6.421 Robotic Manipulation Midterm

Gatlen Culp · Wednesday, 2025 Nov 05

Ch. 01: Introduction

High-level goals, scope, and examples

Overview

- **Core challenges:** Contact-rich dynamics; integrate perception, planning, and control.
- **Beyond pick-and-place:** Task diversity; suction common but insufficient for many tasks.
- **Tools today:** Strong rendering and contact simulation; runnable notebooks for practice.
- **Systems view:** Components/diagrams; ROS message contracts vs Drake state/timing semantics; visualize system graphs for profiling.

Ch. 02: Robot Setup

Standard models enable reuse across robots

Essentials

- **Formats:** URDF/SDF (primary), limited MJCF; Drake Model Directives (YAML) for multi-robot composition/edits.
- **Best practices:** One source of truth for kinematics/inertias/visuals/collisions; validate in visualizer.
- **Frames:** Clear base/eef/camera frames simplify IK and calibration.

Modeling Details

Inertias: Use consistent units; verify link mass m, center c, and inertia matrix about c are physical (symmetric, positive definite).

Collision vs visual: Keep collision simple/convex where possible; ensure no self-collisions in nominal poses.

Limits: Encode $q \in [q_{\min}, q_{\max}]$, $|\dot{q}| \le \dot{q}_{\max}$, $|\tau| \le \tau_{\max}$; validate in the visualizer.

Self-checks: Spawn gravity-only sim $\longrightarrow$ verify $\tau_g(q)$ balancing; drop tests for contacts and restitution.

Control interfaces and why torque sensing matters

Position-Controlled Robots

Definition: Track joint positions/trajectories precisely; typical when torque interface is unavailable.

Why common: Electric motors with large gear reductions break simple $\tau \leftarrow k_t i$ due to backlash, friction, and unmodeled transmission dynamics $\longrightarrow$ robust torque control is hard; closed-loop position control is easier.

Torque-Controlled Robots

Capability: Joint-torque sensing + high-rate torque commands enables compliant behaviors, contact-rich tasks, and force-control.

Example platform: KUKA LBR iiwa used throughout notes for torque control experiments.

Practical Guidance

If you have torque control: You can still do high-accuracy position control; prefer torque mode when contact/stiffness/compliance matters.

If you only have position control: Use impedance/stiffness at trajectory level, add compliance via end-effector mechanisms, and favor contact-robust strategies.

Transmission & Reflected Dynamics

Motor current to torque: $\tau_{\text{motor}} \coloneqq k_t i$ (ideal).

With gear ratio N and efficiency η: $\tau_{\text{joint}} = \eta N \tau_{\text{motor}} - \tau_{\text{fric}}$.

Reflected load: $J_{\text{reflected}} \coloneqq N^2 J_{\text{load}}$, $b_{\text{reflected}} \coloneqq N^2 b_{\text{load}}$; large N amplifies unmodeled dynamics (backlash, friction).

Impedance over Position Interface

Joint-space impedance: $\tau_{\text{ref}} \coloneqq K_p(q_{\text{des}} - q) + K_d(\dot{q}_{\text{des}} - \dot{q}) + K_i E_q + \tau_{\text{ff}}$ where E_q accumulates error. Position-only APIs approximate this by shaping commanded trajectories and internal gains.

Cartesian impedance: $f \coloneqq K_x(x_{\text{des}} - x) + D_x(\dot{x}_{\text{des}} - \dot{x})$, $\tau_{\text{cmd}} \leftarrow J^\top f + N^\top \tau_{\text{null}}$.

Torque-Control Architecture

Gravity compensation: $\tau_g(q)$ added to improve passivity and reduce effort.

Full command: $\tau_{\text{cmd}} \leftarrow \tau_g(q) + \tau_{\text{impedance}} + \tau_{\text{ff}}$; respect $\|\tau_{\text{cmd}}\|_\infty \le \tau_{\max}$ and rate limits.

Trade-offs among dexterous, simple, and special-purpose grippers

Dexterous Hands

Pros: In-hand manipulation, rich contact modalities, versatile.

Cons: Complex control, sensing, calibration; lower reliability in clutter; higher cost.

Simple/Underactuated Grippers

Pros: Robust, cheap, tolerant to pose error; emergent adaptability (underactuation/compliance).

Cons: Limited in-hand reorientation; rely on environment for dexterity.

Suction and Specialized Tools

Suction: Excellent for flat or sealed surfaces; struggles on porous/rough geometry; add sensors for seal detection.

Tooling: Choose end-effectors per task physics (pinch, power grasp, hooks, spatulas) and expected object set.

Contact Modeling Notes

Friction: Use Coulomb cone approximations in planners; account for stick/slide transitions in controllers.

Suction: Seal depends on surface curvature/roughness; add vacuum sensing; treat payload as $w \coloneqq mg$ with margin for accelerations.

Compliance: Underactuation/compliant pads widen successful grasp set but reduce precise in-hand dexterity.

Perception and proprioception for manipulation

Exteroception

RGB/Depth/Cameras: Pose estimation, scene understanding; mount on wrist and/or fixed viewpoints; mind occlusions and calibration.

Tactile/Proximity: Detect incipient contact, slip, and shear; improves grasp reliability and insertion tasks.

Proprioception

Joint encoders & velocities: Core for control/estimation.

Joint torque/force sensors: Enable model-based force control and safe contact.

FT sensor at wrist: Measures interaction wrench; useful for hybrid position/force tasks.

Calibration & Noise

Hand–eye (eye-in-hand): Solve $A_i X = X B_i$ for camera–eef transform X from pairs of robot motions (A_i) and observed camera motions (B_i).

Noise models: $z \coloneqq h(x) + v$, $v \coloneqq\in \mathcal{N}(0, \Sigma)$; propagate to pose and contact estimation; filter with complementary/EKF as needed.

Time sync: Align sensor timestamps with controller loop to avoid phase lag in feedback.

From models to runnable systems

HardwareStation

Purpose: Defines robot(s), sensors, controllers, scene objects, and wiring in one place; consistent interfaces for sim and real.

Usage: Load description files + directives, set controllers (position/torque), expose ports for commands and measurements.

Simulation

Modern rendering: Synthetic images can test/train perception with real-world transfer.

Contact simulation: Improved solvers make multi-body contact practical; validate grasp/contact strategies in silico before hardware.

Workflow Tips

Unified config: Keep station configs shared across sim/real to minimize drift.

Safety first: Rate-limit torques/velocities; test contact behaviors in sim; bring up with high damping/compliance.

Ports & Rates

Inputs: q_{des}, $\dot{q}_{\text{des}}$, τ_{cmd} (mode-dependent). **Outputs:** $q, \dot{q}, \tau$, wrench, images, depth.

Rates: Control (1–2 kHz torque, 250–500 Hz position), perception (30–60 Hz RGB, 10–30 Hz depth); buffer and decimate appropriately.

Bring-Up Checklist

Soft limits: Enforce $q, \dot{q}, \tau$ bounds and collision margins.

Validation: Gravity comp on $\longrightarrow$ move slowly $\longrightarrow$ contact probing at low stiffness $\longrightarrow$ task execution with logging.

Key facts/settings commonly queried in PS01

IIWA14 Facts

Joint count: 7; **Link count (incl. base & eef):** 8; **Joint type:** revolute; **Control:** torque.

Position Controller Setup

Gain: $K_p \approx 100$–200 typical for settling (PS01 uses 120). Tune K_d/K_i as needed, or omit.

Command: Set q_{des}; controller computes τ_{ref}; plant applies τ_{cmd} after gravity/limits.

Initial conditions: Example $q_0 \coloneqq [0.2, 0.2, 0.2, 0, 0, 0, 0]$; $q_{\text{des}} \coloneqq [0, 0, 0, 0, 0, 0, 0]$; simulate for $T \coloneqq 10$ s, then read final $q(T)$.

Systems & Drake Features

Block-diagram systems, **optimization toolkit**, **multi-body dynamics**, **deterministic replay**; not GPU-parallel by default.

HardwareStation Playback

Usage: Launch station (sim or real config), wire controllers/sensors, run for horizon T, log ports; export a screen recording for verification tasks.

Ch. 03: Kinematics & Pick-and-Place

Frames, positions, rotations, transforms

Frames, Points, Positions

Positions: Use monogram with attach.

$$^{A}p^{C}_{F} \coloneqq \text{pos of C measured from A, expressed in F}$$

. Shorthands: if expressed-in equals measured-from, drop subscript. If measured-from is W, drop tl.

Rotations:

$$^{A}R^{B} \coloneqq \text{orient of B measured from A}$$

. Composition/inverse: $^{A}R^{B}\,^{B}R^{C} = {}^{A}R^{C}$; $\left(^{A}R^{B}\right)^{-1} = {}^{B}R^{A}$.

Transforms (poses): $^{A}X^{B}$ bundles translation+rotation. Position/composition: $^{G}p^{A} = {}^{G}X^{F}\,^{F}p^{A}$; $^{A}X^{B}\,^{B}X^{C} = {}^{A}X^{C}$.

Camera-to-World Conversion

If camera frame C has pose $^{W}X^{C}$, a camera point $^{C}p^{P_i}$ maps as $^{W}p^{P_i} = {}^{W}X^{C}\,^{C}p^{P_i}$ with inverse extrinsics $^{C}X^{W}$.

Kinematic tree, frame composition, representations

Kinematic Tree and Joint Frames

Each joint defines $^{J_P}X^{J_C}(q)$ and fixed offsets $^{P}X^{J_P}$, $^{J_C}X^{C}$. Between parent P and child C: $^{P}X^{C}(q) = {}^{P}X^{J_P}\,^{J_P}X^{J_C}(q)\,^{J_C}X^{C}$.

Goal (gripper pose): $X^G \coloneqq f^G_{\text{kin}}(q)$ via recursive composition.

3D Rotation Representations

Rotation matrices, roll-pitch-yaw (RPY), axis-angle, and unit quaternions have trade-offs (RPY gimbal lock at pitch = pi/2). Use quaternions for representation; use angular velocity for derivatives.

Generalized Velocities

Do not assume $\dot{q} = v$. Floating-base often uses quaternions in q and angular velocities in v. Drake: `MapQDotToVelocity`, `MapVelocityToQDot`.

Spatial velocity, geometric vs analytic Jacobians

Spatial Velocity

6D twist: $^{A}V^{B}_{C} \coloneqq \left[{}^{A}\omega^{B}_{C};\quad {}^{A}v^{B}_{C}\right] \in \mathbb{R}^6$. Change of expressed-in frame: $^{A}\omega^{B}_{G} = {}^{G}R^{F}\,^{A}\omega^{B}_{F}$; $^{A}v^{B}_{G} = {}^{G}R^{F}\,^{A}v^{B}_{F}$.

Jacobian Mapping

Geometric Jacobian (w.r.t. generalized velocity v): $^{W}V^{G} = J^G(q)v$. Available: `CalcJacobianAngularVelocity`, `CalcJacobianTranslationalVelocity`, `CalcJacobianSpatialVelocity` (choose w.r.t. $\dot{q}$ or v).

Singularities & Manipulability

Track $\sigma_{\min}(J)$; near-zero inflates $(J)^\dagger$. Kinematic singularities when $\operatorname{rank}(J) < 6$. Manipulability ellipsoid: image of unit ball in joint-velocity space through J.

Pseudo-inverse, QP with bounds, joint centering, pose tracking

Pseudo-inverse as Least Squares

Unconstrained least-squares: $v^\star \coloneqq \arg\min_v \left\|J^G(q)v - {}^{W}V^{G}_{d}\right\|_2^2 = \left(J^G\right)^\dagger\,{}^{W}V^{G}_{d}$.

Minimum-norm when solutions are non-unique; least-squares when overconstrained.

Normal least squares (minimize $\|Ax - b\|_2^2$)

- If A has full column rank: solve $A^\top A x = A^\top b$ (e.g., Cholesky) $\Longrightarrow x = \left(A^\top A\right)^{-1} A^\top b$.
- Equivalently, $x = A^\dagger b$ with $A^\dagger \coloneqq \left(A^\top A\right)^{-1} A^\top$.

Velocity, Position, Acceleration Constraints (QP)

With step h and measured (q, v), solve at each control step: $\min_{v_n} \left\|J^G(q)v_n - {}^{W}V^{G}_{d}\right\|_2^2$ s.t. $v_{\min} \le v_n \le v_{\max}$, $q_{\min} \le q + h v_n \le q_{\max}$, $\dot{v}_{\min} \le \frac{v_n - v}{h} \le \dot{v}_{\max}$.

Convex QP improves robustness near limits and singularities.

Joint Centering (Nullspace)

Add a secondary objective projecting into nullspace $P(q)$ of $J(q)$:

$$\min_{v_n} \left\|J^G(q)v_n - {}^{W}V^{G}_{d}\right\|_2^2 + \varepsilon \|P(q)(v_n - K(q_0 - q))\|_2^2, \quad \varepsilon \langle 1$$

Yields a unique solution and draws joints toward nominal q_0 without disturbing primary task when unconstrained.

Tracking a Desired Pose via Velocity

Convert pose target $^{A}X^{G}_{d}$ to twist: $^{A}v^{G}_{A} = \frac{^{A}p^{G}_{d(A)} - {}^{A}p^{G}_{A}}{h}$, $^{A}\omega^{G}_{A} = \left(\frac{1}{h}\right) \operatorname{axisangle}\left(^{G}R^{G}_{d}\right)$. Feed as $^{A}V^{G}_{d}$ to the QP.

Integrability & Alternatives

Pseudo-inverse paths can be non-integrable (closed task-space loop may end at different q). An alternative imposes directional consistency: $\max_{v_n, \alpha} \alpha$ s.t. $J^G(q)v_n = \alpha\,{}^{W}V^{G}_{d}$, $0 \le \alpha \le 1$. Or component-wise scaling in RPY+XYZ via a coordinate map $E(q)$ for teleop.

Keyframes, interpolation, differentiation

Keyframes

Choose gripper keyframe poses and times: initial, pregrasp, grasp, postgrasp, clearance, preplace, place, postplace.

Interpolation

Positions: First-order hold (piecewise linear). **Orientations:** SLERP with unit quaternions: $\operatorname{slerp}(q_0, q_1; t) \coloneqq \left(\frac{\sin((1-t)\theta)}{\sin(\theta)}\right) q_0 + \left(\frac{\sin(t\theta)}{\sin(\theta)}\right) q_1$, where $\theta \coloneqq \arccos(\langle q_0, q_1 \rangle)$.

Differentiate to Velocity

`PiecewisePose` differentiates to a twist trajectory $^{W}V^{G}(t)$ used by the differential IK controller.

Putting it together

Grasp/Pregrasp via Transforms

With known $^{W}X^{O}$, choose relative gripper frames $^{O}X^{G_{\text{grasp}}}$, $^{O}X^{G_{\text{pregrasp}}}$ with straight approach/retract. Example offsets (m): $^{G_{\text{grasp}}}p^{O} \coloneqq [0, 0.11, 0]^\top$, $^{G_{\text{pregrasp}}}p^{O} \coloneqq [0, 0.2, 0]^\top$; orientation via $\text{MakeXRotation}\left(\frac{\pi}{2}\right)\ \text{MakeZRotation}\left(\frac{\pi}{2}\right)$.

Control

System diagram: keyframe trajectory -> pose-to-velocity -> differential IK QP (limits + nullspace) -> joint velocity -> integrate to joint position for the low-level position controller.

Ch. 04: Geometric Pose Estimation

Pinhole Camera Model

Projection/back-projection and RGB-D specifics Given pixel coords (u, v),

Projection: For $\boldsymbol{P}^C :\in \mathbb{R}^3 : \quad [u, v, 1]^\top = \left(\frac{1}{z}\right) K [x, y, z]^\top$, camera intrinsics $K := \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix}$.

Back-Projection: Given (u, v) and depth d: $\boldsymbol{P}^C := d K^{-1} [u, v, 1]^\top$.

Representations

Data structures & frame annotation

- Depth image $D(u, v)$; Point cloud $\mathcal{P} := \left\{\boldsymbol{P}^C\right\}$; Scene points with frame labels

Rigid Alignment on SE(3)

Point-set registration in 3D **Problem:** Given pairs $(\boldsymbol{p}_i, \boldsymbol{q}_i)$ in $\mathbb{R}^3$,

$$\min_{R :\in \mathrm{SO}(3), \boldsymbol{t} :\in \mathbb{R}^3} \sum_{i=1}^{N} \|R\boldsymbol{p}_i + \boldsymbol{t} - \boldsymbol{q}_i\|^2$$

Solution (Kabsch/Umeyama): center points $\overline{\boldsymbol{p}}, \overline{\boldsymbol{q}}$; form $\tilde{\boldsymbol{p}}_i, \tilde{\boldsymbol{q}}_i$; build $W := \sum_{i=1}^{N} \tilde{\boldsymbol{q}}_i \tilde{\boldsymbol{p}}_i^\top$; SVD $W = U \Sigma V^\top$; set $R^\star := U \ \mathrm{diag}(1, 1, \det(UV^\top)) V^\top$, $\boldsymbol{t}^\star := \overline{\boldsymbol{q}} - R^\star \overline{\boldsymbol{p}}$. Weighted: use $W := \sum_i w_i \tilde{\boldsymbol{q}}_i \tilde{\boldsymbol{p}}_i^\top$ and weighted means.

Point-to-Plane (small-angle approx)

Linearized rotation update for faster convergence

Given target normals $\boldsymbol{n}_i$ at $\boldsymbol{q}_i$: $\min_{R, \boldsymbol{t}} \sum_i \left(\boldsymbol{n}_i^\top (R\boldsymbol{p}_i + \boldsymbol{t} - \boldsymbol{q}_i)\right)^2$. Linearize $R \approx I + \mathrm{skew}(\boldsymbol{\theta})$ and solve normal equations for $(\boldsymbol{\theta}, \boldsymbol{t})$.

Algorithm (Point-to-Point)

Alternate correspondences and pose until convergence

1) Initialize $R_0, \boldsymbol{t}_0$ (e.g., from rough pose or RANSAC) 2) Correspondences: for each $\boldsymbol{p}_i$ find nearest $\boldsymbol{q}_{c(i)}$ (kd-tree) 3) Reject outliers: max distance, normal angle, robust weights 4) Pose update: solve closed-form registration for $(R_{k+1}, \boldsymbol{t}_{k+1})$ 5) Stop when $\|\boldsymbol{t}_{k+1} - \boldsymbol{t}_k\|$ and rotation change are small or max iters reached

Point-to-Plane ICP

Faster convergence near solution; minimize $\sum \left(\boldsymbol{n}_i^\top (R\boldsymbol{p}_i + \boldsymbol{t} - \boldsymbol{q}_i)\right)^2$ via linear least-squares each iteration.

Practical Tips

- Multi-scale: voxel downsample then refine at higher resolutions
- Re-estimate normals after filtering; orient via view vector
- Use max correspondence distance schedule (coarse -> fine)

Outlier Handling

Robustness to partial views and clutter

- Max correspondence distance; normal compatibility threshold
- Robust costs (Huber/Tukey) or trimming (keep best rho% pairs)
- RANSAC on minimal sets (3 non-collinear pairs) to seed pose

Segmentation

Preprocess clouds for registration

- Remove dominant plane (table) via RANSAC plane fit
- Euclidean clustering in 3D, optionally color cues
- Mask known static geometry from the scene model

Soft Correspondence Matrix

Soft assignments and EM-style registration

- $C :\in \mathbb{R}^{N \times M}$, rows sum to 1; C_{ij} = belief $\boldsymbol{p}_i$ matches $\boldsymbol{q}_j$
- Alternate: (E) update C from distances; (M) update $(R, \boldsymbol{t})$ by weighted registration
- Use Gaussian kernels with bandwidth annealing; add uniform outlier row

Global Optimization & Non-Penetration

Pose from depth with physical constraints **Formulation:**

$$\min_{X :\in \mathrm{SE}(3)} \sum_i w_i \left\|X\boldsymbol{p}_i - \boldsymbol{q}_{c(i)}\right\|^2 + \lambda \sum_{\text{mesh facets}} \varphi_+\left(d_{\text{signed}}(X, \text{scene})\right)$$

where φ_+ penalizes penetration only. Use signed distance fields or mesh distance queries; optionally visibility constraints.

Practice:

- Initialize from ICP/RANSAC; refine with smooth SDF penalties
- Jointly estimate multiple objects with mutual non-penetration
- Leverage solver trust-regions; enforce SO(3) via retraction

Surface Normal Estimation

Estimate surface orientation for point-to-plane and filtering

- Cross-product on local grid: finite differences on back-projected rays
- PCA on k-NN: smallest-eigenvector of covariance gives normal
- Orient consistently: flip to face sensor, then smooth

Ch. 05: Grasping & Task Planning

Contact models and quasistatic force balance

Contact Models

Point Contact w/o Friction (PC): $f_n \geq 0$, $\boldsymbol{f}_t = \boldsymbol{0}$. **Point Contact w/ Friction (PCWF)**: Coulomb cone $\|\boldsymbol{f}_t\| \leq \mu f_n$. **Soft-Finger (SF):** Adds torsional moment $|\tau_n| \leq \mu_r f_n$ via limit surface.

Quasistatic Equilibrium

Let m contacts, stacked contact force vector $\boldsymbol{f}_c :\in \mathbb{R}^{km}$ (with $k \in \{1, 2, 3\}$ per model). The grasp matrix $G :\in \mathbb{R}^{6 \times km}$ maps to object wrench: $\boldsymbol{w} := G\boldsymbol{f}_c = [\boldsymbol{f}; \boldsymbol{\tau}]$ Force balance with external wrench $\boldsymbol{w}_{\text{ext}}$: $G\boldsymbol{f}_c + \boldsymbol{w}_{\text{ext}} = \boldsymbol{0}$

Feasible if each contact satisfies its friction/torque limits, e.g. PCWF: $f_n \geq 0$, $\|\boldsymbol{f}_t\| \leq \mu f_n$ (often linearized by a pyramid).

Coverage in wrench space and quantitative quality

CONE Contact/Grasp Wrench Cones

Each contact i generates a convex cone of wrenches $\mathcal{C}_i := \{G_i \boldsymbol{f}_i : (\boldsymbol{f}_i, \text{constraints})\}$. The Grasp Wrench Set (GWS) is the Minkowski sum $\mathcal{W} := \mathcal{C}_1 + \ldots + \mathcal{C}_m$ (convex for linearized cones).

Force Closure vs Form Closure

Force closure: $\mathcal{W}$ contains a neighborhood of the origin $\boldsymbol{0} \Rightarrow$ can resist any small wrench. Equivalent: the convex cone cone$(\mathcal{W})$ spans $\mathbb{R}^6$. **Form closure:** Purely geometric immobilization (typically more contacts, e.g. frictionless ≥ 7 in 3D). In practice we seek force closure.

Quality Metrics (epsilon-metric)

With unit-bounded contact efforts, define the largest ball around $\boldsymbol{0}$ contained in $\mathcal{W}$: $q_\varepsilon := \max_{\varepsilon \geq 0} \min_{\|\boldsymbol{w}\|=1} \boldsymbol{w}^\top \left(\sum_i G_i \boldsymbol{f}_i\right)$.

Intuition: worst-case unit wrench margin resisted by the grasp. Larger is better; sensitive to contact locations, normals, and friction.

Pairs of contacts aligned with friction cones

Condition

Two surface points with outward normals $\boldsymbol{n}_1, \boldsymbol{n}_2$ and line-of-centers direction $\boldsymbol{d}$ are antipodal if $\boldsymbol{d}$ lies within both friction cones:

$$\angle(-\boldsymbol{n}_1, \boldsymbol{d}) \leq \arctan(\mu) \quad \text{and} \quad \angle(\boldsymbol{n}_2, \boldsymbol{d}) \leq \arctan(\mu)$$

For frictionless: require $\boldsymbol{d}$ colinear with $-\boldsymbol{n}_1$ and $\boldsymbol{n}_2$.

Use

Simple, effective heuristic for parallel-jaw grasps; robust under moderate pose error when cones are wide (larger μ).

From geometry to feasible, high-quality grasps

Geometry Cues

From mesh/point cloud: estimate normals and curvature; sample handles/edges; sample antipodal pairs on locally parallel patches; for suction, prefer smooth, planar regions with sufficient area and reachable normal.

Feasibility Filters

- Collision-free closing region; gripper width limits; approach clearance.
- Reachability/IK with margin; avoid joint limits/singularities; plan approach and retreat.

Ranking

Score via q_ε, normal alignment, distance to edges, friction margin $\arctan(\mu) - \angle$; penalize occlusion/uncertainty. Choose top-K diverse grasps, then refine with local optimization.

Composing grasping with perception, motion, and recovery

Finite-State Machines (FSM)

Discrete modes (Detect → Plan → Grasp → Place) with guarded transitions. Simple and explicit, but can become brittle and hard to scale.

Behavior Trees (BT)

Nodes tick top-down. Core controls: Sequence (fails fast on first child failure), Fallback/Selector (succeeds on first child success), Parallel, Decorators (timeouts, retries), and leaf Actions/Conditions. Advantages: modularity, reactivity, reuse. Blackboard shares state.

Robustness Patterns

Timeouts with retry/fallback; perception refresh on failures; re-plan on IK/collision failure; grasp re-ranking; escalate to place-in-bin if precise place fails.

Polyhedral cones enable LP-based checks

Polyhedral Approximation

For each contact i, approximate the circular cone by r generators $D_i := [\boldsymbol{d}_{i1} \quad \boldsymbol{d}_{ir}]$ (unit directions in tangential space and normal). Contact force $\boldsymbol{f}_i := D_i \alpha_i$ with $\alpha_i \geq \boldsymbol{0}$ yields linear constraints. Stack $W := [G_1 D_1 \quad G_m D_m]$ and $\alpha := [\alpha_1; \quad ; \alpha_m]$.

Equilibrium as LP

Feasibility under external wrench $\boldsymbol{w}_{\text{ext}}$:

$$\text{find } \alpha \geq \boldsymbol{0} \quad \text{s.t.} \quad W\alpha + \boldsymbol{w}_{\text{ext}} = \boldsymbol{0}$$

If feasible, the grasp can statically resist $\boldsymbol{w}_{\text{ext}}$. To add joint torque limits $|\boldsymbol{\tau}_j| \leq \tau_{\max,j}$ use $\boldsymbol{\tau} = J^\top F$ where F stacks the contact forces, giving extra linear inequalities.

Worst-case wrench margin from a unit-effort GWS

Ferrari–Canny epsilon-Metric (Computation)

Unit GWS

Using polyhedral cones with $\sum \alpha \leq 1$ defines a compact $\mathcal{W}_1 := \{W\alpha : \alpha \geq \boldsymbol{0} \mid \sum \alpha \leq 1\}$.

epsilon via Directional LP

Sample unit directions $\boldsymbol{u}_k$ on S^5 and solve

$$\text{minimize } \boldsymbol{u}_k^\top \boldsymbol{w} \quad \text{s.t. } \boldsymbol{w} \in \mathcal{W}_1$$

Then $q_\varepsilon = \min_k(-\text{optval}_k)$. More exactly, with an H-representation $A\boldsymbol{w} \leq \boldsymbol{b}$ of $\mathcal{W}_1$, $q_\varepsilon = \min_i \left(\frac{b_i}{\|\boldsymbol{a}_i\|}\right)$ where $\boldsymbol{a}_i^\top \boldsymbol{w} \leq b_i$ are facets.

Mapping contact forces to joint torques and constraints

Wrench–Torque Relations

Stack contact forces F; object wrench $\boldsymbol{w} = GF$; hand joint torques $\boldsymbol{\tau} = J^\top F$. Add bounds $|\boldsymbol{\tau}| \leq \boldsymbol{\tau}_{\max}$, fingertip force bounds, and closure region collisions to the LP/QP for realistic feasibility.

Normal capacity from vacuum; shear from friction/seal

Capacities

Normal: $F_n^{\max} := (\Delta P) A$ from pressure differential ΔP and cup area A. Shear: $F_t^{\max} := \mu_s F_n^{\max}$ (surface friction). Peel/tilt moment limit roughly $M_n^{\max} \sim k A \Delta P r$ (cup stiffness k, effective radius r). Require approach aligned with surface normal and sealable patch.

Efficient sampling and testing on depth data

Procedure

1) Estimate normals via PCA in local neighborhoods; smooth outliers. 2) Sample candidates on near-parallel patches; enforce gripper width and thickness. 3) For pair (p1, p2) with direction $\boldsymbol{d}$, test $\angle(-\boldsymbol{n}_1, \boldsymbol{d}) \leq \arctan(\mu)$ and $\angle(\boldsymbol{n}_2, \boldsymbol{d}) \leq \arctan(\mu)$. 4) Check collision of closing region and approach clearance. 5) Score with friction margin and distance-to-edges; deduplicate by pose.

Tick outcomes and control flow

Execution & Robustness

- **Node statuses:** success/failure/running; Sequence returns first non-success; Selector returns first success; Decorators modify child status; Blackboard shares state (pose, IK, grasp-index).
- **Robust pattern:** Detect → Sample/Rank grasps → Try K: PlanIK → Execute; on failure: try next; if all fail: refresh perception/retry; fallback to place-in-bin.

Robust model fitting via random sampling

Model and Inliers

Plane: $\boldsymbol{n}^\top \boldsymbol{x} + d = 0$ with $\|\boldsymbol{n}\| = 1$. Inlier test for point $\boldsymbol{x}_i$:

$$\left|\boldsymbol{n}^\top \boldsymbol{x}_i + d\right| \leq \tau$$

Minimal sample size: $s := 3$ (non-collinear points define a plane).

Iteration Budget

Given desired success prob $p \in [0, 1]$ and inlier ratio $w \in [0, 1]$:

$$N_{\text{iters}} := \left\lceil \frac{\ln(1-p)}{\ln(1-w^s)} \right\rceil$$

Refit with all inliers, select model with largest consensus set.

Handling mismatches and robustness

Correspondences and Objective

Standard ICP minimizes

$$\sum_i \left\|O\boldsymbol{p}_{j(i)}^m - W\boldsymbol{p}_i^s\right\|^2$$

where $j(i)$ is nearest-neighbor. With outliers, use trimming (keep a fraction of smallest residuals) or robust losses (Huber/Tukey) via weighted least squares.

Practical Notes

- Initialize with reasonable pose; re-estimate correspondences after each update.
- Reject pairs beyond a max distance; enforce normal consistency when available.

For a block of mass m on slope angle theta

Forces

$$f_t = mg\sin(\theta), \quad f_n = mg\cos(\theta)$$

No-slip condition: $\mu \geq \tan(\theta)$.

PCA in local neighborhoods

Covariance and Normal

For neighborhood points $\{\boldsymbol{p}_i\}$, mean $\boldsymbol{p}_| := \left(\frac{1}{k}\right) \sum_i \boldsymbol{p}_i$ and covariance

$$W := \sum_i \left(\boldsymbol{p}_i - \boldsymbol{p}_|\right)\left(\boldsymbol{p}_i - \boldsymbol{p}_|\right)^\top$$

The eigenvector of W with the smallest eigenvalue is the surface normal. Orient consistently (e.g., toward the sensor) if needed.

Tangency conditions and Hessian interpretation

Conditions

Let boundary be $\boldsymbol{p}(t)$ with tangent $\boldsymbol{tan}(t) := \boldsymbol{p}'(t)$. A frictionless antipodal pair (t_1, t_2) satisfies

$$\boldsymbol{tan}(t_1)^\top(\boldsymbol{p}(t_2) - \boldsymbol{p}(t_1)) = 0, \quad \boldsymbol{tan}(t_2)^\top(\boldsymbol{p}(t_1) - \boldsymbol{p}(t_2)) = 0$$

These imply the gradient of a suitable alignment energy in (t_1, t_2) is zero (the pset's $\partial \frac{E}{\partial} t = [0, 0]$ observation).

Hessian and Preference

- Local minima → concave points, local maxima → convex points, saddles → mixed curvature.
- Preferred grasps are at convex (local maxima) regions to avoid penetration.

Ch. 06: Motion Planning

Rich costs and constraints over joint configurations

Problem View

Given $f_{\text{kin}} : q \mapsto X_G$, solve for q with costs/constraints

Formulations

Nominal IK: $\min_q \|q - q_0\|^2$ s.t. ${}^G X^O = f_{\text{kin}(q)}$ (pose match), $q \in [q_{\min}, q_{\max}]$ (joint limits), $d_{\min}(q) \geq 0$ (non-penetration).

Differential IK: one SQP/least-squares step in Δq around q_t; good locally, cannot switch homotopy classes.

Constraint Templates

Position: $p_G(q) = p^\star$ Orientation: $R_G(q) = R^\star$ Distance: $d_{\min}(q) \geq 0$ Joint: $q \in [q_{\min}, q_{\max}]$

Differential IK (Least-Squares)

$\min_{\Delta q} \|J(q_t)\Delta q - v^\star\|^2 + \lambda \|\Delta q\|^2$ s.t. $A\Delta q \leq b$ (vel/avoidance bounds). $\Rightarrow \Delta q = (J^\top J + \lambda I)^{-1} J^\top v^\star$ (damped pseudo-inverse).

Guidance

- Keep objectives simple (joint-centering); encode hard requirements as constraints.
- Write minimal constraints (e.g., partial orientation, point-on-line contacts).
- Collision constraints are nonconvex; solvers like SNOPT often work at interactive rates but offer no guarantees.

Optimize a continuous joint-space trajectory with time-scaling

Trajectory Optimization

$\min_{\alpha, T} T$ s.t. ${}^{G_{\text{start}}}X^O = f_{\text{kin}(q_\alpha(0))}$, ${}^{G_{\text{goal}}}X^O = f_{\text{kin}(q_\alpha(T))}$, $|\dot{q_\alpha}(t)| \leq v_{\max}$ for all $t := [0, T]$.

B-spline Path + Time Rescaling

- Path $r(s)$, $s \in [0, 1]$ as B-spline; trajectory: $q(t) := r(\frac{t}{T})$.
- Bases: $r(s) = \sum_i N_{i,k}(s) c_i$ with control points c_i.
- Derivatives: $\frac{d}{ds} r(s) = \sum_i N_{i,k-1}(s) D_i$ where $D_i := \frac{k}{u_{i+k} - u_i}(c_i - c_{i-1})$.
- Convex hull: $r(s)$ lies in convex hull of active $c_i \Rightarrow$ box constraints on c_i imply forall s bounds on $r(s)$.

Velocity Constraints via Time-Scaling

$s := \frac{t}{T} \Rightarrow \dot{q}(t) = \frac{dr}{ds}(s)\left(\frac{1}{T}\right)$. Enforce $|\dot{q}(t)| \leq v_{\max}$ for all t by linear bounds on D_i: $|D_i| \leq v_{\max} T$ for all i. Higher derivatives scale as T^{-2}, T^{-3}, ... and become nonlinear in (c_i, T); they can be constrained at samples or relaxed.

Global exploration with probabilistic completeness

Sampling-based Planning

RRT (basic): sample q_{rand}, find $q_{\text{near}} = \arg\min_q d(q, q_{\text{rand}})$, step toward by η, keep if collision-free. Good in high-dim; jerky and suboptimal; suffers in bug traps. Key knobs: step size η, goal bias p_{goal}, metric $d(.,.)$, local planner. RRT*: rewiring for asymptotic optimality using radius $r_n \sim \left(\log \frac{n}{n}\right)^{\frac{1}{d}}$. RRT-Connect: bidirectional; grow trees from start and goal; connect repeatedly extends toward the other tree by step η until blocked; very fast first solution; not asymptotically optimal.

AO-RRT-Connect (AORRTC): bidirectional RRT* with rewiring in balls of radius $r_n \sim \left(\log \frac{n}{n}\right)^{\frac{1}{d}}$; attempts connections while rewiring both trees; asymptotically optimal; often keeps RRT-Connect's fast first path.

PRM

Offline roadmap: sample collision-free milestones; connect k-NN or within radius r when straight segment is collision-free; online query via shortest path. Needs post-smoothing for curvature/dynamics.

Post-processing

Shortcutting and anytime B-spline smoothing; tune distance metrics and collision checking for speed.

Bridge global graph search with continuous convex optimization

GCS over Convex Sets

Replace PRM points/edges with convex regions and continuous decisions at vertices/edges. Solve shortest path over a graph of convex sets via a strong convex relaxation (often tight with rounding).

Transcription for Kinematic Planning

- Assume convex decomposition of collision-free C-space (justified below).
- At each visited region, choose two points so line/curve lies within region; enforce equality at overlaps to stitch segments.
- Use Bézier curves per region + time-scaling; impose convex constraints for continuity and velocity limits that hold forall t.

Variables and Constraints

Per vertex (region) V : pick $(q_{\text{in}}, q_{\text{out}}) \in \{V\} \times \{V\}$. Per edge (U, V) : enforce $q^U_{\text{out}} = q^V_{\text{in}}$ when traversed; add convex arc-length/time costs and derivative bounds via Bézier control points.

Objectives and Constraints

- Costs: duration T, path length upper bound, energy int $\|\cdot q(t)\|_2^2 dt$.
- Constraints: derivative continuity, strict velocity bounds forall t, initial/final states, additional convex bounds.

Inflate samples into certified convex C-space regions

IRIS & Region Construction

- IRIS (Euclidean or C-space): alternate separating hyperplanes and MVEE to inflate regions.
- IRIS-NP/IRIS-NP2: nonlinear or improved pipelines; fast, probabilistic.
- IRIS-ZO: zeroth-order, trivially parallel.
- SOS/Algebraic kinematics: rigorous certificates via polynomials; slower but sound (uses stereographic projection coordinates).

Construction Tips

Use minimum clique cover over visibility graph for efficient covering; seed with IK solutions, teleop demos, or other planner rollouts to cover "important" volumes in high DOF.

Iteration Sketch (IRIS)

1) Fix obstacle set; solve for separating hyperplanes that keep a convex polytope P collision-free. 2) Fix P; solve MVEE to maximize enclosed ellipsoid volume. 3) Inflate/clip and repeat until convergence.

Trade-offs

- IK/Diff-IK: fast; local vs global; use constraints not penalties.
- Kinematic TrajOpt: rich constraints, online-capable; local minima; sample collisions sparsely + dense verification.
- RRT/PRM: global completeness notion; limited curvature/dynamics; requires smoothing.
- GCS: global structure + continuous constraints; strong relaxations; needs convex decomposition (IRIS family).

Ch. 07: Mobile Manipulation

Extends table-top manipulation with mobility; raises new perception, planning, and simulation challenges.

Scope & Motivation

Ambition boost: Mobility enables in-home tasks across rooms and scenes. Many tools carry over; critical differences arise in perception, state estimation, and navigation.

Partial views, unknown environments, and robot state estimation become central.

Perception, Mapping & State Estimation

One-sided observations: Head-mounted sensors often see only one side of objects; antipodal grasp heuristics need completion. **Learning is fundamental:** Infer occluded geometry via data (shape completion, semantics). Move sensors to reduce uncertainty (active perception; planning under uncertainty).

Mathematical framing (shape completion & VOI)

$$O := \text{object shape};\ Z := \text{current view}$$

$$\hat{O} := \arg\max_O \texttt{func}[O \mid Z, \mathcal{D}] = \arg\max_O \texttt{func}[Z \mid O]\ \texttt{func}[O \mid \mathcal{D}]$$

$$a^\star := \arg\max_a \mathbb{E}[U(\text{bel}') - U(\text{bel}) \mid a]$$

$$\text{bel}' := \text{posterior after active view}\ \ a$$

Unknown/Dynamic Environments

Representation needs: Fast collision queries, distance fields, scalable updates from raw depth/RGB-D. **Voxel grids:** Discretize space; maintain occupied/free/probabilistic occupancy. Efficient sphere–voxel queries; easy parallelization. **OctoMap:** Octree-based multi-resolution mapping; probabilistic updates incl. free space. Good for large scenes.

Occupancy updates (log-odds)

$$L_{t(v)} := \ln\left(\frac{\texttt{func}[\text{occ}_v \mid z_{1:t}]}{1 - \texttt{func}[\text{occ}_v \mid z_{1:t}]}\right)$$

$$L_{t(v)} = L_{t-1}(v) + l(z_t \mid v) - l_0\ \ \texttt{func}[\text{occ}_v \mid z_{1:t}] = \frac{1}{1 + \exp\left(-L_{t(v)}\right)}$$

where $l(z_t \mid v)$ is the inverse sensor model contribution for voxel v along ray z_t and $l_0 := \ln\left(\frac{p_0}{1 - p_0}\right)$.

Sphere–voxel distance query

$$d(q) := \min_{v \in \mathcal{V}_{\text{occ}}} \text{dist}_2\left(c_{s(q)}, v\right) - r_s$$

$$d(q) > 0 \Rightarrow \text{collision-free; enables fast clearance costs via EDT}$$

Robot State Estimation (Localization)

Beyond fixed-base: Must estimate ${}^W X^C$ and base pose; wheel odometry alone drifts (slip). Fuse IMU, wheels, lidar/RGB-D/RGB. **Classics:** Recursive Bayes filters and smoothing (e.g., pose-graph SLAM / iSAM). **Trends:** Strong visual(-inertial) odometry; monocular depth; dense 3D reconstruction (e.g., NeRF, Gaussian splatting).

Bayes filter (discrete-time)

$$\text{bel}_t(x) := \eta\ \texttt{func}[z_t \mid x] \sum_{x'} \texttt{func}[x \mid u_t, x'] \text{bel}_{t-1}(x')$$

EKF (nonlinear Gaussian models)

$$x_{t+1} = f(x_t, u_t) + w_t, \quad z_t = h(x_t) + v_t,$$
$$w_t \sim \mathcal{N}(0, Q), \quad v_t \sim \mathcal{N}(0, R)$$

$$A_t := \frac{\partial f}{\partial x} |_{x_t}, \quad W_t := \frac{\partial f}{\partial w} |_{x_t}, \quad H_t := \frac{\partial h}{\partial x} |_{x_t}$$

$$P_{t+1|t} = A_t P_t A_t^\top + W_t Q W_t^\top$$

$$K_{t+1} = P_{t+1|t} H_{t+1}^\top \left(H_{t+1} P_{t+1|t} H_{t+1}^\top + R\right)^{-1}$$

$$x_{t+1} = x_{t+1|t} + K_{t+1}\left(z_{t+1} - h\left(x_{t+1|t}\right)\right)$$

$$P_{t+1} = (I - K_{t+1} H_{t+1}) P_{t+1|t}$$

Pose-graph SLAM objective

$$X := (x_0, ..., x_T), J(X) := \sum_i \|r_i(X)\|^2_{\Sigma_i^{-1}}, r_i := \text{between-factor error}$$

Same kinematic tools, with added base DOFs and occasionally nonholonomic constraints.

Base & Kinematics

iiwa + {x,y,z} prisms + yaw: Treat base as extra joints; IK, trajopt, RRT/PRM apply directly. Local trajopt scales well with dimension; sampling planners need care as DOFs grow. **Continuous joints:** Handle wrap-around in metrics/extend; in GCS/IRIS use local coordinates with <= π domain for wrapping joints to avoid long-way paths.

Mobile-base IK (holonomic base)

Variables: $q := (x_b, y_b, z_b, \text{yaw}_b, q_{\text{arm}})$

$$\min_q w_p \left\|p_{\text{ee}}(q) - p_{\text{goal}}\right\|_2^2 + w_R \left\|\log\left(R_{\text{goal}}^\top R_{\text{ee}}(q)\right)\right\|_2^2 + w_c \|q - q_0\|_2^2$$
$$s.t. q_{\min} \leq q \leq q_{\max}, d_{\min}(q) \geq d_{\text{safe}}$$

Orientation error uses rotation log-map. $d_{\min}$ is signed clearance (e.g., EDT over voxels).

Angle wrap metric

$$d_{\theta(a,b)} := \text{atan2}(\sin(a - b), \cos(a - b))$$

Wheeled Bases

Plan with no-slip; track with feedback.

- Holonomic drives: e.g., mecanum/omni; direct (x, y, yaw) commands feasible.
- Nonholonomic drives: differential drive, Dubins (forward-only turns), Reeds–Shepp (with reverse). Distance/extend must respect constraints.

Differential-drive kinematics and Pfaffian constraint

$$\dot{x} = v\cos(\theta), \dot{y} = v\sin(\theta), \dot{\theta} = \omega$$
$$A(q)\dot{q} = 0, A(q) := [-\sin(\theta), \cos(\theta), 0] \Rightarrow -\sin(\theta)\dot{x} + \cos(\theta)\dot{y} = 0$$

Dubins / Reeds–Shepp primitives

$$P := \text{sequences of } \begin{matrix} L \\ R \\ S \end{matrix} \text{ minimizing path length subject to turn-rate bounds}$$

$$\kappa_{\max} := \frac{v}{R_{\min}}, \text{curvature bound}$$

Distance/extend in planners must compose feasible primitives and respect $\kappa_{\max}$.

Legged Bases

As a user of platform APIs (e.g., Spot): Often exposed as holonomic base; low-level balance handled by platform. Watch COM/balance-induced deviations from commanded path; rough terrain introduces richer constraints.

Beyond engines; need environments and assets for mobile tasks.

Exercises (What to Practice)

Simulation Set-Up

Analyze collision geometry of SDFs; compose a manipulation scene.

Mobile-Base IK

Solve IK with and without fixed base; removing base position constraints simplifies optimization.

Ch. 08: Manipulator Control

Torque-level controllers that execute higher-level motion/force commands

Core Systems

PD (PidController): $u \leftarrow K_p(q_d - q) + K_d(\dot{q}_d - \dot{q}) + K_i \int(q_d - q)$. In manipulation, typically set $K_i := 0$ (avoid windup).

Joint Stiffness: $u \leftarrow -\tau_g(q) + K_p(q_d - q) + K_d(\dot{q}_d - \dot{q})$ (gravity comp + PD). Removes steady-state error under constant loads.

Inverse Dynamics Control: $u \leftarrow M(q)\ddot{q}_d + C(q, \dot{q})\dot{q} - \tau_g(q)$. Choose $\ddot{q}_d := \ddot{q}_{\text{ref}} + K_p(q_{\text{ref}} - q) + K_d(\dot{q}_{\text{ref}} - \dot{q})$ for tracking.

Spatial Force Control: Command desired Cartesian force at contact/end-effector; maps to joint torques.

Spatial Stiffness (Operational Space): Program end-effector to behave like mass-spring-damper in task frame; handle nullspace with joint objectives.

Start with 2D point-finger of mass m; gravity along $-z$; contact force f^{F_c}

PD vs Gravity Comp vs Inverse Dynamics

PD: $m\ddot{z} = -mg + k_p(z_d - z) + k_d(\dot{z}_d - \dot{z})$ => steady-state error under constant mg:

$$\tilde{z} := z_d - z = m\frac{g}{k_p}.$$

Gravity-comp Stiffness: $u \leftarrow -\tau_g + K_p(q_d - q) + K_d(\dot{q}_d - \dot{q}) \Rightarrow m\ddot{\tilde{z}} = k_p\tilde{z} + k_d\dot{\tilde{z}}$ (no steady-state error to gravity).

Inverse Dynamics (with accel feedforward):

$$u \leftarrow -\tau_g + m\big[\ddot q_d + K_p(q_d - q) + K_d(\dot q_d - \dot q)\big]$$
$$\Rightarrow \ddot{\tilde z} + k_d\dot{\tilde z} + k_p\tilde z = 0 \quad \text{mass-spring-damper on error.}$$

Direct Force Control

Quasi-static idea

With small accelerations in contact: $f^{F_c} = -mg - u$ => choose

$$u \leftarrow -mg - f^{F_c}_{\text{desired}}.$$

Off-contact, same command accelerates the finger toward contact (useful for autonomous approach without precise geometry).

Free-space vs Contact: Steady State

Controller (1D): $u := k_p(x_d - x) - f^{F_c}_{\text{desired}}$.

- Free space ($f^{F_c} = 0$), steady state ($\dot x = \ddot x = 0$):

$$0 = k_p(x_d - x) - f^{F_c}_{\text{desired}} \quad\Rightarrow\quad x - x_d = -\frac{f^{F_c}_{\text{desired}}}{k_p}.$$

Cannot have $x = x_d$ and nonzero $f^{F_c}_{\text{desired}}$ simultaneously.

- In contact: wall reaction f^{F_c} balances $f^{F_c}_{\text{desired}}$ ⇒ zero steady-state error achievable.

Ch. 08 (cont.): Impedance, Hybrid, and OSC

Indirect Force Control (Stiffness/Impedance)

SPRING Target behavior in Cartesian coordinates

For end-effector F with position $p^F := [x, z]^\top$ and velocity v^F:

$$m\dot v^F + K_d v^F + K_p(p^F - p^{F_d}) = f^F.$$

Implement via gravity-comp PD at joints or operational-space control. Passivity (with K_d "positive definite") grants robust stability under unknown environments.

Hybrid Position/Force Control

World frame form

$$u \leftarrow -\tau_g + \big[k_p(x_d - x) + k_d(\dot x_d - \dot x); -f^F_{\text{desired},W_z}\big].$$

Contact-frame form

With rotation ${}^W R^C$:

$$u \leftarrow -\tau_g + {}^W R^C\Big[k_p\big(p^{F_d}_{C_x} - p^F_{C_x}\big) + k_d\big(v^{F_d}_{C_x} - v^F_{C_x}\big); -f^F_{\text{desired},C_z}\Big].$$

From joint-space dynamics to operational space and nullspace control

Dynamics

$$M(q)\ddot q + C(q,\dot q)\dot q = \tau_{g(q)} + u + \sum_i J_i^\top(q) f^{c_i}.$$

Inverse Dynamics / Computed Torque

$$u \leftarrow M\ddot q_d + C\dot q - \tau_g, \quad \ddot q_d := \ddot q_{\text{ref}} + K_p(q_{\text{ref}} - q) + K_d(\dot q_{\text{ref}} - \dot q).$$

Joint Stiffness (iiwa interface)

$$u \leftarrow -\tau_{g(q)} + K_p(q_d - q) + K_d(\dot q_d - \dot q) + \tau_{\text{ff}}.$$

Diagonal K_p, K_d; use τ_{ff} for Cartesian forces

Operational Space (Cartesian Stiffness)

Let E be end-effector, $p^E = f_{\text{kin}}(q)$, $v^E = J(q)\dot q$. Using $u := J^\top\, {}^B f^E_u$ and assuming only contact at E:

$$M_{E(q)}\ddot p^E + C_{E(q,\dot q)}\dot q = {}^B f^E_g(q) + {}^B f^E_u + {}^B f^E_{\text{ext}},$$
$$M_E := \left(JM^{-1}J^\top\right)^{-1}, \quad C_E := M_E\big(JM^{-1}C - \dot J\big), \quad {}^B f^E_g := M_E J M^{-1}\tau_g.$$

Choose

$${}^B f^E_u \leftarrow -{}^B f^E_g + K_p(p^{E_d} - p^E) + K_d(\dot p^{E_d} - \dot p^E) u \leftarrow J^\top\, {}^B f^E_u$$

to realize spring-damper behavior at the end-effector.

Nullspace Joint Objectives

With dynamically consistent nullspace projector P:

$$u \leftarrow J^\top\, {}^B f^E_u + P\big[K_{p\,\text{joint}}(q_0 - q) - K_{d\,\text{joint}}\dot q\big].$$

Ch. 08 (cont.): Tuning, Flip-Up, RCC

Tuning and Error Dynamics

Point-mass intuition

Error ODE: $\ddot{\tilde z} + k_d\dot{\tilde z} + k_p\tilde z = 0$. Match to $\ddot e + 2\zeta\omega_n\dot e + \omega_n^2 e = 0$:

$$k_d := 2\zeta\omega_n, \quad k_p := \omega_n^2.$$

Joint-space effective inertia

Around configuration q, along joint i with inertia M_{ii}, the closed-loop approx is $M_{ii}\ddot{\tilde q}_i + K_{dii}\dot{\tilde q}_i + K_{pii}\tilde q_i = 0$ so choose

$$K_{dii} := 2\zeta\omega_n M_{ii}, \quad K_{pii} := \omega_n^2 M_{ii}.$$

Task-space (operational) tuning

Using $M_E := \left(JM^{-1}J^\top\right)^{-1}$, set

$$K_d := 2\zeta\omega_n M_E, \quad K_p := \omega_n^2 M_E,$$

to achieve approximately isotropic second-order error dynamics at the end-effector.

Force-Based Flip-Up (Constraints)

CONE Friction cones

Finger-on-box at contact frame C and ground at corner frame A:

$$f^{B_C}_{\text{finger},C_z} \ge 0, \quad \left|f^{B_C}_{\text{finger},C_x}\right| \le \hat\mu_C f^{B_C}_{\text{finger},C_z},$$
$$f^{B_A}_{\text{ground},A_z} \ge 0, \quad \left|f^{B_A}_{\text{ground},A_x}\right| \le \hat\mu_A f^{B_A}_{\text{ground},A_z}.$$

Torque about pivot

About A: $\tau^{B_A}_{\text{total},W_y} := \tau^{B_A}_{\text{gravity},W_y} + \tau^{B_A}_{\text{ground},W_y} + \tau^{B_A}_{\text{finger},W_y}$. With $\tau_{\text{ground}} = 0$ about its own point,

$$+\tau^{B_A}_{\text{total},W_y} = \tau^{B_A}_{\text{gravity},W_y} + \tau^{B_A}_{\text{finger},W_y} > 0 \quad \text{to flip up.}$$

Constrained least-squares control

Given estimate $\hat\theta$ and quasi-static balance at B:

$$\min_{f^{B_C}_{\text{finger},C}, f^{B_A}_{\text{ground},A}} \left|\tau^{B_A}_{\text{finger},W_y} - \mathrm{PID}\big(\theta_d, \hat\theta\big)\right|^2,$$

subject to friction-cone inequalities above and force balance

$$f^{B_A}_{\text{ground},A} + \hat f^{B_A}_{\text{gravity},A} + f^{B_C}_{\text{finger},A} = 0.$$

√x Useful bounds (small-angle, square box)

About A with C at distance L and com at $\frac{L}{2}$:

$$f^{B_C}_{\text{finger},C_x} L \ge mg\left(\frac{L}{2}\right) \quad\Rightarrow\quad f^{B_C}_{\text{finger},C_x} \ge \frac{mg}{2}.$$

Finger friction: $\left|f^{B_C}_{\text{finger},C_x}\right| \le \mu_C f^{B_C}_{\text{finger},C_z}$ ⇒ lower bound on normal:

$$f^{B_C}_{\text{finger},C_z} \ge \frac{mg}{2\mu_C}.$$

Ground no-slip at A: $\left|f^{B_A}_{\text{ground},A_x}\right| \le \mu_A f^{B_A}_{\text{ground},A_z}$ with $f^{B_A}_{\text{ground},A_x} = -f^{B_C}_{\text{finger},C_x}$ and $f^{B_A}_{\text{ground},A_z} = mg - f^{B_C}_{\text{finger},C_z}$ gives an upper feasibility condition coupling μ_A and $f^{B_C}_{\text{finger},C_z}$.

Hybrid Control: Feasibility

- Require commanded normal force within friction limits at sticking contacts and below slip threshold at sliding contacts.
- For planar push with normal along z: need $\left|f_{\text{tangent}}\right| \le \mu f_{\text{normal}}$ at sticking interface; for sliding interface, target $\left|f_{\text{tangent}}\right| \approx \mu f_{\text{normal}}$ with direction opposing slip.
- Book-drag condition (gripper/table): favor $\mu_{\text{gripper}} > \mu_{\text{table}}$ to stick at gripper while sliding on table (ratio threshold near 1).

Reflected Inertia and Gearing

Reflected quantities (Gearboxes)

$$J_{\text{reflected}} := N^2 J_{\text{load}}, \quad b_{\text{reflected}} := N^2 b_{\text{load}}, \quad \tau_{\text{motor}} = \left(\frac{1}{N}\right)\tau_{\text{load}}.$$

Large N reduces sensitivity to load changes at the motor; tune gains against effective inertia seen at actuator.

Control Summary (for psets)

- $\tau = J^\top f$ maps Cartesian force to joint torques; OSC shapes end-effector dynamics.
- Stiffness control needs gravity feedforward; smaller K_p increases compliance for uncertainty.
- Direct force control enables pivoting/insertions but must honor friction cones.
- Velocity plant (integrator): P for setpoint; PI for constant-velocity tracking; add feedforward of desired velocity.

Implementation Recipes

InverseDynamicsController

1) Compute $\ddot q_d := \ddot q_{\text{ref}} + K_p(q_{\text{ref}} - q) + K_d(\dot q_{\text{ref}} - \dot q)$.
2) Command $u \leftarrow M\ddot q_d + C\dot q - \tau_g$.

Cartesian Stiffness

1) Pick frame E at the intended contact.
2) Compute p^E, v^E, J.
3) Choose K_p, K_d (optionally using M_E shaping).
4) Compute ${}^B f^E_u \leftarrow -{}^B f^E_g + K_p(p^{E_d} - p^E) + K_d(\dot p^{E_d} - \dot p^E)$, then $u \leftarrow J^\top\, {}^B f^E_u$.

Practical iiwa Notes

- Cannot send desired joint velocities; firmware differentiates positions (adds small delay) to preserve passivity guarantees.
- Cartesian impedance mode exists but switching modes/frames requires stopping; commonly stay in joint impedance and use τ_{ff} for Cartesian force cues.
- Gravity comp uses configured tool inertia; updates not applied online when grasp changes.

Peg-in-Hole & Compliance

- Avoid jamming with appropriate stiffness about the remote contact center.
- Remote-Centered Compliance (RCC) devices realize this mechanically (infinite bandwidth, no sensing).

Drake Program Outline & Core APIs

End-to-end skeleton

- Setup/Diagram/Sim: `StartMeshcat()` → `LoadScenario/MakeHardwareStation` → `DiagramBuilder` add/connect → `Build` → `Simulator.AdvanceTo(T)`.

Core APIs

- Types/frames: `RigidTransform`, `RotationMatrix`, `EvalBodyPoseInWorld`, `GetFrameByName`.
- Traj: `PiecewisePose.MakeLinear` → `.MakeDerivative()` for `V_G`; scalars via `PiecewisePolynomial.*Hold`; feed with `TrajectorySource`; integrate with `Integrator(7)`.
- Controllers: Diff-IK `v = pinv(J) V_G` (from `CalcJacobianSpatialVelocity`); also PD/Impedance, InverseDynamics.
- Station ports: in `iiwa.position`, `wsg.position`; out `iiwa.position_measured`, camera/point-cloud ports.
- Gotchas: choose `kV` vs `kQDot` consistently; initialize integrator with current `q`; time WSG open/close around grasp.

PseudoInverseController (minimal pattern)

- **Ports:** `V_WG: RR^6`; `iiwa.position: RR^7` → `iiwa.velocity: RR^7`.
- **Compute:** set plant `q`; `J <- CalcJacobianSpatialVelocity(..., kQDot, G, 0, W, W)[:,0:7]`; `v <- pinv(J) V_G` (damped if needed); output `v`.
- **Note:** If state uses generalized velocity `v`, use `kV` and map `qdot <-> v`.
- **Slice dofs:** Use the IIWA block only (e.g., `[:, 0:7]`), or slice via `iiwa_joint_1.velocity_start()` to `iiwa_joint_7.velocity_start()+1`.

Traj recipe

- **Poses:** `pose_traj <- PiecewisePose.MakeLinear(t, X_WG)` → `traj_V_G <- pose_traj.MakeDerivative()`.
- **WSG:** `traj_wsg <- PiecewisePolynomial.FirstOrderHold(t, fingers)`.
- **Sources:** `V_G_source`, `wsg_source` from the above.
- **Shapes/time:** `fingers` is `1xL`; times `t` must be strictly increasing.

Builder wiring

- Add `station`; get `plant`. Add `controller`, `integrator`.
- Connect: `V_G_source -> controller`, `controller -> integrator -> station.iiwa.position`.
- Feedback: `station.iiwa.position_measured -> controller.iiwa.position`; `wsg_source -> station.wsg.position`.
- Init: `integrator.set_integral_value(current_q)`
- **Get current_q:** `ctx <- station.CreateDefaultContext()`; `plant_ctx <- plant.GetMyContextFromRoot(ctx)`; `current_q <- plant.GetPositions(plant_ctx, plant.GetModelInstanceByName("iiwa"))`.

HardwareStation quick notes

- **Scenario:** YAML adds models/welds/drivers; `LoadScenario` → `MakeHardwareStation(meshcat)`.
- **Poses:** objects via `EvalBodyPoseInWorld`; gripper goals `X_WG = X_WO @ X_OG`.
- **Drivers/weld:** `IiwaDriver(position_only)` with `hand_model_name=wsg`; `SchunkWsgDriver{}`; weld `wsg::body` to `iiwa_link_7` with `Rpy[90,0,90]`.

MathProgram / Kinematic TrajOpt

- Build: `trajopt <- KinematicTrajectoryOptimization(num_q, N)`; `prog <- trajopt.get_mutable_prog()`.
- Path constraints: start/goal `PositionConstraint` at `s=0,1`; add joint/vel bounds; zero end velocities.
- Time: `AddDurationConstraint(Tmin, Tmax)`; costs `AddDurationCost`, `AddPathLengthCost`.
- Collisions: `MinimumDistanceLowerBoundConstraint` added as path constraints at sampled `s`.
- Init: warm-start (`trajopt.SetInitialGuess(karate_chop_traj)` or BSpline from RRT); `result <- Solve(prog)`; `traj <- ReconstructTrajectory(result)`; visualize with `PublishPositionTrajectory`.

Frames & Jacobian

- `CalcJacobianSpatialVelocity(context, wrt, B, p_BoBp_B, A, E)` → 6xN mapping for `V_ABp` expressed in `E`.
- For world-expressed gripper twist: `A=W`, `E=W`, `B=G`, `p=[0,0,0]`; slice to IIWA dofs.
- Damped LS near singularities: $v = \left(J^{\top} J + \lambda^2 I\right)^{-1} J^{\top} V$.