JOHNS HOPKINS
WHITING SCHOOL
of ENGINEERING

EN.601.498 / EN.601.698

Hands-on Robot Learning

Fall 2026 · 3 credits

Course logistics

When?	Thursdays, 3:00–5:30 PM
Where?	Ames 234
Instructor	Krishna Murthy Jatavallabhula — kmj@jhu.edu
Instructor's office	Hackerman 226E
Office hour(s)	Thursdays, 12:00–1:00 PM (Hackerman 226E), please email ahead of time to confirm
Course assistants	Xinyi Yin, Yinzhe Zhou, Aleks Santari (office hours to be announced)
Course website	https://hands-on-robot-learning.github.io/
Canvas	Assignments, readings, submissions, grades, and announcements

About this course

Robot learning has made significant strides in just the last 2 years. For the first time, there exist 'generalist' manipulation systems that one can download from the web, and deploy on their robots and environments without substantial finetuning, and achieve non-trivial task success rates. At the same time, making robot learning models (a.k.a. 'policies') work, and generalize across a fairly broad set of tasks and environments, is an open research problem. This has caused an undeniable shift in the skillsets that robotics practitioners (engineers, scientists) must now have, to get into (and succeed at) frontier robotics labs.

This course aims to address that gap. In this class, we will take a 'full-stack' perspective of robot learning – we will understand (and build) robot hardware (i.e., a small tabletop robot arm) and discuss the entirety of the modern robot learning lifecycle. This includes task design and selection, curating and processing data, training neural network models on the processed data, deploying these trained models, and evaluating how the trained approaches fare.

This is not an introductory class. We expect students to already have requisite knowledge in deep learning (training and debugging transformer models, diffusion models) or be able to pick them up independently. We will do an extremely short recap of the essentials for the class (one 2.5-hour lecture). We then will look at various data curation and processing strategies, single-

task imitation learning, multi-task policy learning, and will discuss some of the frontiers (world models, for example).

This is a collaborative class. Assignments will be completed in teams of about 3 students each. Each team will receive an SO-101 robot arm kit and will assemble their robot from scratch. There are no written exams.

Roughly half of your time in this course will be spent outside the classroom, building, collecting data on, deploying policies on, and debugging, the SO-101 robot arm. Robot hardware work is *hard*; that is a large part of what this course is about. Brace for hardware and/or software failures happening frequently, and inevitably before/during deadlines/demos.

Prerequisites

Required: Introduction to Machine Learning (EN.601.475/675) or Deep Learning (EN.601.482/682).

You will train, debug, and evaluate transformer and diffusion models on data you will curate yourself. You should already be comfortable with the deep learning workflow: data processing and loading, model specification, training and validation, and deploying a trained network for inference. We will not teach this in class.

Preferred: Introduction to Robotics or equivalent (EN.601.463, EN.601.495, EN.530.420). Required for undergraduates.

This is not an introductory robotics course. We will recap some concepts in Week 2, but you should be familiar with robot kinematics (forward and inverse kinematics), rigid body transforms, position and velocity control, joint vs end-effector control. We will offer basic safety guidance (electronic as well as mechanical) when using the robots that's specific to SO-101 robots we'll use in class.

Course goals

By the end of the course, you should be able to:

- Describe the modern landscape of data-driven robot learning and the common system designs used in robot learning.
- Assemble, calibrate, and safely operate a robotic manipulator built from readily available electromechanical components.
- Implement, train, and evaluate modern imitation learning approaches for simple manipulation tasks.
- Deploy a vision-language-action (VLA) model on a physical robot and characterize its success and failure modes.
- Compare the tradeoffs between policy architectures, low-level design choices, and data collection methods.

- Define evaluation metrics and failure taxonomies, and conduct reproducible robot experiments.

Tentative schedule

Topics and dates below are the current plan. Guest lectures, hardware realities, and how quickly the class moves will change some of this. Readings and assignment deadlines are posted on Canvas and on the course website.

Date	Topic
Module 1 — Introduction to robot learning	
Sep 3	Modern robot learning: an overview. What changed relative to a decade ago, what a full data-driven robotics stack looks like end to end, and what this course expects of you.
Sep 10	Essentials recap: kinematics, control, LeRobot, and the SO-101. Rigid body transforms, forward and inverse kinematics, joint versus end-effector control, and the low-level controller design choices that matter later.
Module 2 — Data for robot learning	
Sep 17	Teleoperation and data sources. Kinesthetic teaching and why it fails in its naive form, leader-follower and VR interfaces, in-the-wild collection with universal interfaces, and what makes a demonstration dataset useful.
Sep 24	SO-101 assembly and calibration (in-class hands-on session).
Module 3 — Single-task policy learning	
Oct 1	Behavior cloning and action chunking. Imitation learning basics, covariate shift and compounding error, interactive imitation learning, and the action chunking transformer (ACT).
Oct 8	Generative policies and diffusion models. Diffusion models, and how they are applied to policy learning.
Oct 15	The policy inference stack. Chunk size and execution frequency, inference latency, real-time action chunking, temporal inconsistency, and interactions with the low-level controller.
Oct 22	**Fall break — no class.**
Module 4 — Generalist policy learning	
Oct 29	Vision-language-action (VLA) models. From LLMs to VLMs to VLAs; early VLA architectures and their failure modes; the first successfully deployed VLA recipes.
Nov 5	VLA models, continued. Two-stage VLA case studies from industry, and VLA post-training.

Nov 12	Coding agents as policies.
Nov 19	World models and policy learning. Learned dynamics models, and what they buy you for robot policies.
Nov 26	**Fall recess — no class.**
Module 5 — Evaluation and research frontiers	
Dec 3	Evaluating generalist robot models. Evaluation protocols, statistical significance, distributed and peer evaluation, and evaluation in simulation and in world models.
Dec 10	Final demos (Assignment 3 due)

Assignments and grading

There are no written exams. We have four assignments, completed in teams of up to 3 students each. Together, A0–A3 make up 80% of the course grade. Assignments receive a team score based on the quality of the submitted system, experiments, analysis, and documentation. In cases of substantially uneven contribution or demonstrated understanding, an individual's assignment grade may be adjusted away from the team score. The remaining 20% of the course grade reflects class participation and individual contribution to the team.

	Scope	**Weight**
A0	Assemble the SO-101, configure the motors, calibrate the arm, and document a working setup.	5%
A1	Implement and evaluate a manipulation skill using an explicit perception-and-control pipeline.	10%
A2	Collect demonstrations, train a task-specific policy, deploy it, and analyze its behavior.	30%
A3	Train or adapt a policy across multiple tasks and evaluate where transfer helps or fails.	35%
	Class participation, Individual contribution to the team	20%

Every assignment is submitted with a short report (<1 page unless specified otherwise), codebase, and more importantly data and evaluation logs, which include unedited videos of the robot runs you are reporting.

How grading works

A team grade won’t translate to an individual grade if contribution is highly uneven.

Most of the work in this course is collaborative, because building and running a robot learning workflow in most frontier industry labs is a huge team effort. Assignments are carried out in

teams (usually of size 3). A successful assignment that receives a full grade is not necessarily one where the robot does all tasks with high success rates (the course staff will be dubious if this happens, since we'd expect success rates for most task to range between 30 and 65%, esp. if using the methods suggested for each assignment). Rather, successful teams build systems that are well-thought and more importantly, they're able to explain every single low-level decision that was made (e.g., why did we collect data in a specific manner? how many training episodes were collected? how was evaluation carried out? what model architecture and training hyperparameters were used and why?). The latter type of team is more likely to receive a higher grade than a team that gets everything working on the robot but cannot provide satisfactory explanations for low-level details. (As you will read through in our AI policy, you are free to use any AI tooling for this class, but you will be accountable for all code, design choices, and consequences (including robot damage) the AI tools produce.) We will evaluate the overall quality of the system produced by the team, whether the method is technically sound, how the evaluation was designed and carried out, what the analysis says, and whether someone else could reproduce it from what you submitted. That forms the basis of scoring everyone in the team.

We will test whether each person contributed substantially to the work and can explain every low-level design choice for that assignment. We also will gather (confidential) individual feedback from team members about their peers. Presuming everyone contributed meaningfully, the team grade gets translated to individual grades for that assignment. This individual contribution / adjustment is primarily intended for teams where an individual does far less work than their teammates, or where just one individual does all the work. If three people are working reasonably equitably, we are not interested in establishing whether someone contributed 30% or 37%. You get the same grade.

Note: At some level, specialization is expected and is welcome. The course staff realizes that robotics is inherently interdisciplinary, and that each team member may potentially be excited about one (or some) aspects of robotics more than the other. One team member may end up obsessing over model training hyperparameters, while another may be more involved in data pre-/post-processing. That (in our view) is a healthy and functioning team. What we do ask is that everyone understands the whole system at a reasonable level. "I only worked on the cameras, so I do not know how our policy works" is not an acceptable answer in a demo or a check-in.

What we look at

We use several signals when translating a team grade to an individual grade:

- The contribution statement submitted with each assignment. (This contribution statement is made by the team as a whole – as such, all team members will be able to access this. See further below for an example contribution statement).
- Artifacts produced per team member: datasets collected, experiments run, evaluation logs, analysis, documentation.
- Demos, check-ins, and conversations with the course staff during assignment demos.

- Peer and self assessment at the end of each assignment.

Grading FAQs

Does everyone on a team normally get the same assignment grade?
Nominally, yes. However, if there is evidence of substantial inequity in workload within a team, individual grades may be adjusted away from the team grade.

What if one person does much less work?
The team artifact can still earn a strong score. That person's individual grade may be adjusted based on contribution and demonstrated understanding.

What if one person does almost everything?
That is also not a successful team outcome. Part of what you are learning here is how to distribute work on a system that has hardware, data, training, and evaluation in it. Tell us early rather than after the assignment is over.

Can a teammate who is not contributing lower my grade?
Their lack of contribution should not lower your individual assessment. It is still a team assignment, though, so the quality of what the team submits sets the common baseline. This is another reason to raise problems early instead of absorbing the work silently for three weeks.

Does every team member have to be equally good at everything?
No. See above: specialization is fine, and to some degree, expected.

Teamwork and collaboration policies

Team formation

Teams of (usually 3 students) will share one robot kit for the duration of the course. Robot kits will be given out late September, and will be returned at the end of the last lecture of the semester (which is, in fact, a demo session). Teams are assigned by the course staff. Given the heterogeneous mix of skill sets and learning objectives across the entire class, we will not be able to accommodate all of your teaming requests. However, we will attempt to accommodate them to some extent – for this, please fill out the **background and teaming questionnaire** on Canvas before the end of class on Week 2. Failing to do so means you get assigned randomly into a team.

When assigning teams, we look for a few criteria (here's your cue to influence our teaming process, if you've read this detail closely): enough common meeting times outside class (most assignment work is done outside of class), trying to have teams that have breadth across electronics/mechanica/learning/systems, complementary learning goals, and reasonable compute access across the class. You may name 3-4 people you would especially like to team up with. Additionally, you may confidentially name anyone you should not be placed with. The questionnaires are confidential and only accessed by the course staff and you.

Teams, once formed, remain the same across the semester. In case of serious, irreparable collaboration issues, please see the team staff *well in advance*.

Contribution statement

Assignments *require* a contribution statement that tells us *who-did-what*. It can be extremely brief – e.g., the following contrib statement is perfectly acceptable:

- Alice: teleoperation interface and data collection
- Bob: model training and evaluation infrastructure
- Charlie: deployment and analysis w.r.t. baselines
- All: task design, robot experiments, debugging, report

Peer- and self-assessment

After each assignment, team members will answer a short questionnaire that serves as a self-assessment (i.e., how much did you think you contributed to the team) and a peer-assessment (i.e., how much did you think your peers contributed). We'd assess whether you felt work was distributed reasonably, everyone contributed meaningfully, and if there's anything the course staff privately would need to know about. (you won't be asked for an exact percentage split/share of work each of the team members did)

When a team isn't working for you

Communicate. Early and often. This shouldn't be the week/day before a major milestone or demo is due. Don't silently absorb someone else's share of work and then disclose that in the final peer assessment. Talk to the instructor or the course staff (preferably in-person, instead of trying to sort all this out over an email thread).

Working with other teams

Talking to other teams is encouraged; so is helping them with simple debugging (e.g., hardware/software troubleshooting). What you submit must be your team's own work: your own data, your own training runs, your own evaluations, your own writing. Sharing data, code, reports, results, and other artifacts is a violation of academic integrity. If another team helped you substantially with something, say so in your report to ensure you duly credit them.

Robot kits and your responsibilities

Each team receives one SO-101 kit in late-September. It stays with your team for the rest of the semester – it is returned at the end of the final demo in December, and the kit will be reused in future offerings of this class. The robot kits are a shared resource - please treat them that way.

Team Responsibility

All team members are jointly responsible for their robot kit. In particular, we require teams to:

- keep the robot, cameras, power supplies reasonably organized
- maintain current calibration and config files to ensure the robot runs

- periodically inspect the robot for any wear and tear (loose cables, overheating motors, excessive jitter, or anything else out of the ordinary)
- report missing and/or malfunctioning behaviors / parts promptly so the course staff may address it
- store and transport the robot safely (e.g., don't let it soak in rain or snow)
- never exceed the operating limits of the robot (e.g., payload, temperature, duration of operation, voltage levels, etc.)

This said, robots are physical systems, and will experience nominal wear-and-tear over the course of the semester. Ordinary wear-and-tear during responsible use is expected. Robots break. Ordinary wear, hardware failures, and accidental damage during reasonable course use are expected and, by themselves, won't affect grades. But unsafe use e.g., exceeding operating limits, voltages, stalling, unauthorized modifications, excessive payloads, etc. are considered willful negligence. This includes cases where the behaviors are generated by AI agents – the team is responsible for testing and auditing AI-generated code before running it on the robot.

Ensuring the robot is safely used

While we will discuss more about this in class, please bear the following in mind when working with the SO-101 robot arm.

- Secure the base to the table before running anything. Else, the arm can fall, and the fall can damage motors or links.
- Keep the power switch or cable within reach whenever the arm is powered, and know how to kill power instantly.
- Keep hands, hair, cables, and liquids (e.g., coffee) out of the workspace during autonomous runs.
- Never command the robot beyond its intended joint limits or operating range.
- Do not rewire the robot while the arm is powered. Wiring faults are the most likely way to burn a servo.
- Do not attempt to stall the motor. Do not hold the motor shafts or links with your hand to prevent movement. If the robot is pushing into an object but is stalling, cut the power immediately to prevent overheating and motor burnout.
- Do not leave the robot powered on for long periods of time – the SO-101 motors will heat up and be damaged if left idling.
- Do not leave the robot kit unattended.
- Stop and ask if something smells hot, gets hot, buzzes, or behaves in a way you cannot explain.
- Report any injury, however minor, to the course staff.

Compute

Assignments are designed to be completed with modest GPU access: smaller models, efficient fine-tuning, and training runs that fit within limited resources. The course staff is unable to provide additional compute for training models. We encourage students to avail free education-tier tooling on Google Colab, Kaggle, HuggingFace, or similar providers. If you have access to other compute that you are permitted to use for coursework, that's allowed.

Use of AI tools

You are free to use any AI tools you like, at any stage of your work: coding assistants, chat models, agents, whatever helps. We do not consider this cheating and won't penalize this.

That said, we ask that you:

1. **Disclose it.** Every submission should include a short AI use statement: which tools you used, and what you used them for. A few sentences is enough. "We used a coding assistant to write up much of the code, and an LLM to brainstorm potential ideas" is a perfectly acceptable statement.
2. **Own what you submit.** You are responsible for every line of code, every number, and every claim in your report, regardless of what produced it. If you cannot explain your own submission, reproduce your own result, or defend a claim you made, that will be penalized. Additionally, you will also be held accountable for any damage to the robot hardware caused by the use of AI tools (e.g., LLM-generated code that violates joint limits and ends up burning your robot's motors).

Do not paste privileged information or personally identifiable data into third-party tools. This includes video and images of people who have not consented to it (something to factor in when collecting datasets for this class).

Academic integrity and Ethics

The strength of the university depends on academic and personal integrity. In this course, you must be honest and truthful, abiding by the Computer Science Academic Integrity Policy.

Cheating is wrong. Cheating hurts our community by undermining academic integrity, creating mistrust, and fostering unfair competition. The university will punish cheaters with failure on an assignment, failure in a course, permanent transcript notation, suspension, and/or expulsion. Offenses may be reported to medical, law or other professional or graduate schools when a cheater applies.

Violations can include cheating, plagiarism, reuse of assignments without permission, unauthorized collaboration, alteration of graded assignments, forgery and falsification, lying, facilitating academic dishonesty, and unfair competition. Ignorance of these rules is not an excuse. The university will punish cheaters with failure on an assignment, failure in a course, permanent transcript notation, suspension, and/or expulsion.

This course is built almost entirely on group work. Cite any external code, model, or resource you use. Your submissions must list everyone who participated.

Reporting results / Scientific integrity. In a hands-on course, reporting a number you did not actually get on the robot amounts to cheating and academic dishonesty. To be explicit:

- Falsifying success rates or any other reported metric is academic dishonesty.
- Intervening manually during an autonomous evaluation run, and reporting that run as autonomous, is academic dishonesty.
- Evaluation videos must be unedited, continuous shots. Cutting a video to hide a failure or a reset is academic dishonesty.
- Reporting results from a run you cannot reproduce or did not perform is academic dishonesty.

A policy with a 20% success rate, honestly measured and carefully analyzed, will earn a better grade in this course than a claimed 90% we cannot reproduce. Negative results, when well-documented, are valuable and interesting; we would much rather prefer you report your current results and reasoning behind your model's performance – as long as reasonable effort was made to get the system to address the tasks outlined in the assignment, we'd value that.

Report any violations you witness to the instructor. More information on university misconduct policies:

- Undergraduates: https://studentaffairs.jhu.edu/policies-guidelines/undergrad-ethics/
- Graduate students: http://e-catalog.jhu.edu/grad-students/graduate-specific-policies/

Attendance, deadlines, and late work

We do not have an official attendance requirement (but that shouldn't mean you shouldn't attend class). We will cover a lot of ground in class – this class will be challenging, yet rewarding. We do not have a routine late-work grace period. Most assignments culminate in scheduled demos, which makes moving deadlines difficult. Exceptions for SDS accommodations, illness, emergencies, or other genuine circumstances will be handled case-by-case. Talk to the instructor as early as possible; we would much rather hear about a problem before a deadline than after.

Illness, emergencies, and everything else. Talk to the instructor. Extensions for genuine reasons are not a problem, and we would much rather hear about a situation before the deadline than after it.

Robot damage before a deadline. Talk to the instructor. The course staff will investigate and determine the nature and extent of damage. Intentional or negligent damage may incur a penalty on the assignment grade for the team. Teams are all required to adhere to robot safety and usage policies outlined herein.

Accommodations and support

If you need accommodations, register with JHU Student Disability Services and let me know early in the semester so that we can make arrangements. Requests made through SDS are handled confidentially.

The physical nature of this course may raise more accessibility questions than the course staff have thought about. If any of this is a concern, talk to the instructor and we will work it out. If you are struggling, for any reason, please reach out — to the instructor, to the course assistants, or to your academic advisor.