Robot Learning: A Tutorial

Francesco Capuano **Adil Zouitine** **Pepijn Kooijmans** **Steven Palma** **Martino Russi** **Caroline Pascal** **Thomas Wolf** **Michel Aractingi**

École Normale Supérieure Paris-Saclay, Hugging Face

Abstract

Recent advancements in foundation models have enabled unprecedented progress in robotics. Leveraging learning-based approaches, the community has produced the first generalist models for tasks such as manipulation and whole-body control, with many results openly released. This work presents a comprehensive tutorial on modern robot learning, introducing the conceptual underpinnings of the most prominent approaches and pairing them with reproducible implementations using the open-source LeRobot framework. The tutorial is designed to serve both as a structured academic reference and as a practical guide for researchers in the field of robot learning.

1 Introduction

1.1 Motivation: The Interdisciplinary Nature of Robotics in the Machine Learning Era

Robotics in 2025 sits at the intersection of classical model-based control, machine perception, and large-scale machine learning. Foundational problems in locomotion, manipulation, and whole-body control demand reasoning across rigid-body dynamics, contact modeling, planning under uncertainty, and high-dimensional function approximation. At the same time, end-to-end learning has matured from proof-of-concept demonstrations to systems that benefit from internet-scale multimodal pretraining and robotics-specific fine-tuning, closing the gap between laboratory benchmarks and deployment in unstructured settings. This tutorial embraces that interdisciplinarity: it treats modern robot learning as a synthesis of classical priors and data-driven policies rather than a replacement of one by the other.

1.2 Scope and Contributions of This Work

This document serves two purposes. First, it provides a concise, academically rigorous overview of core concepts in classical robotics that remain essential for principled system design (kinematics, dynamics, planning, and control). Second, it surveys contemporary learning-based methods for robotic control, with an emphasis on (i) reinforcement learning (RL) and imitation learning (IL), (ii) single-task policy architectures such as transformer-based action chunking and diffusion policies, and (iii) multi-task vision–language–action (VLA) models. Throughout, we complement exposition with *reproducible* implementation guidance using the open-source `LeRobot` framework, to lower the barrier between concept and practice while maintaining scientific rigor.

1.3 Structure of the Report

Section 2 reviews classical robotics foundations and their limitations in contact-rich, high-DOF regimes. Section 3 formulates learning-based control via RL and IL, highlighting problem setups, algorithms, and known failure modes (e.g., reward misspecification, simulation gaps, and safety constraints). Section 4 details single-task policy families (transformer chunking, diffusion) and their practical training recipes. Section 5 surveys multi-task VLA models (e.g., RT-1/RT-2, OpenVLA, $\pi_0/\pi_{0.5}$, and SmolVLA), together with integration patterns and experimental evaluation protocols. Section 6, not covered here, discusses emerging directions (e.g., world models and post-training) beyond the present scope.

2 Classical Robotics Foundations

2.1 Core Focus Areas: Locomotion, Manipulation, and Whole-Body Control

Robotic *locomotion* concerns generating dynamically feasible motions for legged or wheeled platforms under terrain variability and contact constraints. *Manipulation* addresses grasping, non-prehensile actions, and dexterous interactions with objects, often under partial observability and frictional contact. *Whole-body control* integrates locomotion and manipulation to coordinate many degrees of freedom subject to nonholonomic constraints, torque limits, and stability criteria (e.g., ZMP, centroidal dynamics). These capabilities have historically been enabled by model-based planning and control pipelines that exploit structure in rigid-body mechanics.

2.2 The Traditional Robotics Paradigm

2.2.1 Kinematics and Dynamics Modeling

Classical formulations model robot geometry via forward and inverse kinematics and derive equations of motion using Lagrangian or Newton–Euler methods. Screw theory and $\mathrm{SE}(3)/\mathfrak{se}(3)$ provide compact Lie group representations for rigid-body motion. Accurate geometric and inertial modeling underpins control law synthesis and trajectory optimization.

2.2.2 Motion Planning and Control

Sampling-based planners (e.g., PRM, RRT*) and trajectory optimization compute collision-free, dynamically feasible references in configuration or task spaces. Tracking is achieved with controllers ranging from joint-space PID and resolved-rate schemes to inverse-dynamics control, impedance/operational-space control, and model predictive control (MPC). Whole-body MPC combines contact planning with torque-constrained optimization to satisfy friction cones, center-of-mass dynamics, and task priorities.

2.3 Limitations of Classic Robotics

Despite their maturity, purely model-based pipelines encounter difficulties in (i) modeling complex contact with stiction and stick–slip transitions, (ii) calibration drift and unmodeled compliance, (iii) scalability to high-DOF whole-body behaviors with tight feedback latencies, and (iv) brittle performance under distribution shift. These limitations motivate *learning-augmented* systems that retain geometric structure while leveraging data-driven policies to close residual gaps.

3 Learning-Based Approaches to Robotics

3.1 Reinforcement Learning (RL) for Robotic Control

3.1.1 Problem Formulation and Control Objectives

We pose control as a Markov decision process (MDP) with state space $\mathcal{S}$, action space $\mathcal{A}$, transition kernel P, reward r, and discount γ. A policy $\pi_\theta(a \mid s)$ maximizes $J(\theta) = \mathbb{E}\big[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)\big]$. Practical objectives include setpoint tracking, task success, energy regularization, and safety constraints (e.g., joint limits, collision avoidance).

3.1.2 Policy Optimization Methods in Robotics

Modern practice spans on-policy (e.g., TRPO/PPO) and off-policy (e.g., DDPG/TD3/SAC) actor–critic methods, as well as model-based RL that learns or exploits dynamics for sample efficiency. Off-policy approaches are favored for real robots due to data efficiency and the ability to incorporate replay. Auxiliary techniques—domain randomization, dynamics identification, and constraint handling—facilitate sim-to-real transfer and safe exploration.

3.1.3 Practical Implementation: Training RL Policies with `LeRobot`

`LeRobot` provides standardized dataset and model abstractions, and can be integrated with established RL backends. A typical workflow comprises: (i) environment and observation/action specification (simulated or real), (ii) policy class selection (e.g., SAC/PPO wrappers), (iii) logging and evaluation hooks, (iv) sim-to-real calibration (actuation limits, latency compensation), and (v) deployment with safety interlocks. The framework's dataset utilities simplify offline RL pretraining or behavior cloning initialization before online fine-tuning.

3.1.4 Limitations of RL in Real-World Robotics: Simulators and Reward Design

Notwithstanding successes in quadruped locomotion and manipulation, practical bottlenecks persist: expensive data collection, reward misspecification and shaping burden, partial observability, and safety during exploration. High-fidelity simulation reduces real-world trials but introduces transfer gaps; reward engineering often requires domain expertise and iterative tuning. Stable, sample-efficient, and safety-aware RL remains an active research frontier.

3.2 Imitation Learning (IL) for Robotics

3.2.1 Leveraging Real-World Demonstrations

IL bypasses online reward optimization by learning from expert trajectories. In visuomotor settings, demonstrations pair high-dimensional observations (RGB, depth, proprioception) with action sequences. Policy learning reduces to supervised learning under covariate shift, necessitating strategies that mitigate compounding errors.

3.2.2 Reward-Free Training and Data-Centric Perspectives

Data-centric IL emphasizes broad, diverse, and real-world demonstration corpora to improve generalization. Pre-training on heterogeneous demonstrations and fine-tuning on task-specific data can produce robust policies without explicit rewards, especially for contact-rich manipulation.

3.2.3 A Taxonomy of IL Approaches

Behavior Cloning (BC) minimizes one-step imitation loss; interactive IL (e.g., DAgger) aggregates data under the learner's state distribution; adversarial/apprenticeship methods recover reward surrogates; and diffusion- or flow-based action generators model multimodal action distributions. Interactive feedback and on-the-fly corrections further reduce covariate shift in deployment.

4 Single-Task Policy Architectures

4.1 Action Chunking with Transformers

4.1.1 Model Architecture and Training Objectives

Action Chunking with Transformers (ACT) models short horizons of low-level actions as fixed-length "chunks" conditioned on recent observations and proprioception. A transformer parameterizes a distribution over action sequences, enabling parallel prediction of temporally coherent control signals. Supervised training on teleoperated demonstrations minimizes sequence-level losses (e.g., mean-squared error on joint velocities/torques), optionally with scheduled sampling or consistency regularization to improve rollout stability.

4.1.2 Practical Implementation in `LeRobot`

A practical ACT pipeline comprises (i) synchronized demonstration collection (vision and robot states) with precise timestamping, (ii) dataset serialization into a standardized episodic format, (iii) chunked windowing of action targets, (iv) transformer training with early stopping on validation rollouts, and (v) deployment with action-rate limiting and safety monitors. Careful calibration of control frequency and chunk length is critical for fine manipulation.

4.2 Diffusion-Based Policy Models

4.2.1 Generative Modeling for Action Sequences

Diffusion policies treat action generation as conditional denoising: starting from noise in action space, a learned score model iteratively refines to produce feasible control sequences conditioned on observations (and optionally goals or language). This formulation naturally captures multimodality (multiple valid strategies), scales to higher-dimensional actions, and supports receding-horizon execution.

4.2.2 Practical Implementation

Training proceeds by corrupting ground-truth action sequences and fitting a time-indexed network to denoise under teacher forcing; inference uses a small number of reverse steps with horizon MPC-style rollouts. In practice, datasets benefit from action normalization, viewpoint augmentation, and careful tuning of denoising steps versus control latency. Within `LeRobot`, diffusion policy trainers can be paired with standardized dataset loaders and evaluation callbacks to track success rates and robustness under perturbations.

5 Multi-Task Policies: Vision–Language–Action (VLA) Models in Robotics

5.1 Overview of Major Architectures: RT-1, RT-2, OpenVLA, π_0, $\pi_{0.5}$, SmolVLA

VLA models integrate perception, instruction following, and action generation within a unified network. RT-1 demonstrated large-scale, robot-collected datasets and transformer policies capable of executing hundreds of distinct manipulation tasks; RT-2 extended this idea by co-training with web-scale vision–language corpora and representing actions as discrete tokens, improving semantic generalization to novel objects and instructions. OpenVLA adopts an open 7B backbone that fuses pretrained vision encoders with a language model and trains on large, diverse robot demonstrations, reporting strong cross-embodiment performance and efficient fine-tuning. The π_0 family introduces flow-matching action heads atop a pretrained VLM to produce continuous, high-frequency motor commands for dexterous skills; $\pi_{0.5}$ augments co-training for broader open-world generalization. Complementing these, SmolVLA targets efficiency and accessibility with a compact ($\sim 4.5 \times 10^8$ parameters) architecture and an interleaved attention "action expert" for low-latency control.

5.3 Practical Implementation: Integrating VLAs with `LeRobot`

Integration typically follows three phases. *Pretraining:* initialize the VLM backbone (e.g., SigLIP/DINOv2 features) and attach an action head (autoregressive, diffusion, or flow-matching) trained on heterogeneous demonstrations curated in a standardized format. *Task adaptation:* fine-tune with instruction-augmented, robot-specific data using low-rank adaptation where possible; calibrate action scaling, control-rate chunking, and observation encodings. *Deployment:* quantize or distill for on-robot inference; implement asynchronous inference stacks to decouple perception and control loops; add safety fallbacks and confidence gating. `LeRobot` dataset and model utilities simplify all three phases by providing consistent IO, evaluation tools, and model hosting.

5.4 Experimental Evaluation

A principled evaluation protocol should measure (i) success rate across held-out tasks and objects, (ii) language grounding fidelity under paraphrases and compositional instructions, (iii) cross-embodiment transfer (train/test robot mismatch), and (iv) robustness to distribution shift (lighting, clutter, distractors). Where possible, report zero-shot and few-shot adaptation performance, ablate pretraining datasets and backbones, and include real-world trials with fixed seeds and video evidence. Public, diverse datasets and standardized success criteria are critical to comparable results.

References