Lecture #:24 *Planning in Task Space Regions*

Scribe: *Vivek Tiwari, Anirudh Kulkarni, Jingfeng Zhou, William Wang, Damon Acoba*

November 18, 2019

Contents

1 Recap

1.1 Forward Kinematics

Previously we covered that we can represent the local frame of an object with respect to a certain frame as the product of relative transformation matrices as follows:

$$T_n^0 = T_1^0 \cdot T_2^1 \dots T_n^{n-1} \tag{1}$$

Where T_n^0 represents the frame of object n with respect to origin 0 and each transformation takes the form of:

$$T = \begin{bmatrix} R & P \\ 0 & 1 \end{bmatrix} \tag{2}$$

R is the rotation matrix and P the translation column vector. We also note the inverse of the transformation as:

$$T^{-1} = \begin{bmatrix} R^{-1} & -R^{-1}P \\ 0 & 1 \end{bmatrix} \tag{3}$$

In equation (3) $R^{-1} = R^T$ due to the orthogonality constraint for a rotation matrix.

2 Constraint Representation

2.1 General Representation

$C(q)$ represents the "constraint evaluation function", which determines whether the constraint is met for a certain configuration q. It is a function of the configuration only. We represent a path in configuration space as $\tau : [0,1] \to Q$, where 0 and 1 are placeholders for the start and end configuration and Q is the configuration space of the robot. Constraints are generally represented as the pair:

$$\{C(q), S\} \ s.t. \ C(q) \in \mathbb{R} \geq 0 \tag{4}$$

S represents the domain of the constraint, i.e. when we want to enter the constraint and check its validity, it is represented as the subset

$$S \subseteq [0,1] \tag{5}$$

Therefore, S is an interval such as $[0.5, 0.6]$ because it represents a domain through time.

2.2 Constraint Manifold

The Constraint Manifold is the set of all possible configurations that satisfy the given constraint C(q).

$$M_C = \{q \in Q : C(q) = 0\} \tag{6}$$

For example, take the line y=x. If we want to constraint a robot to move along this line, it would be beneficial to define a constraint like C(x, y) = y-x. If the robot is on this line, the value of the constraint will be 0.

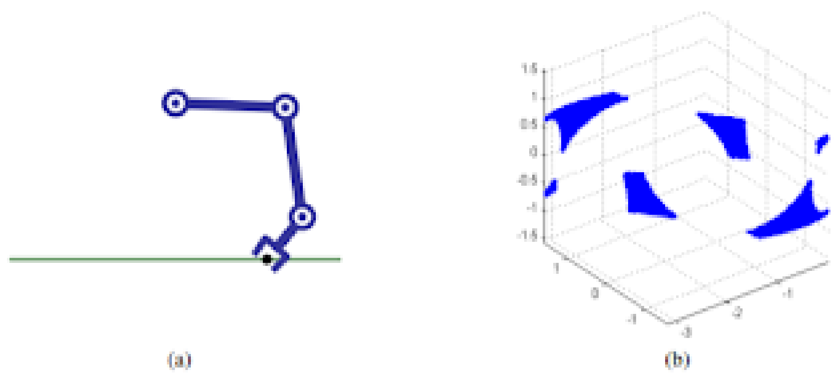


Figure 2: (a) Pose constraint for a 3-link manipulator: The end-effector must be on the line with an orientation within ±0.7rad of downward. (b) The manifold induced by this constraint in the C-space of this robot.

3 Constraint Path Planning

In this section we cover three sampling strategies to satisfy constraints in path planning, Rejection Sampling, Projection Sampling and Direct Sampling depicted in the figure below.

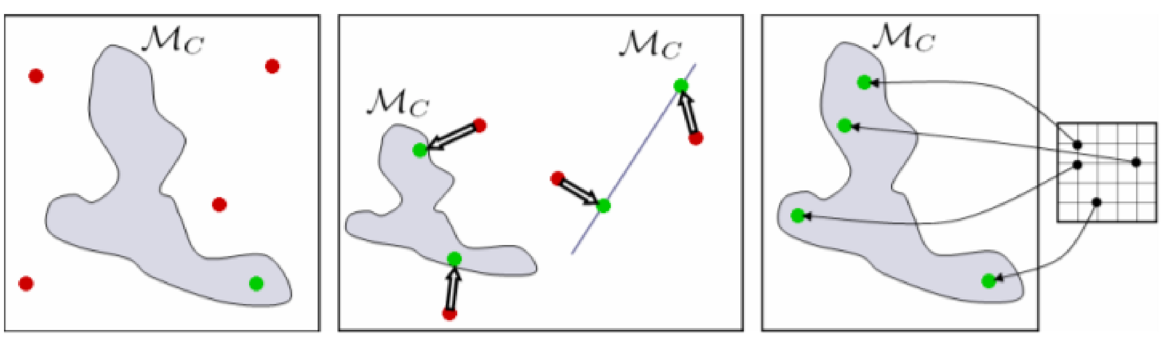


Figure 3: The three sampling strategies used in our framework. Red dots represent invalid samples and green dots represent valid ones. (Left) Rejection sampling (Center) Projection sampling (Right) Direct sampling from a parameterization of the constraint

3.1 Rejection Sampling

Approach: Sample random points and check if they are in the constraint manifold.
Typically, the constraint manifold has fewer dimensions than the configuration space, so the probability of getting a sample that is in the configuration space and satisfies the constraint manifold is exactly 0. In order to sample properly, the constraint manifold needs to occupy a significant volume of the configuration space. This approach does not work for low-dimensional constraints.

3.2 Projection Strategy

Approach: Build a tree, then project the vertices back onto the constraint manifold surface.
This approach is more robust towards more stringent constraints, such as ones whose manifolds don't occupy a significant volume of the configuration space. Because of this application, there needs to be some notion of "distance" between the sampled point to the constraint manifold. So, each Cr(q) for each sampled point, r, encodes some distance metric with a cosine function. The sampled point's $C(q) > 0$.However, at the surface of the constraint manifold, $C(q) = 0$. As such, we can use iterative gradient descent to map the sampled point back onto the manifold by following a descending slope towards the manifold surface. This strategy is used to sample on lower-dimensional constraint manifolds. Gradient Descent optimization is used in this approach to reach a point that is close to the constraint manifold.

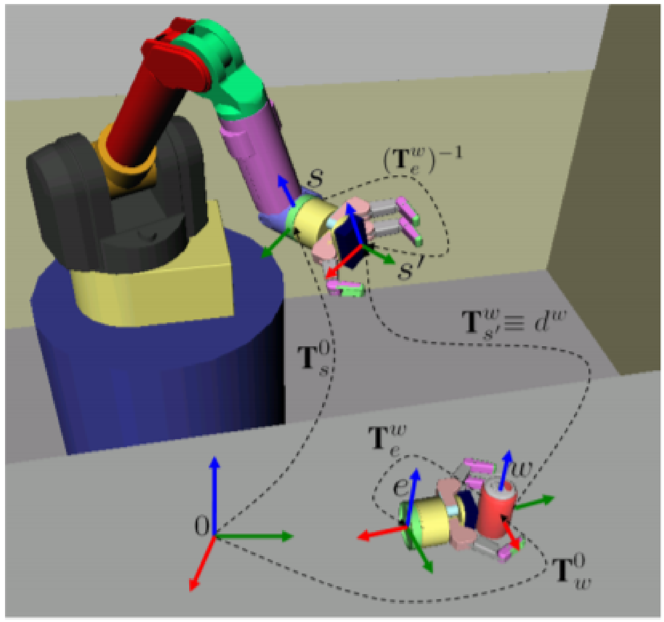
3.3 Direct Sampling

Approach: Directly represent constraint and sample points from the parameterized constraint.
This strategy is used for sampling goal configurations for our path planning algorithm.

4 Task Space Planning

4.1 Task Space Regions (TSR)

For a point in 3D space, the degrees of freedom can be defined using SE(3) i.e. Special Euclidean Group, providing 6 degrees of freedom.



T^0_w: transform from origin 0 to TSR frame w.
T^e_w: end effector offset in coordinate of w.

This allows the user to set an offset from w to the origin of the end effector e. This is useful in specifying a TSR for an object held by hand or a grasping location that is offset from e. This helps to incorporate the object properties like orientation and the surface dynamics from the sampled goal state but also properties of the end effector like wideness and orientation.

B^w: 6x2 matrix of bounds in coordinates of w.

$$B^w = \begin{bmatrix} X_{min} & X_{max} \\ Y_{min} & Y_{max} \\ Z_{min} & Z_{max} \\ \psi_{min} & \psi_{max} \\ \theta_{min} & \theta_{max} \\ \phi_{min} & \phi_{max} \end{bmatrix} \tag{7}$$

The first three rows of B^w bound the allowable translation along the x, y, and z axes (in meters) and the last three bound the allowable rotation about those axes (in radians), all in the w frame. Note that this assumes the Roll-Pitch-Yaw (RPY) Euler angle convention, which is used because it allows bounds on rotation to be intuitively specified.

T_S^0: end effector with respect to origin.

$$T_S^0 (T_e^w)^{-1} = T_{S'}^0 \tag{8}$$

Pose of the object grasped by the robot with respect to the origin.

$$T_0^w T_{S'}^0 = T_{S'}^w \tag{9}$$

Once we get, $T_{S'}^w$ We break it down in to rotational and translational part. We convert the rotational matrix into euler angles, thus we get 3 euler angles and 3 x,y,z variables (for position). These 6 variables form a vector Δx.

$$\dot{x} = J(q)\dot{q} \tag{10}$$

$$\frac{\Delta x}{\Delta t} = J(q)\frac{\Delta q}{\Delta t}$$
$$\Delta x = J(q)\Delta q$$

$$\Delta q = J\#\Delta x \tag{11}$$

$$J\# = J^T (JJ^T)^{-1} \tag{12}$$

Equation (10) is used to derive control input. Δq is change in configuration. Δx is the displacement vector. $J\#$ is the Jacobian pseudo-inverse, which must be computed at every time step due to the Jacobian changing as the configuration changes.

5 References

- Task Space Regions: A Framework for Pose-Constrained Manipulation Planning
  https://www.ri.cmu.edu/pub_files/2011/10/dmitry_ijrr10-1.pdf
- https://en.wikipedia.org/wiki/Rejection_sampling