Deck 06 · Motion planning: optimization and sampling

4.3 From a goal configuration to a motion

4.4 Find routes, refine them, and check them

Özgür S. Öğüz · Bilkent University · Fall 2026

Known: forward kinematics, local optimization, constrained inverse kinematics.

New task: connect the current posture to an allowed goal.

Waypoint + reference motion + controller

Goal: reach the desired tip position at a chosen future time.

1 · Waypoint
IK chooses an allowed goal posture for the target.

→

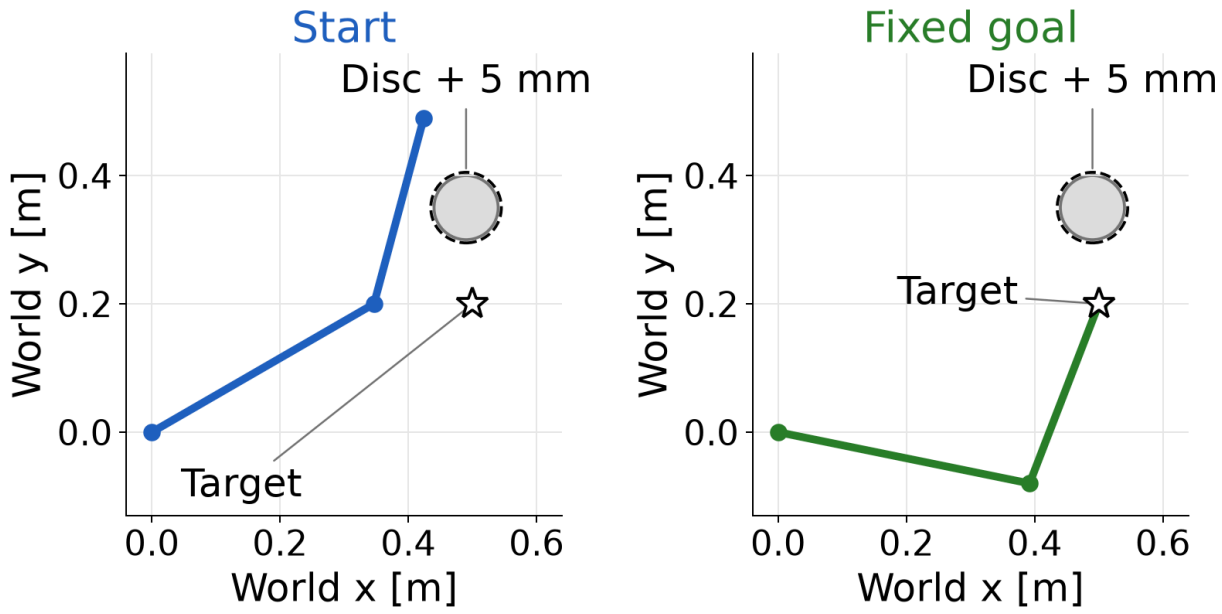
2 · Reference motion
Plan and check a route; choose when to traverse it.

→

3 · Controller
Use the reference and measurements to send robot commands.

Three horizons: future goal → motion along the route → short-term corrections.

An allowed goal still needs a connecting motion



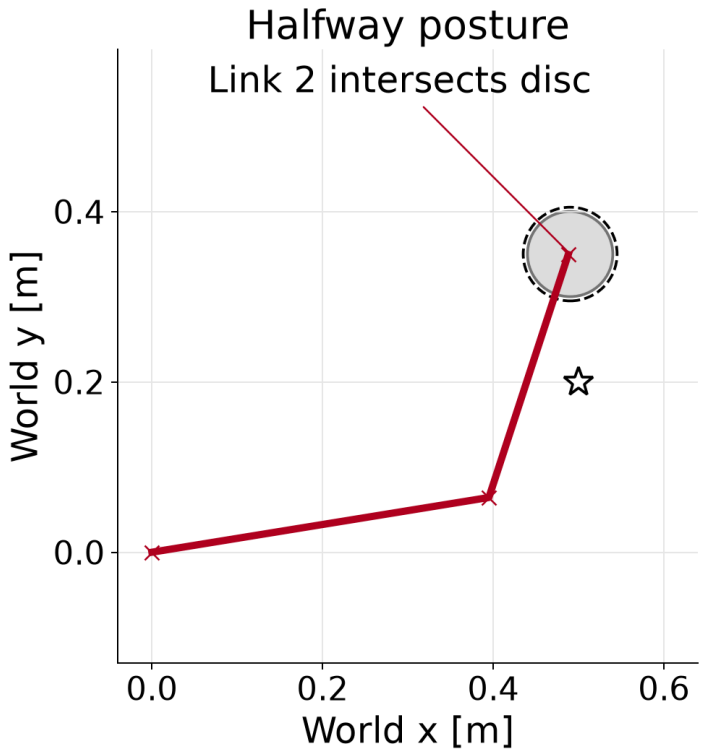
Model: 0.4 m and 0.3 m whole links; 5 mm clearance from the disc.

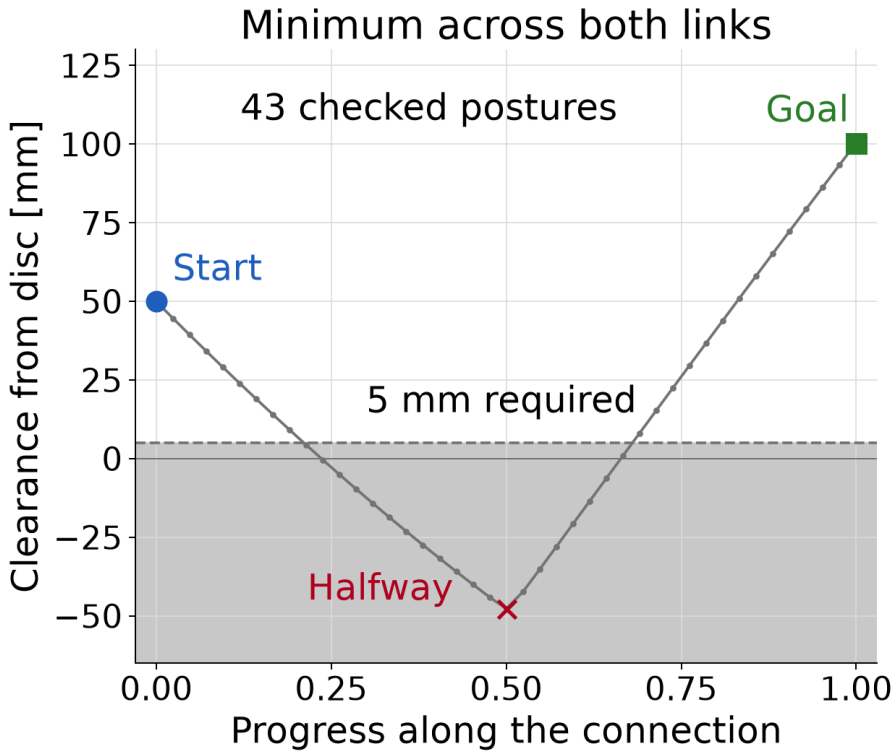
Joint range: –180° to 180° each; no wrapping or 60° elbow limit.

Endpoints: $(30^\circ, 45^\circ)$ to $(-11.517^\circ, 80.406^\circ)$; tip goal $(0.5, 0.2)$ m.

Predict: move both angles straight to their goals. Does the arm stay clear?

The direct connection crosses the obstacle

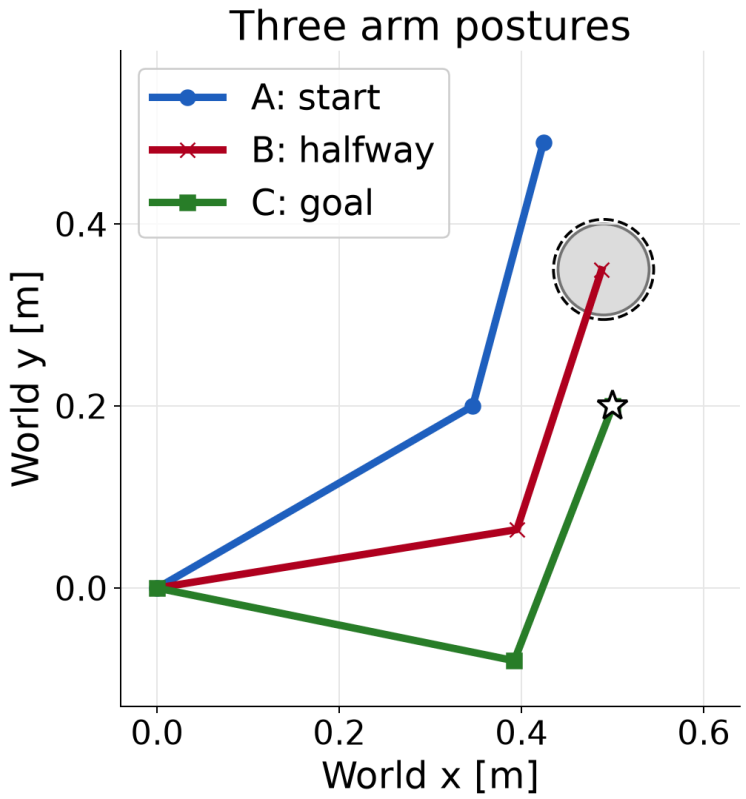


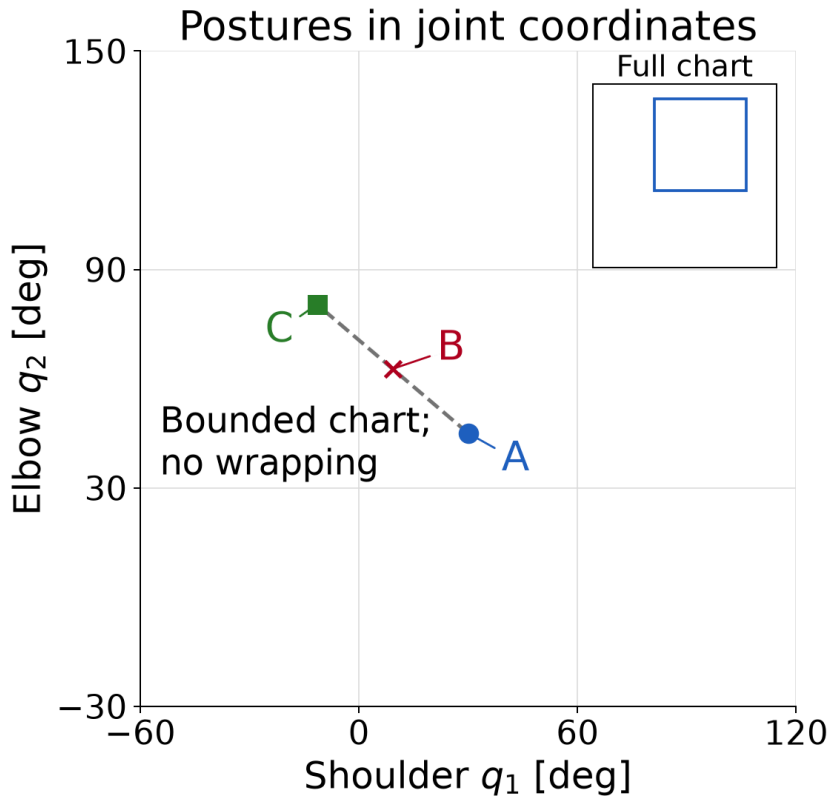


Halfway posture: second link penetrates the disc by 47.7 mm.

Clear endpoints do not establish a clear connection.

One arm posture becomes one point

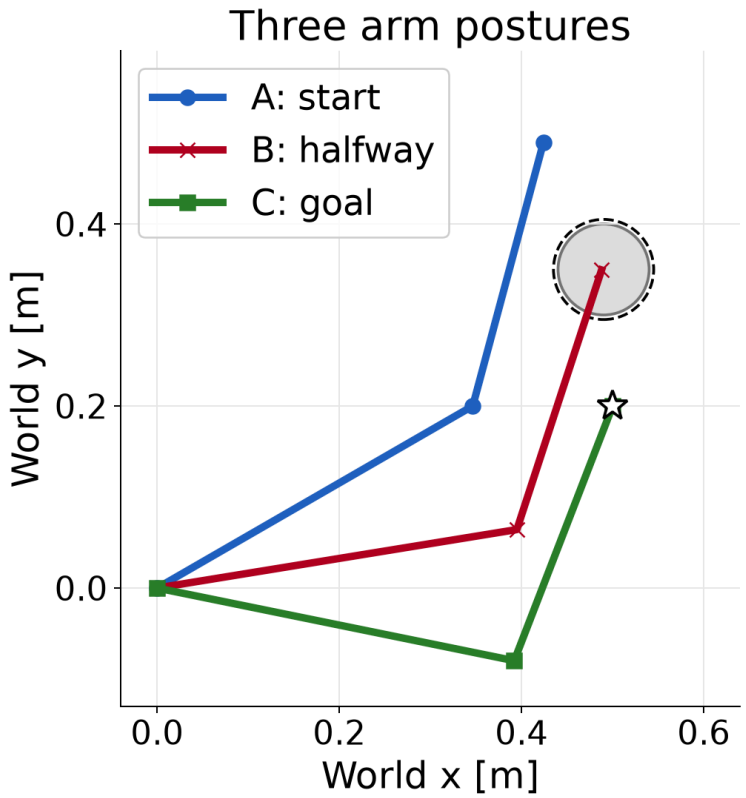


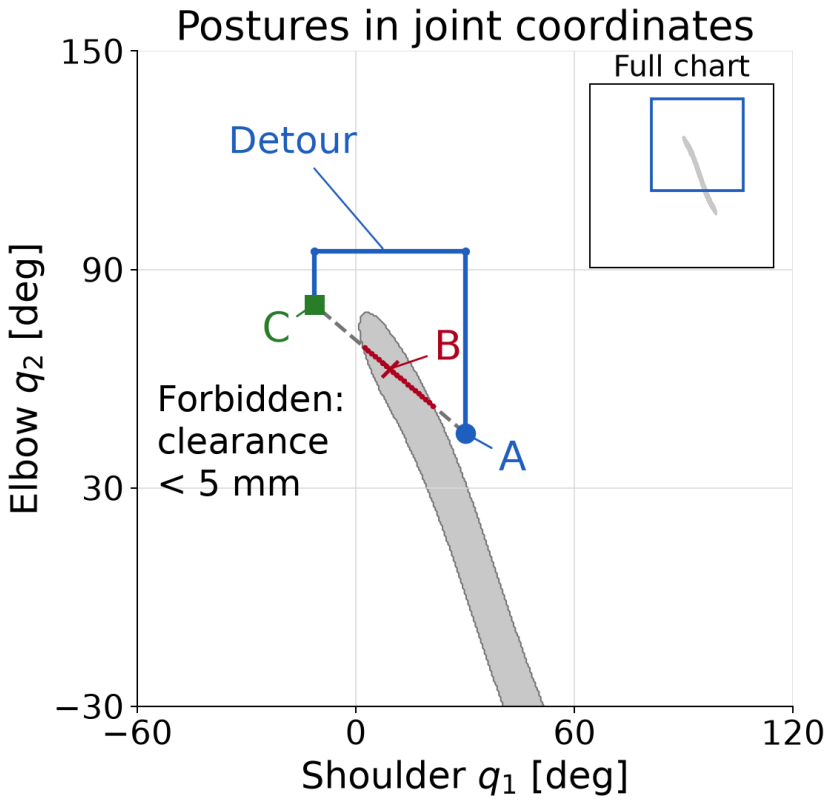


Horizontal coordinate: shoulder angle. **Vertical coordinate:** elbow angle.

The pair of angles specifies the whole arm posture.

Forbidden postures form regions





$$\mathcal{Q} = [-\pi, \pi]^2, \qquad \mathcal{Q}_{\text{free}} \subseteq \mathcal{Q}$$

Allowed: both links at least 5 mm clear. **Shaded:** violations. Axes use degrees.

A path specifies the route

Given: robot and obstacles, coordinate bounds, clearance rule, fixed start and goal.

Find a continuous path through allowed configurations:

$$\mathbf{q}_{\text{path}}(s) \in \mathcal{Q}_{\text{free}}, \qquad 0 \le s \le 1$$

$$\mathbf{q}_{\text{path}}(0) = \mathbf{q}_{\text{start}}, \qquad \mathbf{q}_{\text{path}}(1) = \mathbf{q}_{\text{goal}}$$

s: progress along the path; no time units.

A waypoint list needs a rule for connecting adjacent waypoints.

Timing specifies when each posture is reached

Same route, different progress schedules:

$$\mathbf{q}(t) = \mathbf{q}_{\text{path}}(s(t))$$

$$s(0) = 0, \qquad s(t_{\text{f}}) = 1$$

t: time in seconds. t_{f}: duration. $s(t)$: continuous, nondecreasing progress.

Geometric path	Timed trajectory
Which configurations, in which order	Which configuration at each time
Clearance along the route	Velocity and acceleration also matter

A connection needs checks between its endpoints

Configuration check: whole-link clearance at one posture.

Edge check: interpolate joint angles and test intermediate postures.

$$\mathbf{q}_{\text{edge}}(s) = (1 - s)\mathbf{q}_a + s\mathbf{q}_b, \qquad 0 \le s \le 1$$

Rule for this scene	Direct start-to-goal edge
At most 1° per joint between checks	42 subintervals; 43 postures (indices 0–42)
At least 5 mm clearance at every check	First failure at index 9; reject the edge

One failed intermediate posture rejects the edge.

Passing checks establishes clearance only at the checked postures.

rrt_two_link.py --edge ▶ check the edge live

Choose the free knots together

$$\mathbf{z} = (\mathbf{q}_1, \ldots, \mathbf{q}_T) \in \mathbb{R}^{nT}$$

- Current posture $\mathbf{q}_0$: fixed
- Remaining T postures: chosen together by trajectory optimization
- $n = 2, T = 20$: **40 decision coordinates**, 21 postures including the start

t is a **discrete configuration index**; RAI calls the T free ones **time slices**.

Predict: does one optimizer step move only the colliding knots?

Demo: optimize the posture sequence ▶ **run it live**

Joint-angle differences give angular velocity and acceleration

Equal intervals over normalized time 0–1.

Joint angular velocity over one interval:

$$\Delta t = \frac{1}{T}, \qquad \mathbf{v}_t = \frac{\mathbf{q}_t - \mathbf{q}_{t-1}}{\Delta t}$$

Joint angular acceleration from consecutive velocities:

$$\mathbf{a}_t = \frac{\mathbf{v}_t - \mathbf{v}_{t-1}}{\Delta t} = \frac{\mathbf{q}_{t-2} - 2\mathbf{q}_{t-1} + \mathbf{q}_t}{\Delta t^2}$$

Joint angular velocity uses **two knots**. Angular acceleration uses **three**.

Physical duration comes later.

Specify the start and scale the smoothness cost

Fixed prefix for an initially resting robot:

$$\mathbf{q}_{-1} := \mathbf{q}_0, \qquad \mathbf{a}_1 = \frac{\mathbf{q}_1 - \mathbf{q}_0}{\Delta t^2}$$

Smoothness cost: approximate integrated squared angular acceleration.

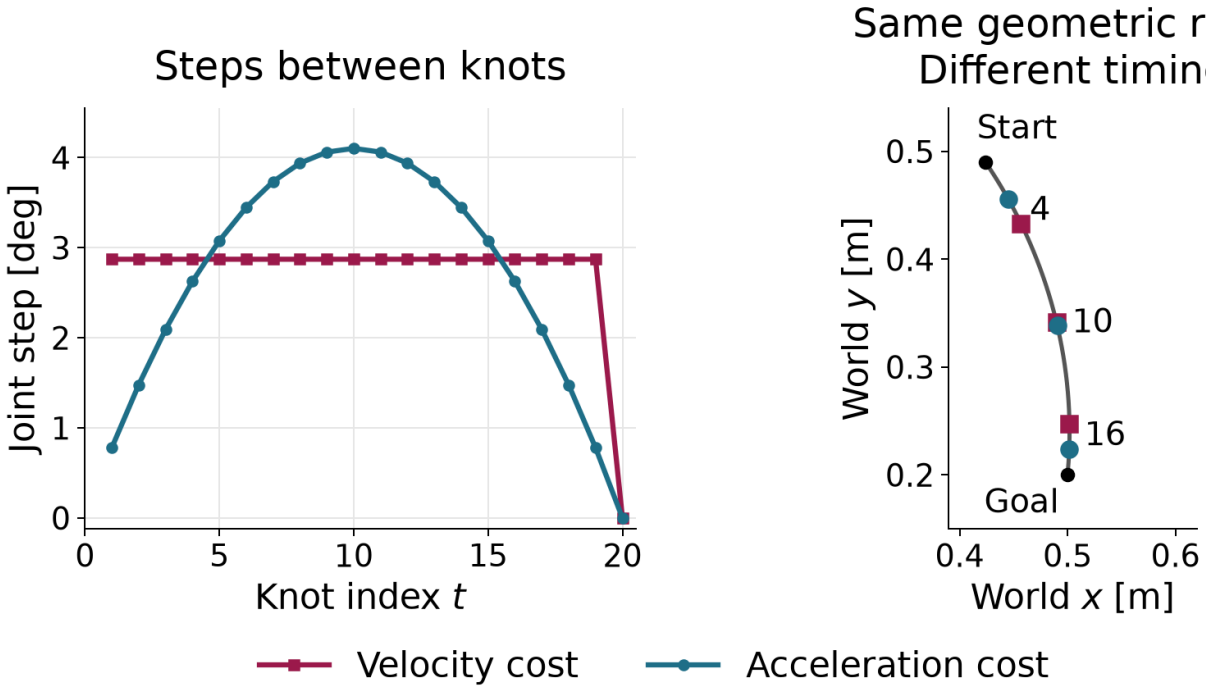
$$f(\mathbf{z}) = \sum_{t=1}^{T} \Delta t \, \|\mathbf{a}_t\|^2 = \sum_{t=1}^{T} \|\mathbf{r}_t\|^2$$

$$\mathbf{r}_t = \sqrt{\Delta t}\, \mathbf{a}_t \quad (\text{acceleration}), \qquad \mathbf{r}_t = \sqrt{\Delta t}\, \mathbf{v}_t \quad (\text{velocity})$$

This cost ranks motions; obstacle clearance is a separate requirement.

The first step is penalized; it is **not forced to zero**.

Velocity cost against acceleration cost



Same start, positive-elbow goal and final rest.

Cost	Joint step sizes
Velocity: first differences	19 steps of 2.87°, then zero
Acceleration: second differences	0.78° → 4.10° → zero

Both motions follow the same path, but **move along it at different speeds**.

Live comparison: **disc removed**.

Require the goal, final rest and obstacle clearance

Reach the selected goal and stop.

$$\mathbf{h}(\mathbf{z}) = \begin{pmatrix} \mathbf{q}_T - \mathbf{q}_{\text{goal}} \\ (\mathbf{q}_T - \mathbf{q}_{T-1})/\Delta t \end{pmatrix} = \mathbf{0}$$

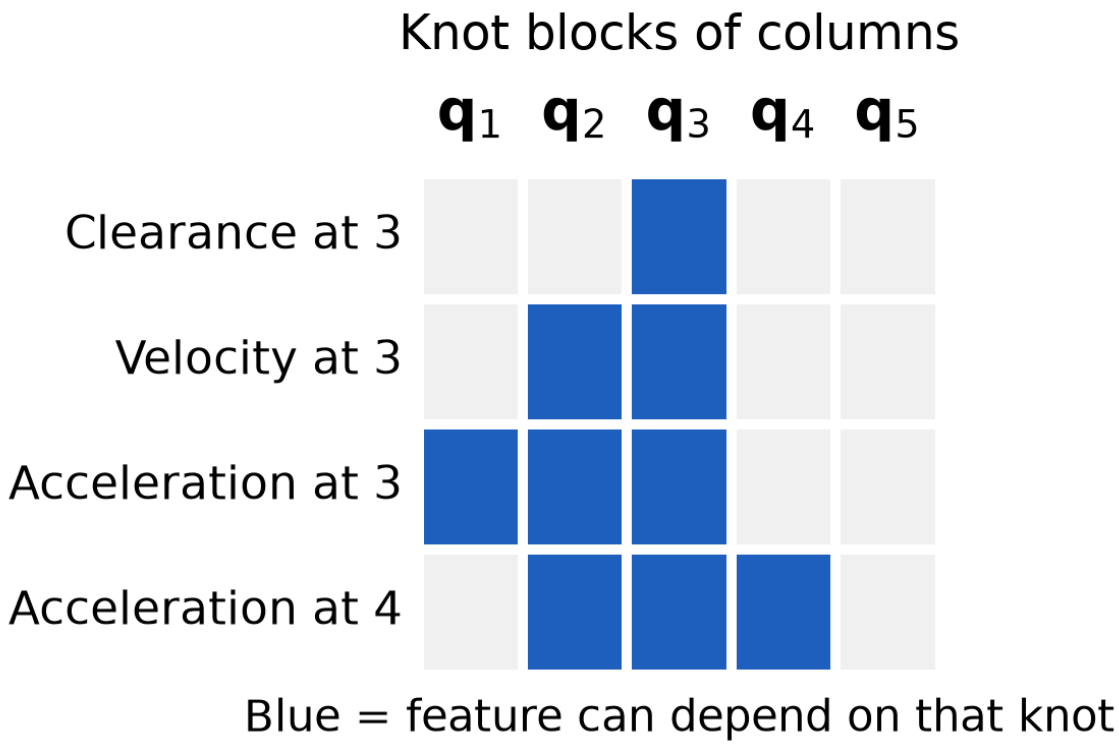
At each knot, keep every point of both links at least 5 mm from the disc.

$$g_{t,\ell} = r_o + d_{\text{req}} - \operatorname{dist}(\mathbf{c}, \mathcal{B}_\ell(\mathbf{q}_t)) \le 0$$

$\mathcal{B}_\ell$: link segment; $\mathbf{c}$: disc center; r_o: disc radius; $d_{\text{req}} = 5\,\text{mm}$.

Check the motion between knots separately.

Each feature depends on nearby knots



Columns: joint coordinates, grouped by knot.

Rows: motion or task feature groups.

Feature	Knots used
Clearance	One
Velocity	Two neighbors
Acceleration	Three neighbors

Sparse Jacobian; block-banded normal matrix.

Neighboring-knot features define the KOMO program

$$\min_{\mathbf{q}_{1:T}} \sum_{t=1}^{T} \left\| \mathbf{r}_t(\mathbf{q}_{t-k:t}) \right\|^2$$

$$\mathbf{g}_t(\mathbf{q}_{t-k:t}) \leq \mathbf{0}, \qquad \mathbf{h}_t(\mathbf{q}_{t-k:t}) = \mathbf{0}$$

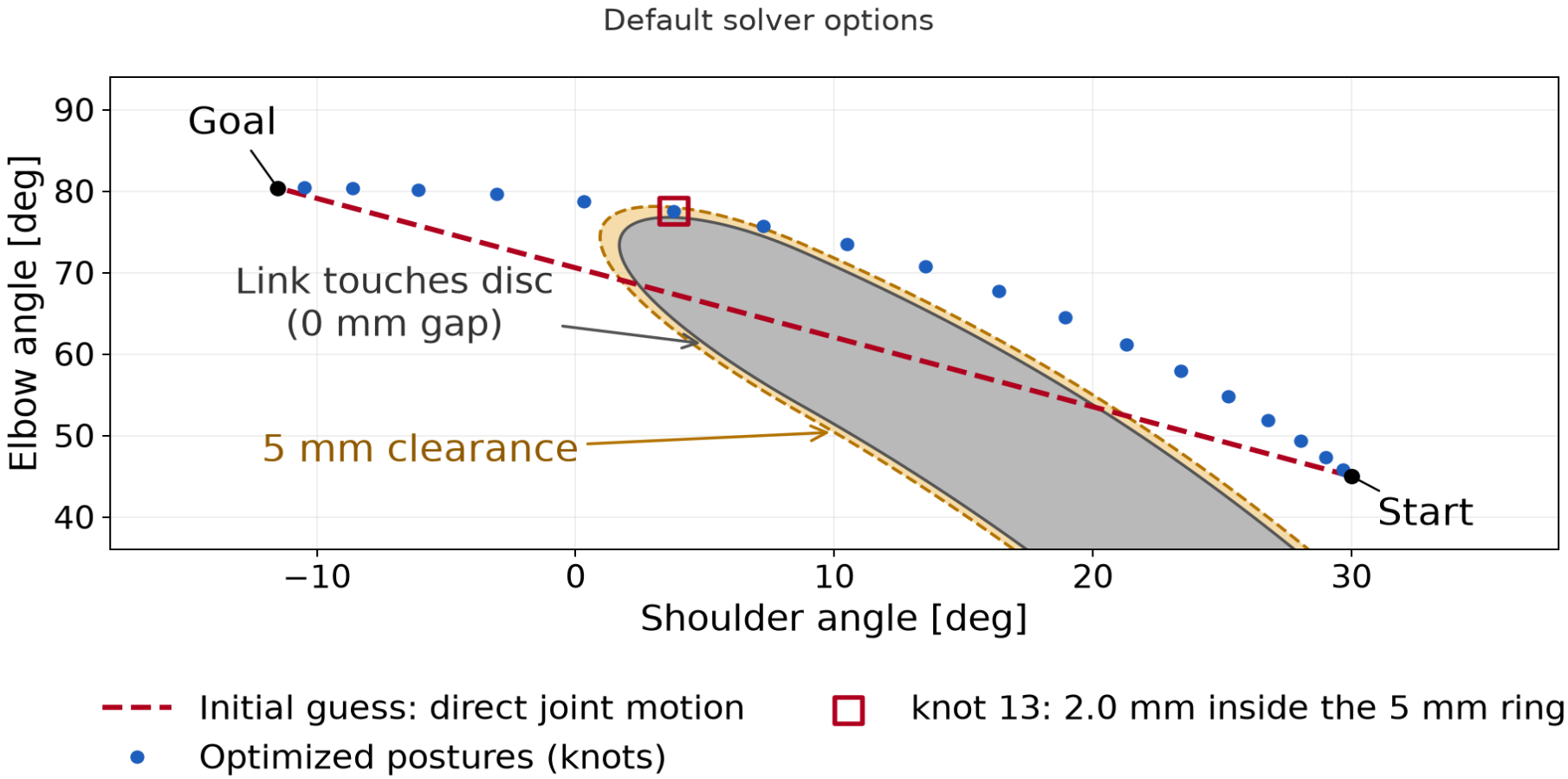
k: largest difference order; here $k = 2$.

k-order Markov Optimization (KOMO) describes the structured formulation.

RAI's **augmented Lagrangian** adjusts the knot sequence using damped Newton steps.

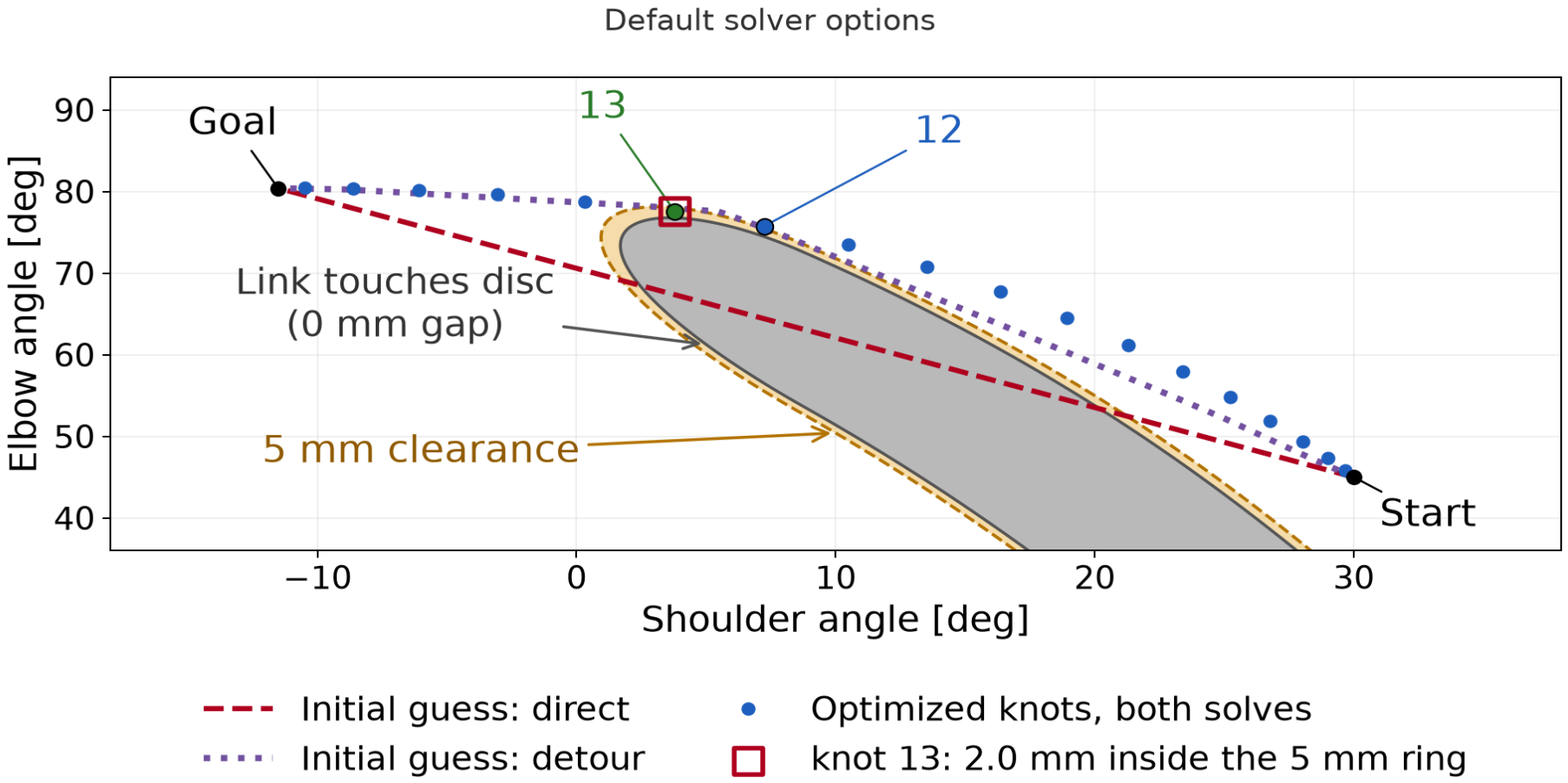
Local structure helps computation. It does not imply a global optimum.

RAI's default call can stop short of the requirement



Knot 13 misses the 5 mm requirement by 2.0 mm. RAI reports `feasible`.

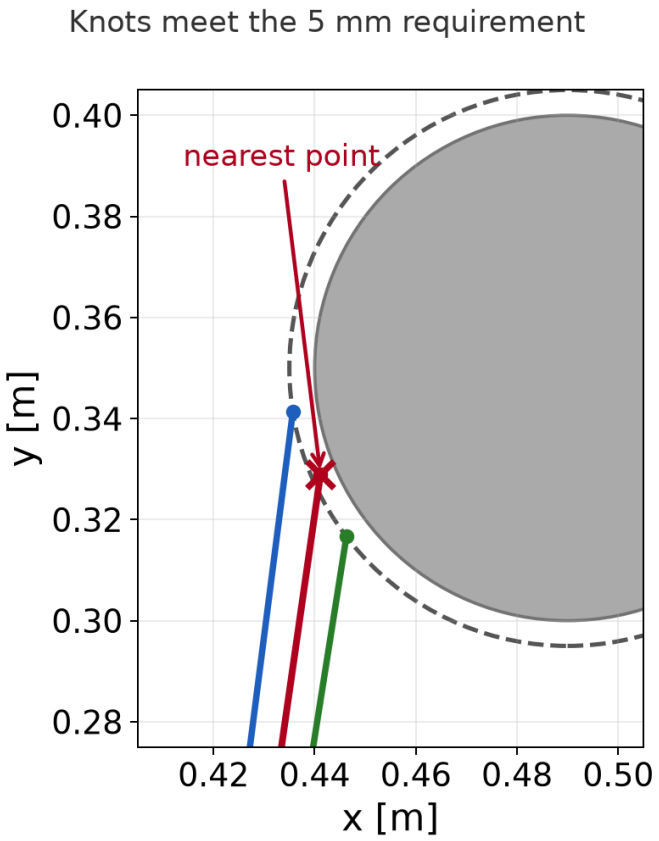
Two initial guesses give the same result here



Same endpoint postures. Both solves return the same knots here.

Check the motion between knots 12 and 13

Now examine a solution whose knots meet the clearance requirement.



Blue: knot 12. Green: knot 13. Red: halfway posture.

Dashed: 5 mm ring. Gray: physical disc.

Knots: **5 mm clearance**, within numerical tolerance.

Halfway: **1.8 mm inside the clearance ring**, outside the physical disc.

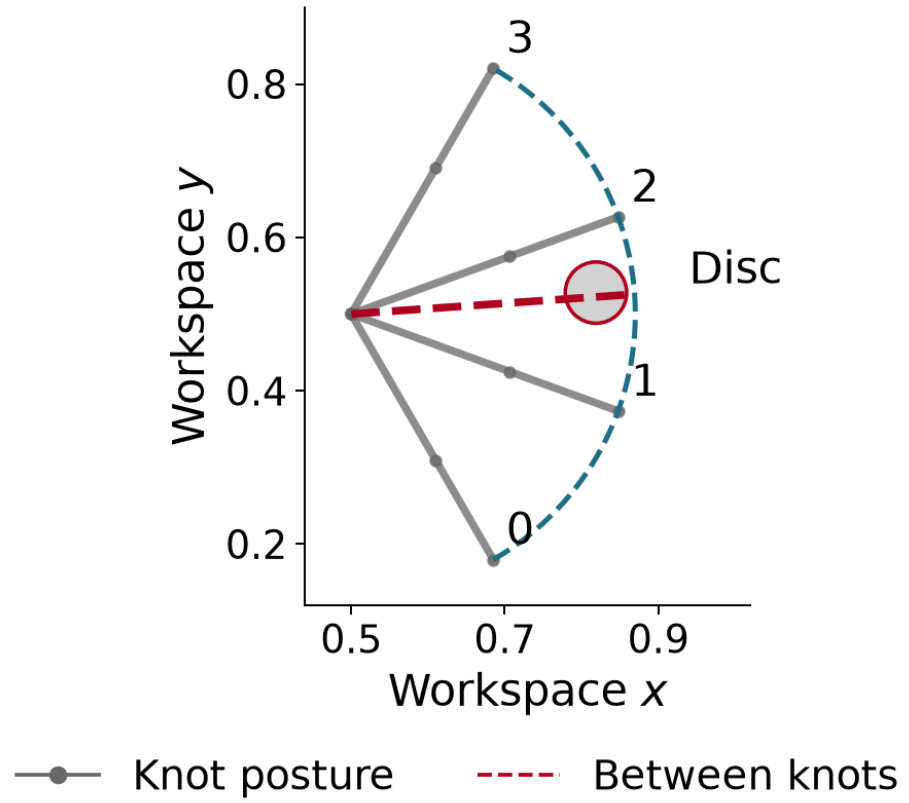
Reference settings: backup 46 · fig/komo-shapes-gap.png

Live demo: clear knots can hide a collision

- **Four gray postures:** clear; obstacle cost **0**
- **Red posture between knots:** collision
- **Add midpoint knots:** positive obstacle cost

Extra checks expose the collision.
The motion stays the same.

Separate demo arm; soft collision penalty, no extra margin.



modules/opt-explorer · Trajectory tab · Between-knot collision ▶ run it live

Break

We can optimize a knot sequence and inspect what its constraints omit.

Next: construct candidate routes by testing sampled connections.

Search using geometry checks as needed

Operation	Choice for this two-link arm
Sample a posture	Draw each joint angle uniformly within its allowed range.
Compare postures	Use the Euclidean distance between their joint-angle vectors.
Connect two postures	Interpolate both joint angles along a straight segment in joint space.
Check the connection	Space checks by at most 1° per joint; require both complete links to stay at least 5 mm from the disc at every check.

Check selected postures and connections without constructing the full forbidden region.

Probabilistic roadmap (PRM): build and query a graph

Inputs: robot and obstacle geometry, joint limits, required clearance, sampling budget and neighbor-selection rule.

Query: start and goal postures.

Output: a route through checked edges, or no route found in this roadmap.

Construction · use the checks from slide 22

1. **Sample** configurations; keep valid ones as graph nodes.
2. **Test** nearby pairs; store accepted edges with joint-space length as cost.

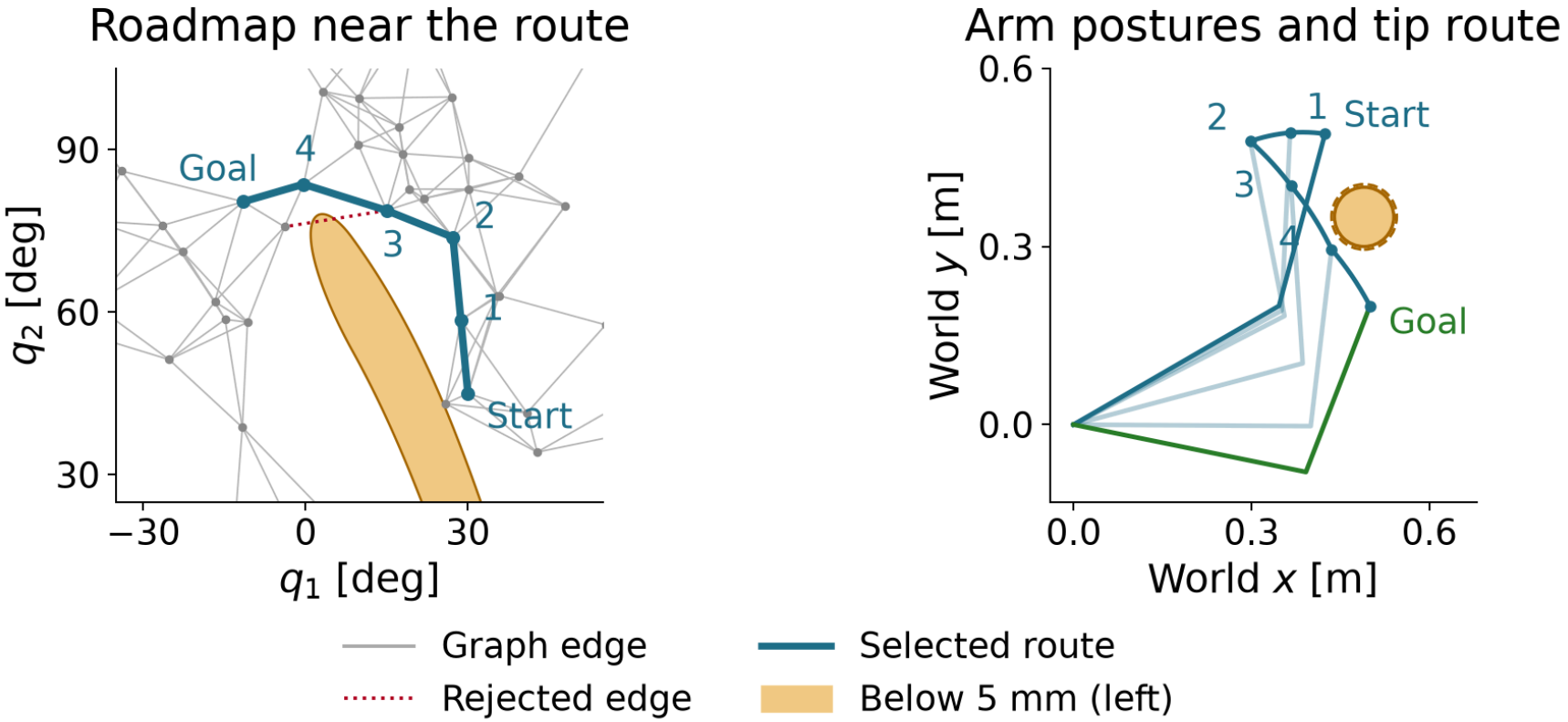
Query

3. **Validate** start and goal; attach them through checked edges.
4. **Search** for minimum total edge cost; report if no route exists.

Reuse the roadmap for new start–goal queries with the same scene, joint limits, clearance requirement and checking resolution.

Kavraki, Švestka, Latombe & Overmars (1996)

A graph search selects a route through the roadmap



Selected route: five checked connections from start to goal.

Shortest in this graph; not a claim about every possible path.

Expand the cheapest known route first

Dijkstra's search on nonnegative edge lengths

1. **Start** cost is zero; all other tentative costs are infinite.
2. **Settle** (declare final) the unsettled node with smallest tentative cost.
3. **Try** each neighbor through that node; keep an improvement.
4. **Stop** when the goal is settled; follow parent pointers backward.

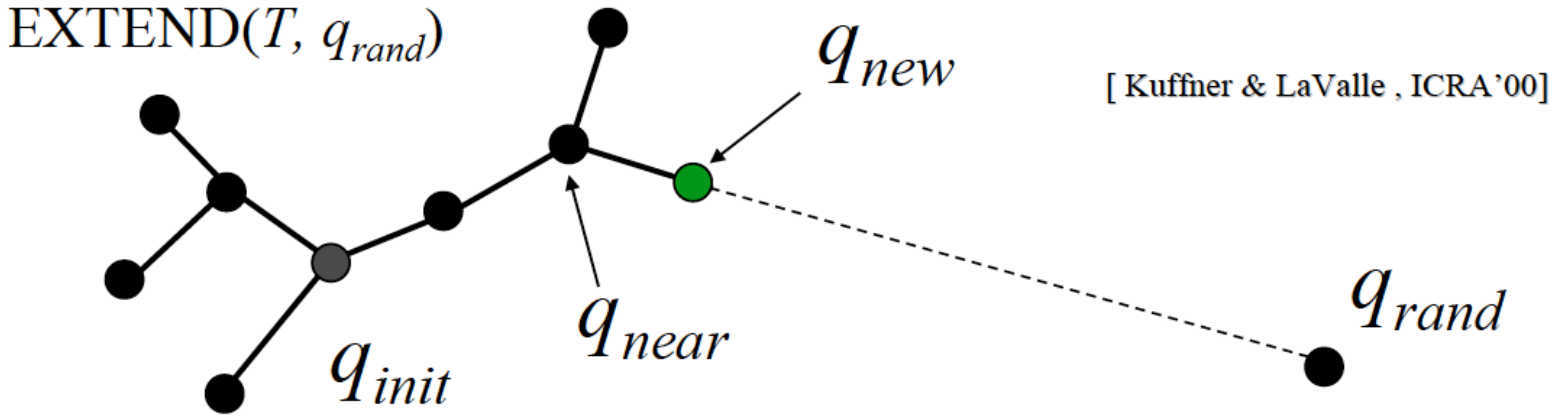
$$D(v) \leftarrow \min\{D(v),\ D(u) + w(u, v)\}$$

D: route length from start. w: checked edge length.

The result is shortest among routes in this graph.

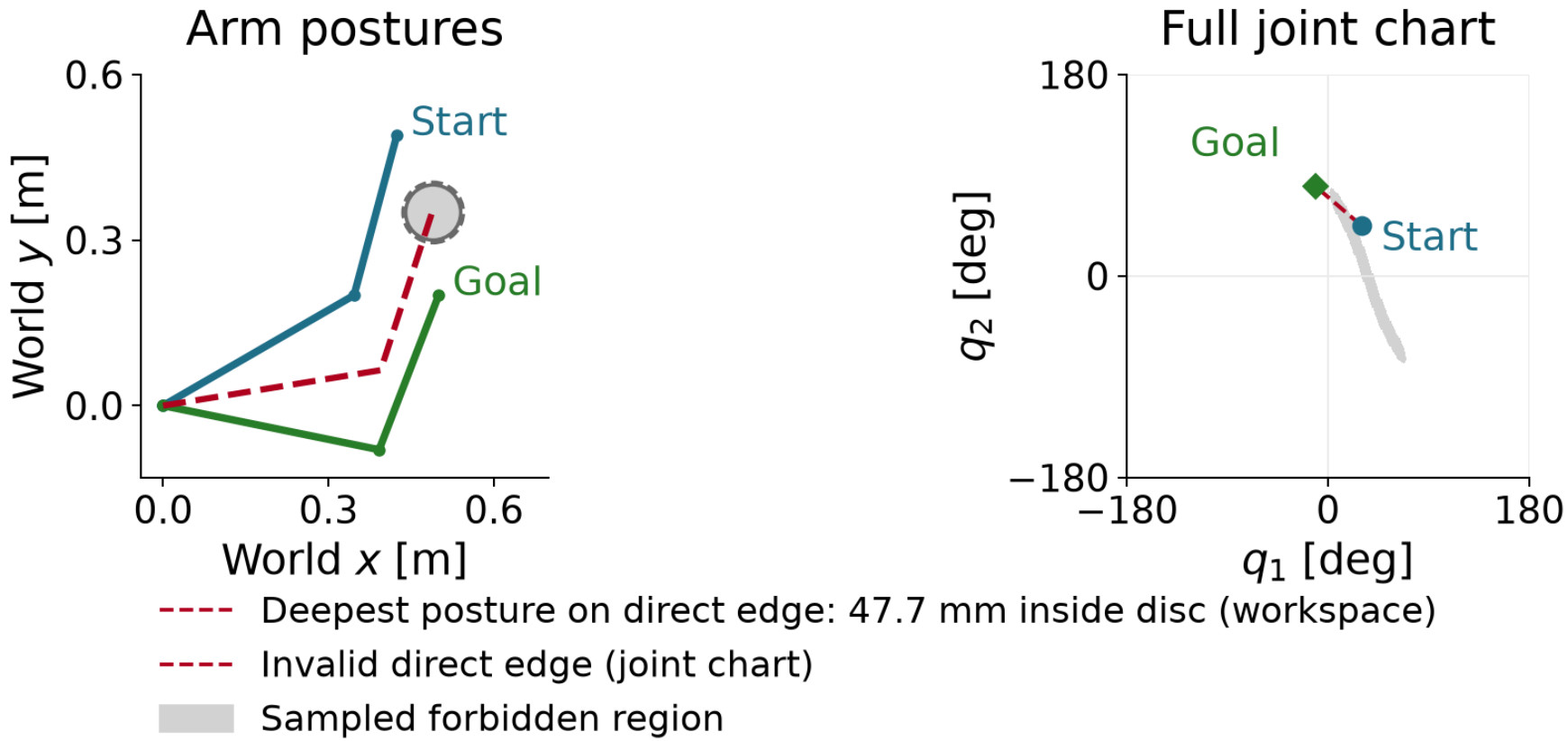
RRT: grow a tree toward sampled configurations

1. **Sample** $\mathbf{q}_{\text{rand}}$; choose the goal with probability p_{goal} (e.g., 0.1).
2. **Find** the nearest tree node $\mathbf{q}_{\text{near}}$.
3. **Move** at most d_{step} (e.g., 10°) toward the sample to propose $\mathbf{q}_{\text{new}}$.
4. **Check** the edge; add the new node and parent only if it passes.
5. **Try** a checked goal connection when the goal is within d_{step} of $\mathbf{q}_{\text{new}}$.



LaValle (1998); figure reused from 2026 Motion Planning, slide 31 (credit in figure).

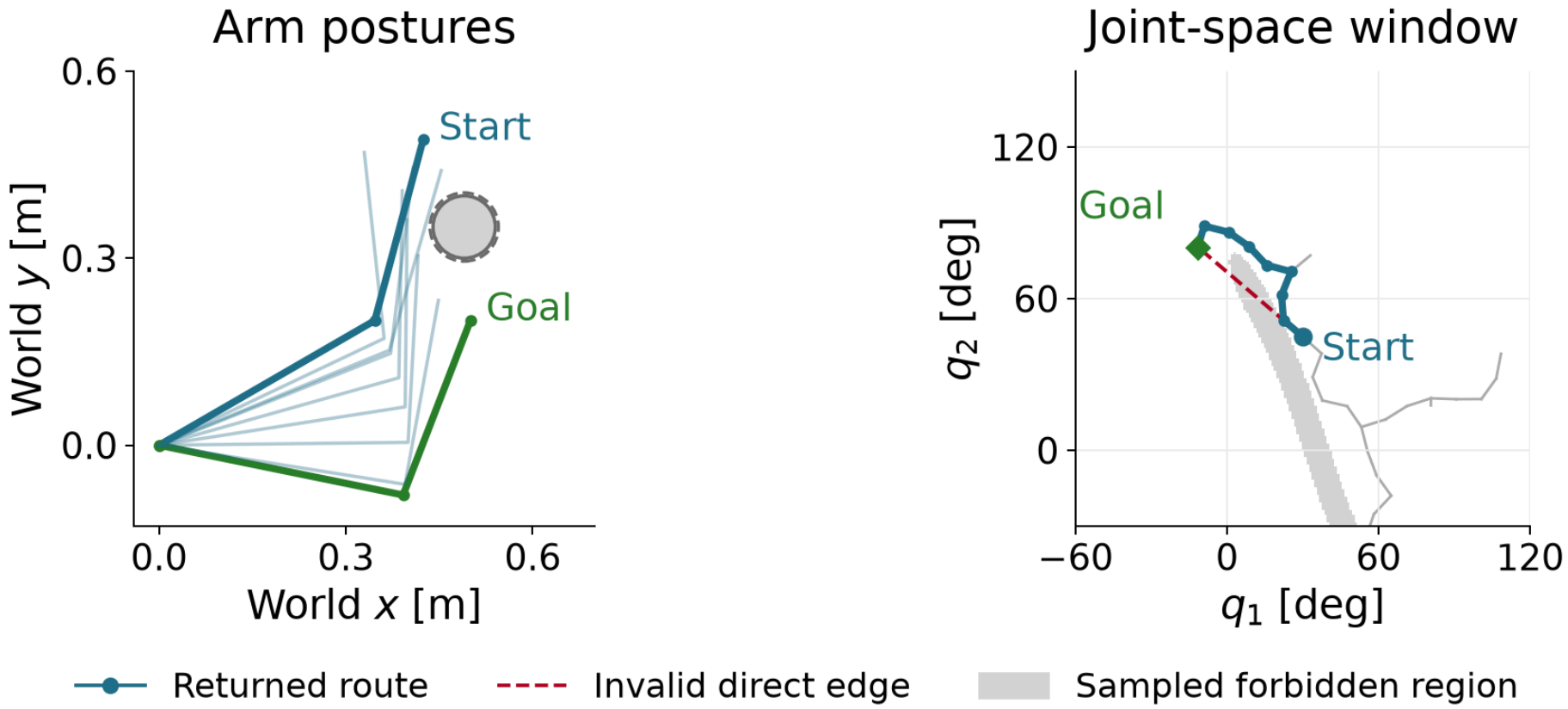
Watch the tree decide whether to grow



Before Step: predict whether the proposed connection will pass.

rrt_two_link.py ▶ step through RRT

Extract the start-to-goal path from the tree



Playback pace is illustrative; execution timing remains to be chosen.

rrt_two_link.py ▶ finish the RRT

Bidirectional RRT: grow from both ends

Initialization

1. **Root** one tree at the start and the other at the goal.

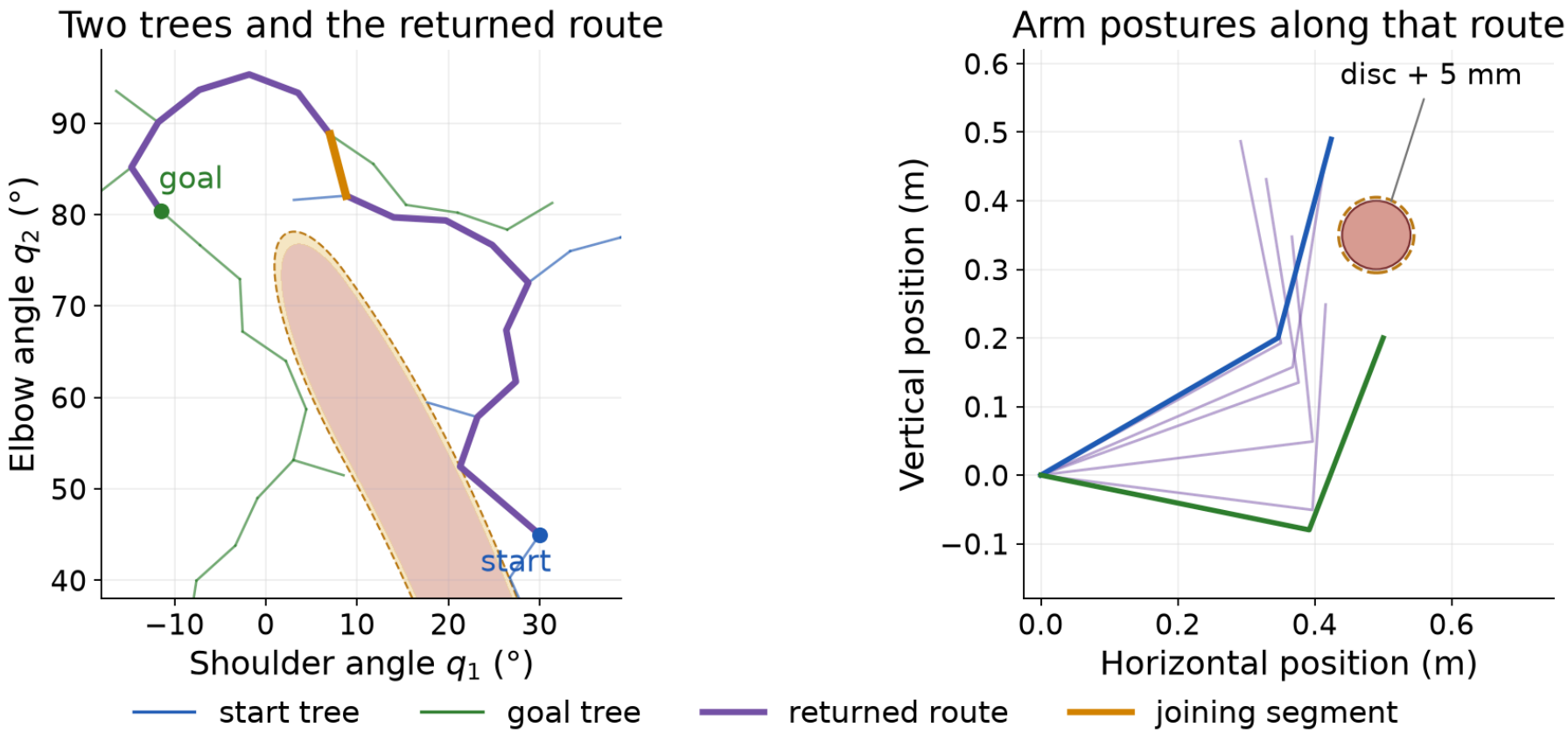
Repeat until joined or the budget is exhausted

2. **Sample a target:** draw a posture uniformly within the joint limits, or randomly select a node in the other tree.
3. **Extend** the active tree toward it; add the proposed node if the extension passes its checks.
4. **Join** if the new node is close enough to the other tree; otherwise let the other tree attempt an extension.

Check the arm's clearance along every segment of the final route.

RAI-style procedure; detailed notes §8.5.

Two trees supply one start-to-goal route



Join the parent chains, then check the returned path.

RAI-style bidirectional RRT; detailed notes §8.5 ▶ grow two trees

Probabilistic completeness

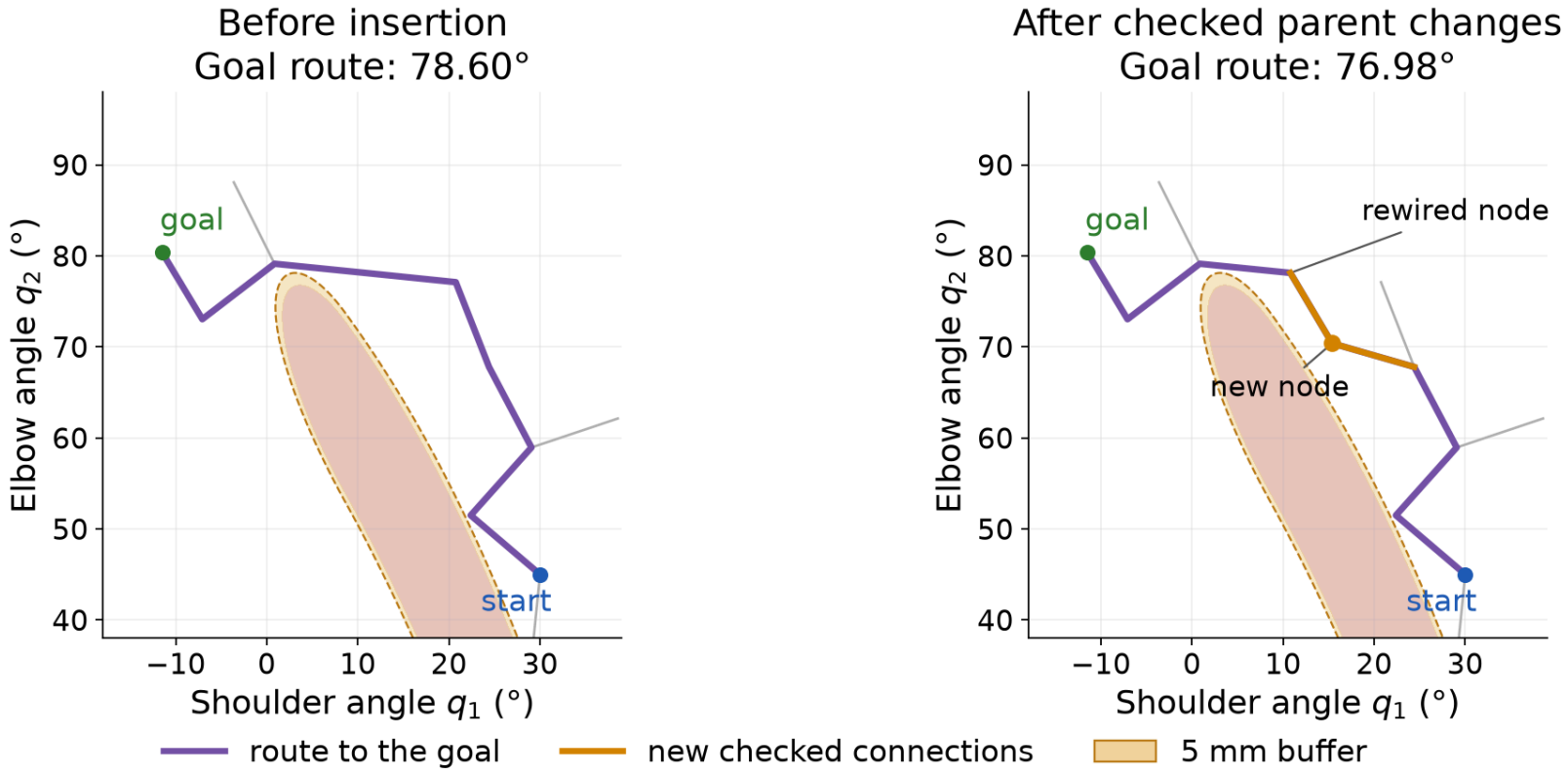
> For a problem with a valid path, the probability of finding one **approaches 1 as the number of samples grows**, under suitable assumptions.

Assumes a route with clearance to spare, suitable sampling and connections, and sound validity tests.

A finite failed search does not prove that no path exists.

Karaman & Frazzoli (2011); detailed notes §8.6 ▶ exhaust a small budget

RRT*: improve routes by changing parents



Choose the cheapest checked nearby parent.

Rewire a neighbor if a checked route through the new node is cheaper.

Descendants inherit the saving.

Karaman & Frazzoli (2011), Algorithm 6; one-insertion illustration ▶ **step through rewiring**

After finding a route, shorten it with checked shortcuts

Input: a path that passes the connection check.

Repeat for the chosen attempt budget

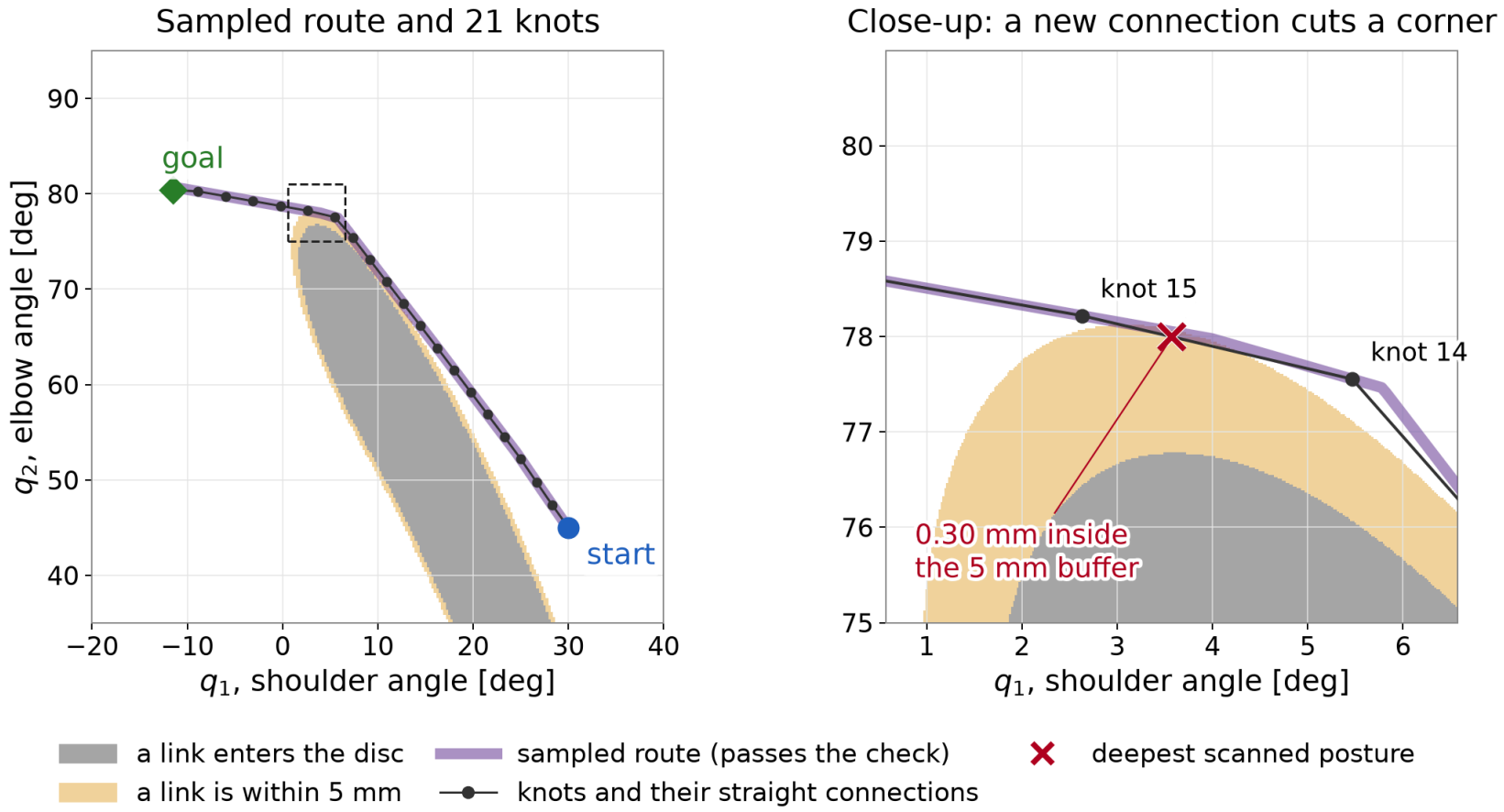
1. **Select** two positions along the current path.
2. **Check** the straight joint-space connection between them.
3. **Replace** the intervening detour if the whole candidate path passes; otherwise keep the current path.

Works on routes from **PRM, RRT and other planners**.

A shorter route still has corners and needs timing.

Detailed notes §9.1 ▶ try shortcuts

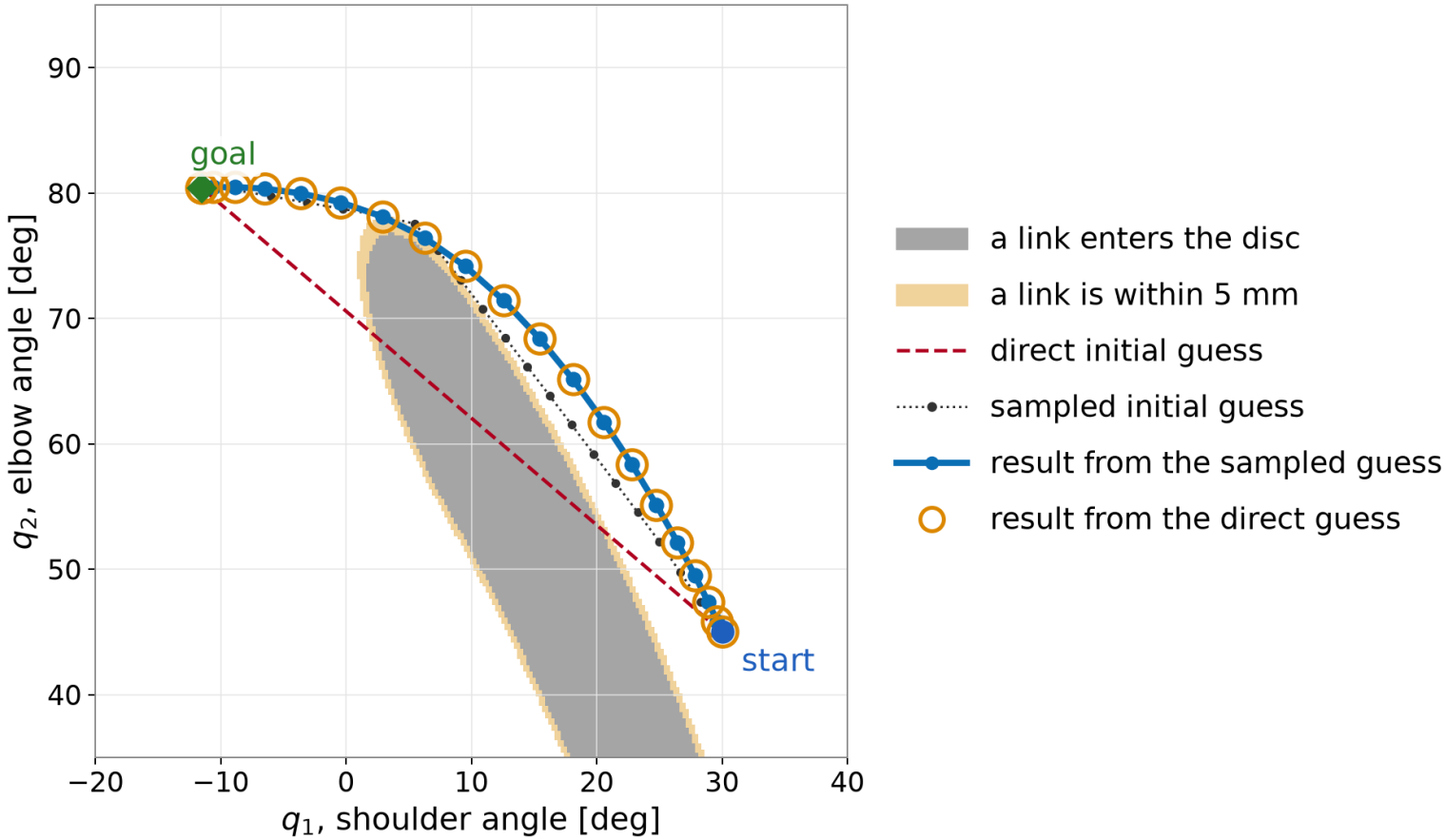
Use the sampled route to initialize optimization



The new connections can cut across the route's corners.

Whole-link example · Detailed notes §9 · Supporting comparisons: backup 49 ▶ prepare the knots

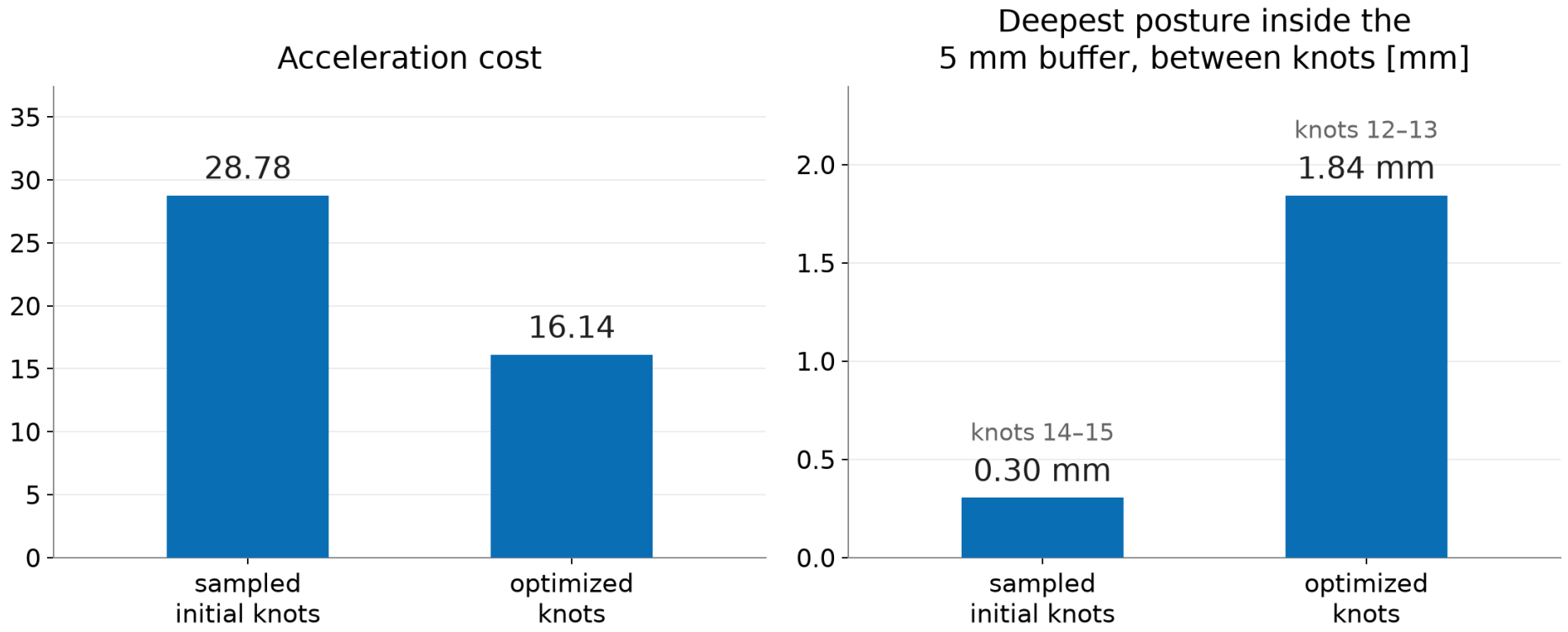
Both initial guesses reach the same knots here



A sampled start need not produce a different solution.

Whole-link example · Detailed notes §9 · Supporting comparisons: backup 49 ▶ compare results

A lower cost does not establish a valid motion



Knot clearance deficits are below 10^{-6} mm; the 1.84 mm deficit lies between knots.

Check the connections again after optimization.

Whole-link example · Detailed notes §9 · Supporting comparisons: backup 49 ▶ check the refined motion

Choose timing after checking the route

Here: assign timing to the checked RRT route (backup 43).

$$\mathbf{q}(t) = \mathbf{q}_{\text{path}}(s(t))$$

t: physical time in seconds; s: progress along the route, from 0 to 1.

Choose a schedule that respects joint speed and acceleration limits.

Trajectory optimization can also choose the route and timing together.

Check the motion before execution

Check	Use the execution model
Clearance	Actual interpolation, robot shapes and self-collisions
Joint limits	Positions, velocities and accelerations under the chosen timing
Actuator limits	Required forces or torques from a dynamics model

A controller tracks the timed reference

Planner output: a validated route and a schedule checked against motion limits.

Controller input: reference motion and measured robot state.

Controller output: commands through a stated robot interface.

$$\text{reference} \longrightarrow \text{controller} \longrightarrow \text{robot}$$

$$\text{robot} \longrightarrow \text{measured state} \longrightarrow \text{controller}$$

Tracking error and model error still require feedback and monitoring.

State what the planned motion establishes

- **Optimization:** adjust coupled knots under written costs and constraints.
- **Sampling:** search checked connections for a route.
- **Refinement:** shorten or smooth, then validate the changed motion.
- **Execution handoff:** choose timing, check limits, and track the reference.

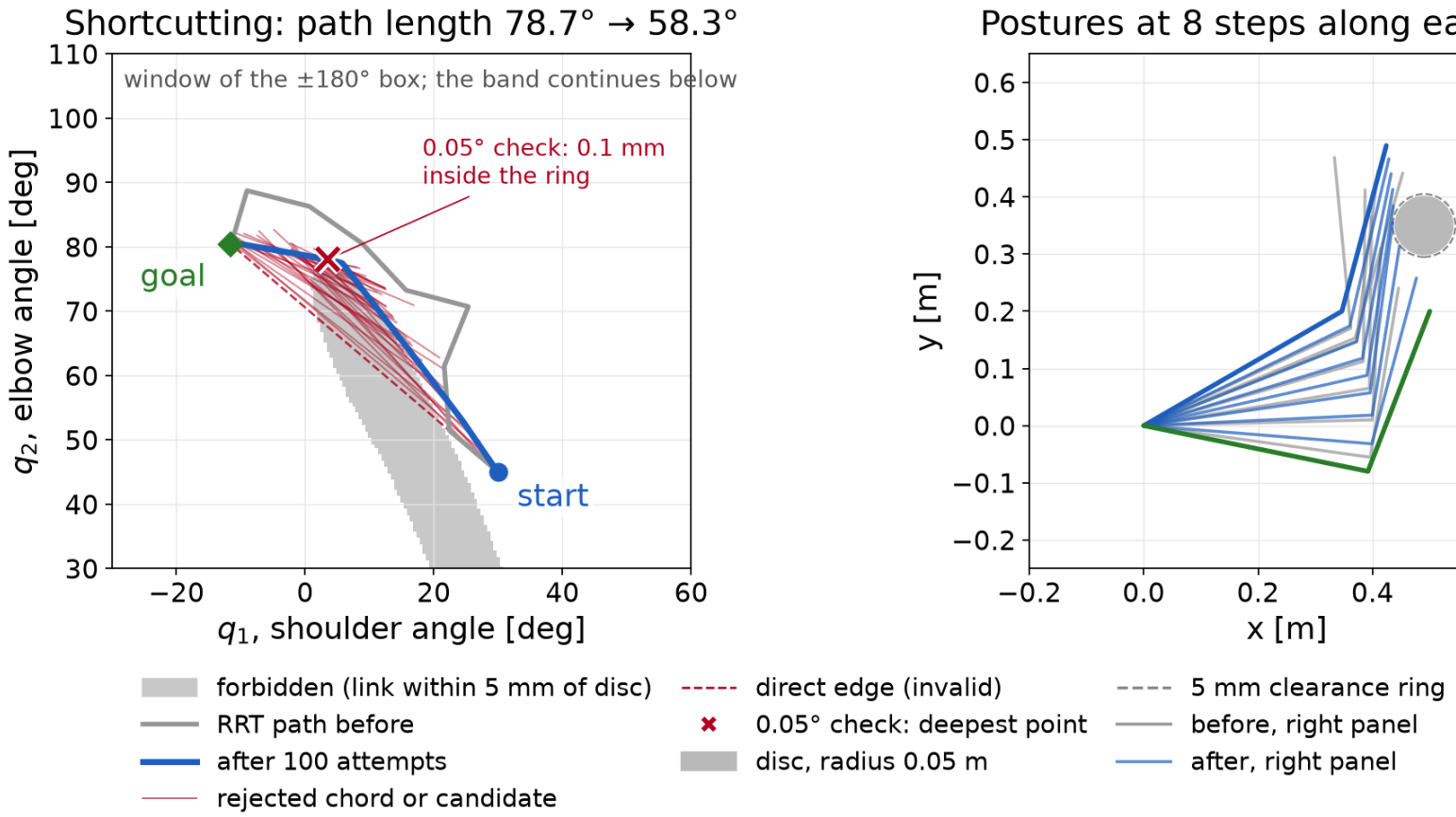
Next, session 5.1: learned samplers, feasibility predictions and search heuristics; retain validation.

R 5: quiz 2 (Gauss-Newton, KKT, KOMO, control stack, RRT); A1 due.

Backup slides

Retained Jacobian bookkeeping, operation counts, solver comparison and API mapping.

Backup · A finer check can reject the shortened route



Length: 78.71° → 58.34°. **1° check:** minimum clearance 5.1 mm.

0.05° check: about 0.1 mm inside the required 5 mm ring.

Optional detail: relate checking spacing to robot size and clearance (backup 43).

Detailed notes §9.1 ▶ shorten and recheck

Backup · Bound motion between checked postures

Link lengths are $l_1 = 0.4\,\mathrm{m}$, $l_2 = 0.3\,\mathrm{m}$.

For any material point on either link under joint-linear interpolation:

$$\|\Delta \mathbf{p}\| \le (l_1 + l_2)|\Delta q_1| + l_2|\Delta q_2|$$

For 1° changes in both joints:

$$(0.7 + 0.3)\frac{\pi}{180}\ \mathrm{m} \approx 17.45\ \mathrm{mm}$$

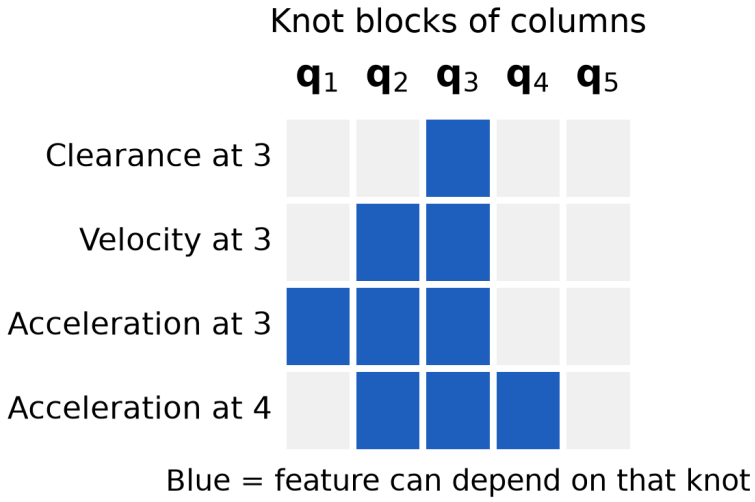
From the nearer checked endpoint: **half the bound, at most 8.73 mm**, for any point on either link.

Sufficient on every subinterval: both endpoint clearances ≥ **13.73 mm**.

The shortcut's **5.1 mm** does not meet that condition.

Optional derivation; not required for the main lecture.

Backup · Locality and operation counts



Full feature Jacobian: 84×40 for 20 two-joint slices.

The normal matrix is block-banded.

Estimated solve work: banded $O(T(k+1)^2 n^3)$; dense $O((nT)^3)$.

Backup · Two Jacobians, both sparse

Rows	Full feature Jacobian	Inner residual Jacobian at solution
Smoothness	40	40, times $\sqrt{2}$
Endpoint and rest	4	4, times $10\sqrt{2\mu}$
Whole-link clearance	40	2 active rows, times $10\sqrt{2\mu}$
Size	84×40	46×40

Only link 2 at knots 12 and 13 has a positive multiplier.

The active set and multiplier shifts can change the inner residual system.

Backup · Check solver residuals in task units

Same program and 5 mm requirement · RAI `robotic` 0.4.1

Solver configuration	Largest knot deficit
Untouched numerical options	2.0 mm
Constraint rows × 10, tight stops	Below 10^{-6} mm

Both report `feasible`. Check the residuals.

The accurate knot solutions still fail the between-knot check.

stage4_komo_rai.py · modules/motion-planning/fixtures/stage4_rai.json

Backup · The shape-based program in RAI

```
komo = ry.KOMO(C, phases=1, slicesPerPhase=20,
               kOrder=2, enableCollisions=False)
komo.addControlObjective([], 2, 1.)
komo.addObjective([1.], ry.FS.qItself, [], ry.OT.eq, [1.], q_goal)
komo.addObjective([1.], ry.FS.qItself, [], ry.OT.eq, [1.], order=1)
for link in ("link1", "link2"):
    komo.addObjective([], ry.FS.negDistance,
                      [link, "disc"], ry.OT.ineq, [1.])
```

Capsule links and an adjusted disc sphere reproduce the segment-distance rows.

Feature scales and solver settings are separate choices.

Joint bounds are checked afterward; this program has no bound rows.

stage4_komo_rai.py · modules/motion-planning/fixtures/stage4_rai.json

Backup · One free knot recovers a posture cost

$$T = 1, \qquad \Delta t = 1, \qquad \mathbf{q}_{-1} = \mathbf{q}_0$$

$$\mathbf{r}_1 = \mathbf{q}_1 - \mathbf{q}_0, \qquad f = \|\mathbf{q}_1 - \mathbf{q}_0\|^2$$

Deck 05's convention:

$$\frac{1}{2}\|\mathbf{q} - \mathbf{q}_{\text{ref}}\|^2_{\mathbf{D}}, \qquad \mathbf{q}_{\text{ref}} = \mathbf{q}_0, \quad \mathbf{D} = 2\mathbf{I}$$

Omit terminal rest; with one interval it would force the only step to zero.

Backup · Compare refinement and solver accuracy

Same whole-link model and 5 mm clearance requirement.

Sequence	Intervals	Acceleration cost	Largest scanned deficit between knots
Resampled shortcut	20	28.78	0.30 mm
Optimized, RAI defaults	20	16.19	2.54 mm
Optimized, tighter accuracy	20	16.14	1.84 mm
Optimized, tighter accuracy	40	16.39	0.43 mm

Defaults also miss clearance **at a knot by 2.00 mm**; solver details: backup 46.

Original shortcut: 1° check passes; 0.05° scan finds a **0.09 mm** deficit (backup 42).

staged.md §4; fixtures/stage4_rai.json; opt-explorer/fixtures/refine_rai.json