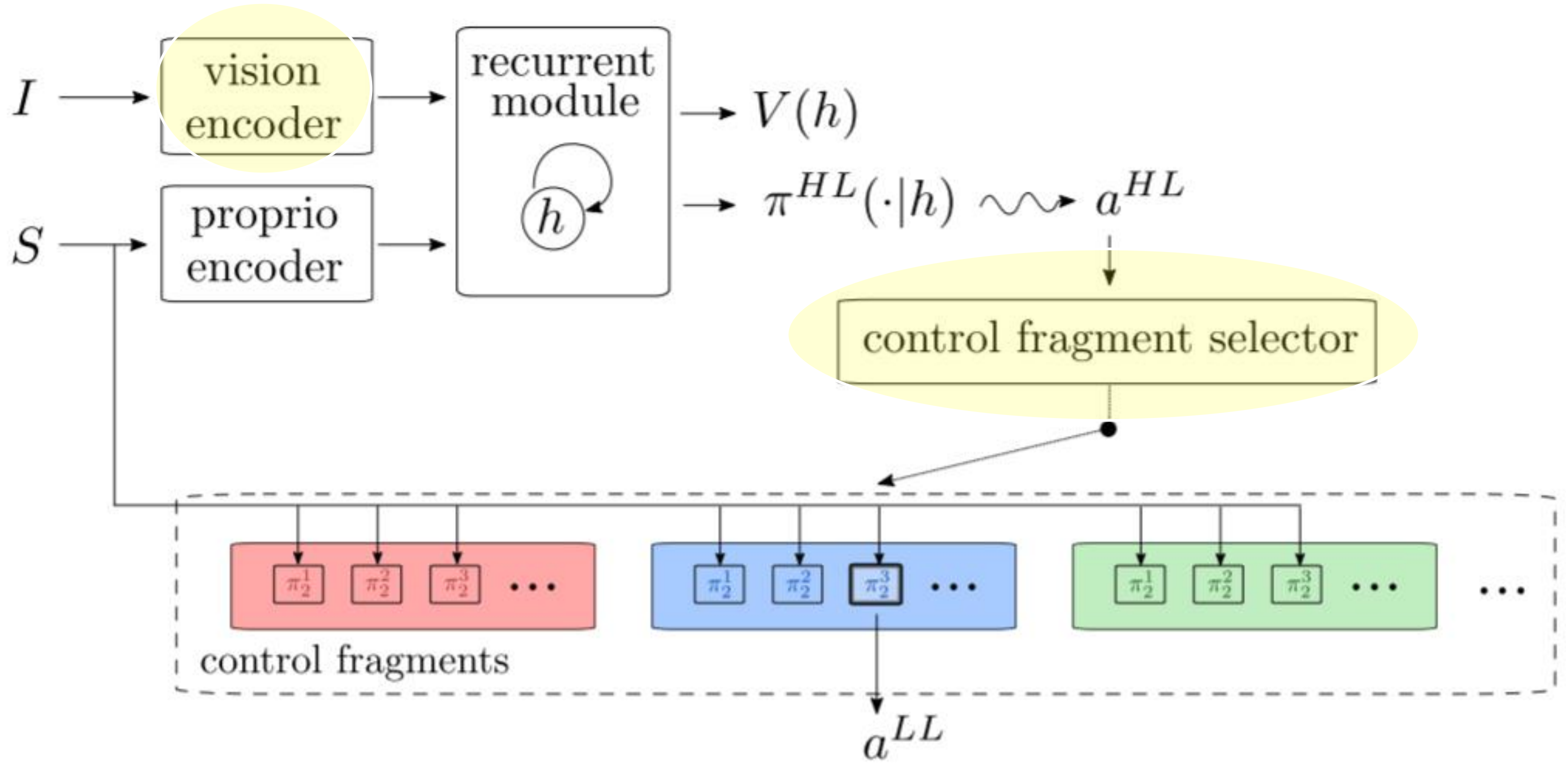


Figure 4: Schematic of the architecture: a high-level controller (HL) selects among multiple low-level (LL) control fragments, which are policies with proprioception. Switching from one control fragment to another occurs every k time steps.

* Gaps test: 12 single-clip policies, 359 fragments (0.09s)

Q & A