Towards Non-Parametric Bayesian Learning of Robot Behaviors from Demonstration

Stéphane Magnenat, Cédric Pradalier, and Francis Colas

ETH Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich
Autonomous Systems Lab

Problem Statement

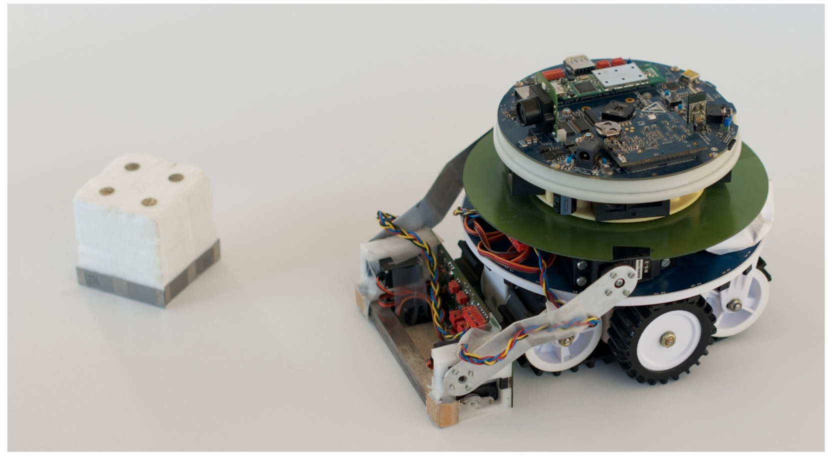
Real-world robotic applications typically require basic actions:

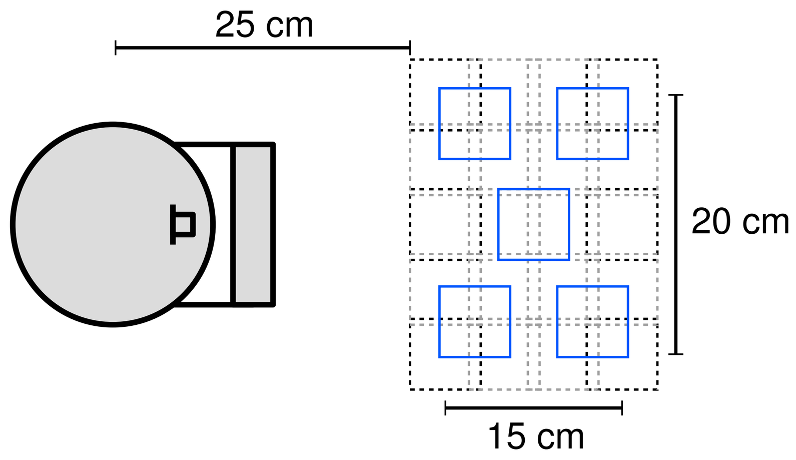
- Grasping an object, docking to a recharge station, etc.
- Typically uses FSM: slow to program, to debug, to tune.

We want to specify these actions by demonstrating them:

- *Trajectory following* replay (ex. GMR) are typically reactive.
- *Sequence segmentation* demands a definition of step changes.
- Both approches require many parameters/design choices.

Experiments

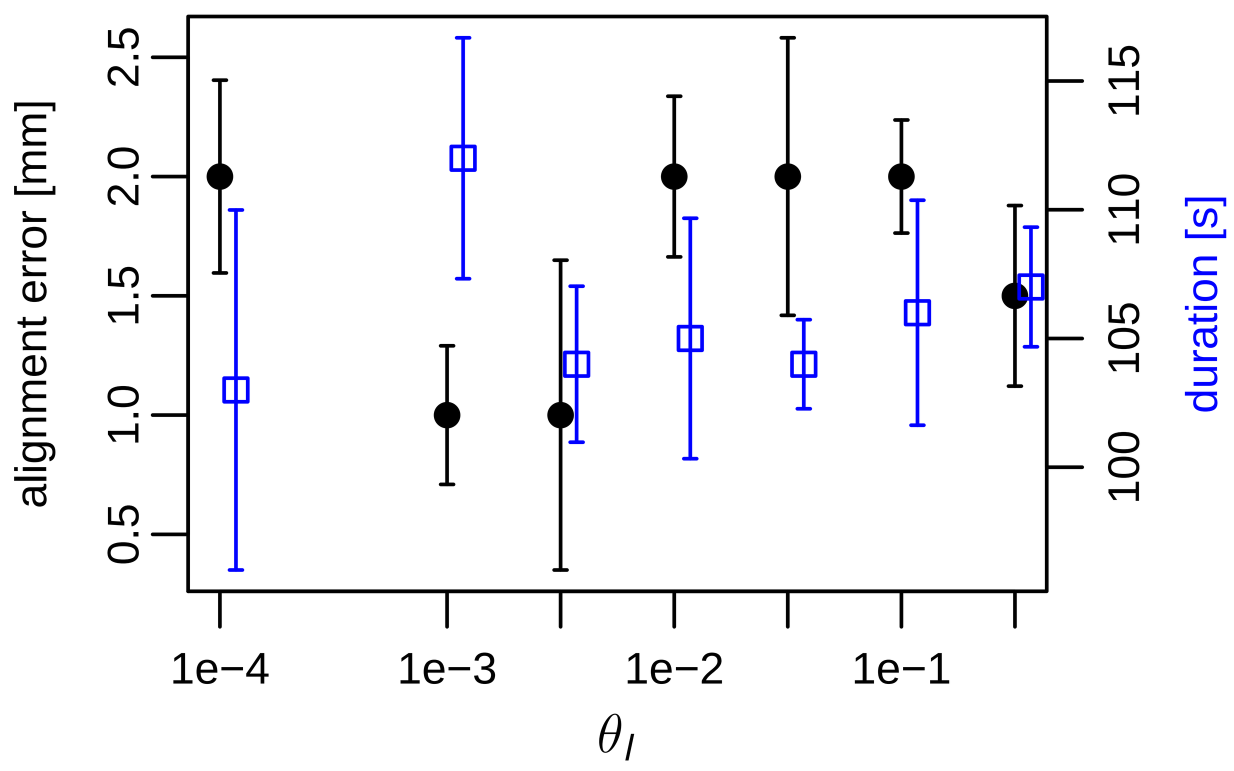




- 20 recorded trajectories (black dots), 3 × 5 different tested positions (blue)
- θ_τ=0.05, 7 different tested values for θ_I

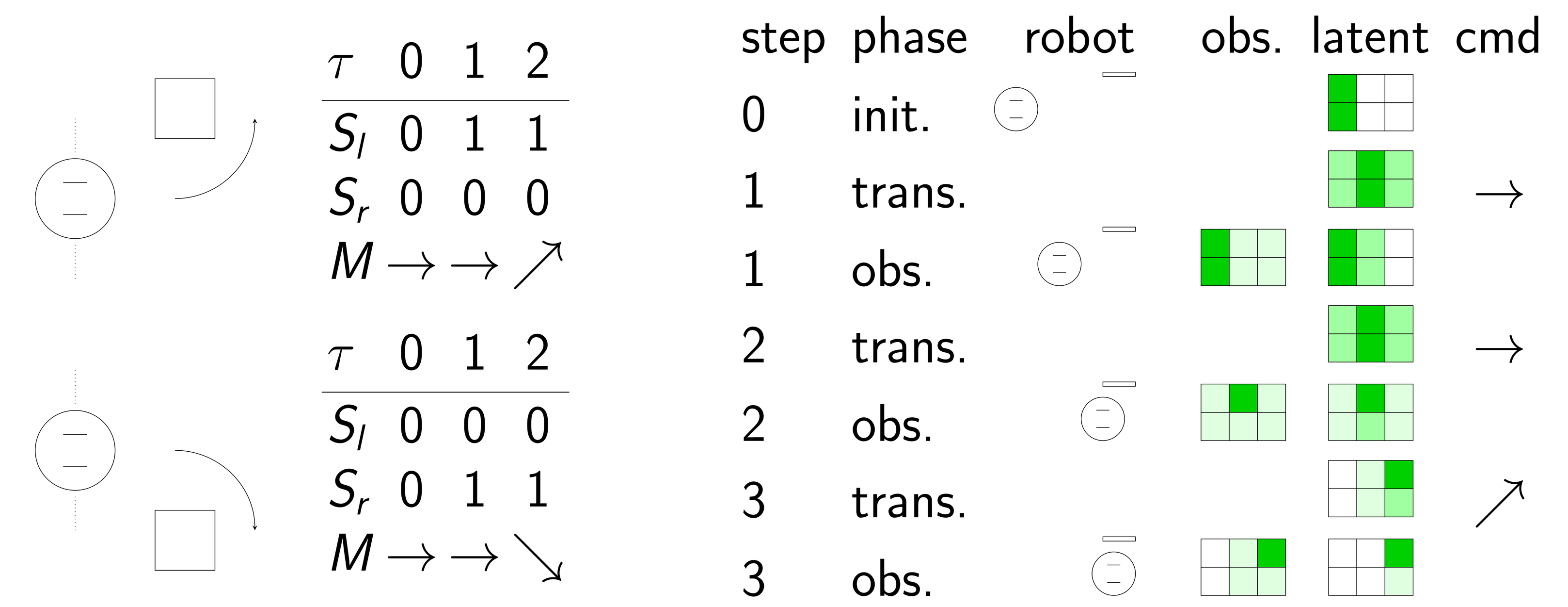
Results

with 20 demonstrations:				
θ_I	S	Fs	Fg	Fb
1e-0.5	14	1	0	0
1e-1	13	1	0	1
1e-1.5	13	0	1	1
1e-2	14	0	1	1
1e-2.5	7	6	1	1
1e-3	11	2	2	0
1e-4	9	5	0	1
with 6 demonstrations:				
1e-0.5	11	3	1	0



- For θ_I comprised between 1e-0.5 and 1e-2, the success rate is high at 90 %.
- A large θ_I is better because training runs differ mostly at the beginning.
- In case of success, the alignment error is small and the duration constant.
- Most of the failures (60 %) are linked to the controller stopping the robot indefinitely, due to a fixed or cyclic distribution on I_t, τ_t.
- With only 6 recorded trajectories, performances degrade gracefully: there are more failures but on successful runs mean error and duration are similar.

Example of Model Execution



Current and Future Work

- Test on search-and-rescue robot, PR2
- Observation model (Cauchy) and transition model (Log-normal)
- Sensor weighting (feature selection)
- Abstraction (similar subsequences, loops, branching)

Conclusion

- Real-time on laptop in Python/Cython, complexity: $O(L \times N)$
- Successful application to cube grasping
- Strong potential for other types of robots

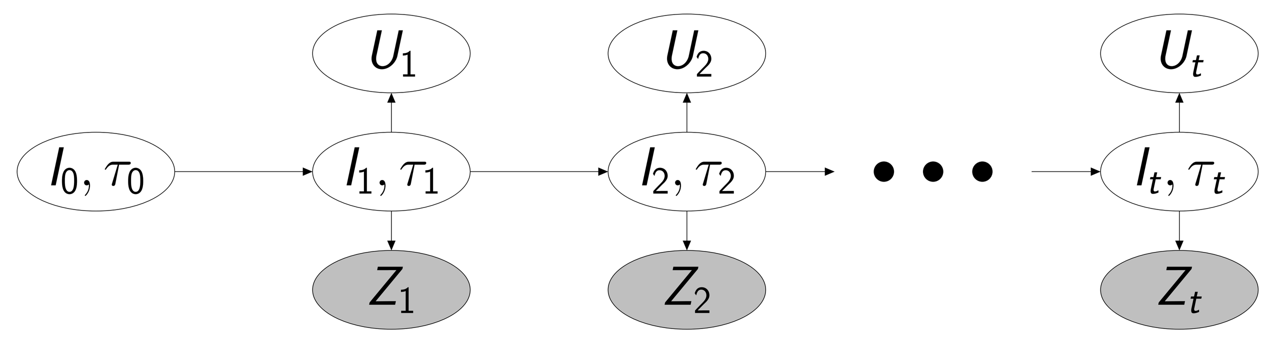
Proposed Model

- Execution based on tracking in the training data.
- Observations and commands are conditioned by trajectory and time indices.
- Updates this latent space with observations and transition models.
- Generates commands by averaging recorded motor data.

Variables

- $\Pi = \{\zeta_t^i, \upsilon_t^i | \forall i \in (1, N), \forall t \in (1, L_i)\}$ Records of N trajectories; trajectory i at record time step t has sensor data ζ_t^i and actuator command υ_t^i.
- I_t Index of trajectory at replay time t, ranges from 1 to N.
- τ_t Position on trajectory at replay time t, ranges from 1 to $\max_i L_i$.
- U_t Actuator command at replay time t, vector value.
- Z_t Observation (sensor data) at replay time t, n_s-dim vector value.

Distributions



$$p(U_{1:t}, Z_{1:t}, I_{1:t}, \tau_{1:t}) = p(U_{1:t-1}, Z_{1:t-1}, I_{1:t-1}, \tau_{1:t-1}) p(U_t|I_t, \tau_t) p(Z_t|I_t, \tau_t) p(I_t|I_{t-1}) p(\tau_t|\tau_{t-1})$$

Parameters

The model has only *number of sensors* + 2 meta parameters:

- θ_I: how much trajectories can change,
- θ_τ: how much time can get stretched,
- σ_{ζ^k}: for each sensor, indicates when two values are different.

Observation model

$$p(Z_t|I_t = i, \tau_t = j) = \prod_k \left[\int_{\zeta_{j-1}^i}^{\zeta_j^i} \frac{1}{2(\zeta_j^i - \zeta_{j-1}^i)_k} \mathcal{N}(t, \sigma_{\zeta^k}^2) dt + \int_{\zeta_j^i}^{\zeta_{j+1}^i} \frac{1}{2(\zeta_{j+1}^i - \zeta_j^i)_k} \mathcal{N}(t, \sigma_{\zeta^k}^2) dt \right]$$

Transition model

$$p(\tau_t|\tau_{t-1}) = \begin{cases} \theta_\tau & \text{if } \tau_t = \tau_{t-1} \\ 1 - 2\theta_\tau & \text{if } \tau_t = \tau_{t-1} + 1 \\ \theta_\tau & \text{if } \tau_t = \tau_{t-1} + 2 \\ 0 & \text{otherwise} \end{cases} \qquad p(I_t|I_{t-1}) = \begin{cases} 1 - \theta_I & \text{if } I_t = I_{t-1} \\ \frac{\theta_I}{N-1} & \text{otherwise} \end{cases}$$

alternatively, a Log-normal distribution

Initial conditions and termination criterion

$$p(I_0 = i, \tau_0 = j) = \begin{cases} 1/N & \text{if } j = 0 \\ 0 & \text{otherwise} \end{cases}$$

The task is considered completed if $p(\tau_t$ in last 10 time steps$) > 0.9$.

Questions

- Update due to time, involving the prediction model:

$$p(I_t, \tau_t|Z_{1:t-1}) = \sum_{I_{t-1}, \tau_{t-1}} p(I_t, \tau_t|I_{t-1}, \tau_{t-1}) p(I_{t-1}, \tau_{t-1}|Z_{1:t-1})$$
$$= \sum_{I_{t-1}} p(I_t|I_{t-1}) p(I_{t-1}|Z_{1:t-1}) \sum_{\tau_{t-1}} p(\tau_t|\tau_{t-1}) p(\tau_{t-1}|I_{t-1}, Z_{1:t-1})$$

- Generation of a command, involving a decision function:

$$p(U_t|Z_{1:t-1}) = \sum_{I_t, \tau_t} p(U_t|I_t, \tau_t) p(I_t, \tau_t|Z_{1:t-1})$$

$$D(p(U_t|Z_{1:t-1})) = \sum_{I_t, \tau_t} D(p(U_t|I_t, \tau_t)) p(I_t, \tau_t|Z_{1:t-1}) = \sum_{I_t, \tau_t} \upsilon_{\tau_t}^{I_t} p(I_t, \tau_t|Z_{1:t-1})$$

- Update due to observations, involving the observation model:

$$p(I_t, \tau_t|Z_{1:t}) \propto p(Z_t|I_t, \tau_t) p(I_t, \tau_t|Z_{1:t-1})$$

Acknowledgements: European projects: Noptilus (FP7-270180), myCopter (FP7-266470), NIFTi (FP7-247870); Willow Garage