

Task Space Regions

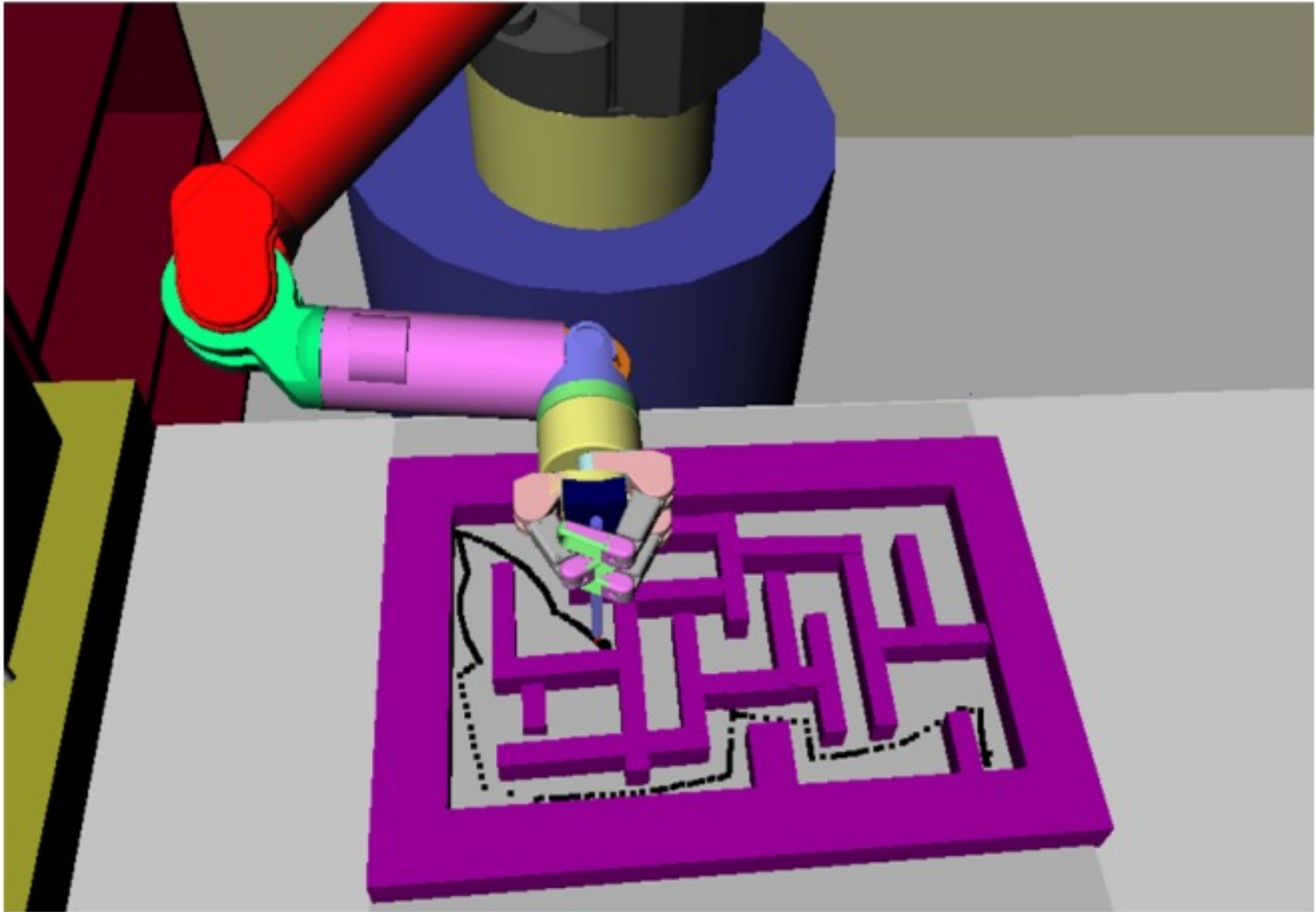
CSCI 545 Introduction to Robotics
Instructor: Stefanos Nikolaidis

Transformation Properties Recap

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

$$T_i^{i-1} = \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix}$$

How can we represent constraints?



How can we represent constraints?



How can we represent constraints?



Representing Constraints

- We let $C(q)$ a constraint evaluation function
- A constraint is defined as a pair $\{C(q), s\}$, where $C(q) \in \mathbb{R} \geq 0$
- $C(q)$ determines whether a constraint is met at a given q and $s \in [0, 1]$
- A constraint is satisfied when $C(q) = 0$

Representing Constraints

- A manifold of configurations that meet this constraint

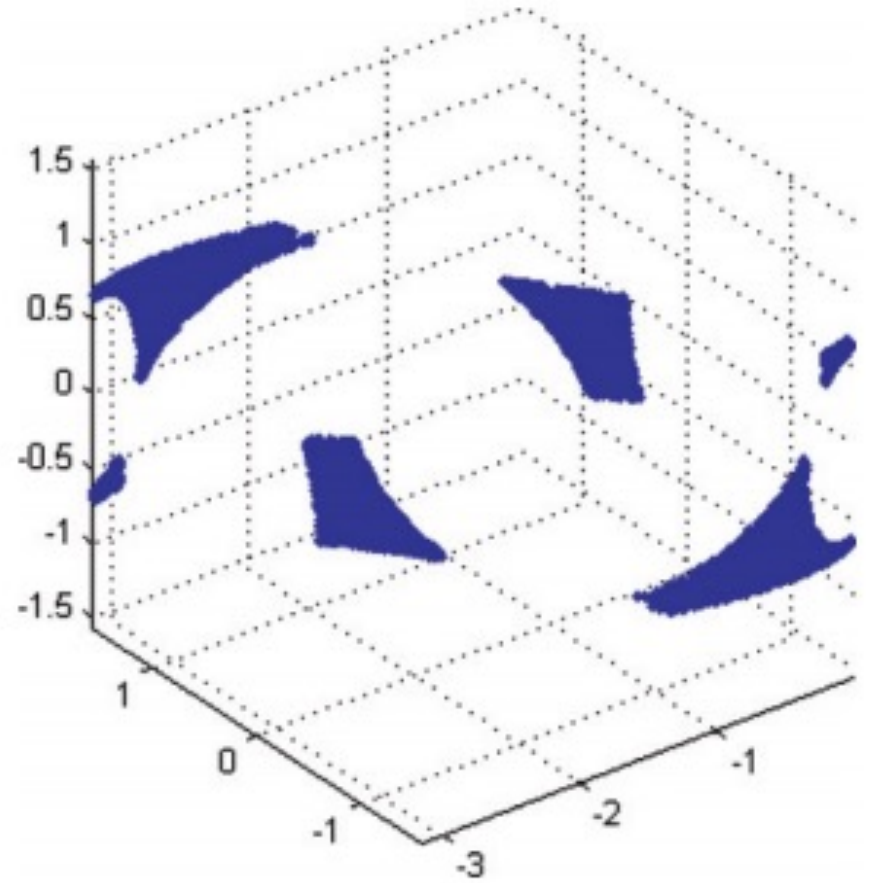
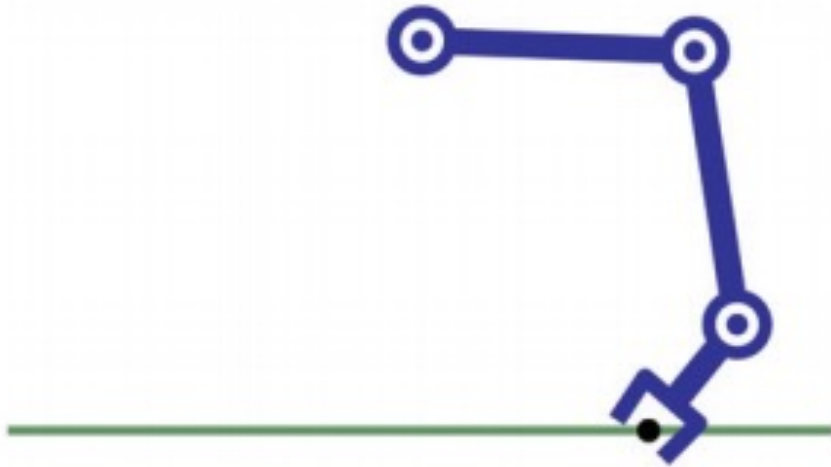
$$M_c : \{q \in Q : C(q) = 0\}$$

- We can have many constraints $M_{c_i} : \{q \in Q : C_i(q) = 0\}$

- Constrained Motion Planning Problem:

$$\tau : q \in M_{c_i} \forall q \in \tau(s), \forall s \in [0, 1], i \in \{1, \dots, n\}$$

Example



Direct Sampling

- If there exists a parameterization of the constraint, we can directly generate samples

- Used to generate samples in a desired goal region

Direct Sampling

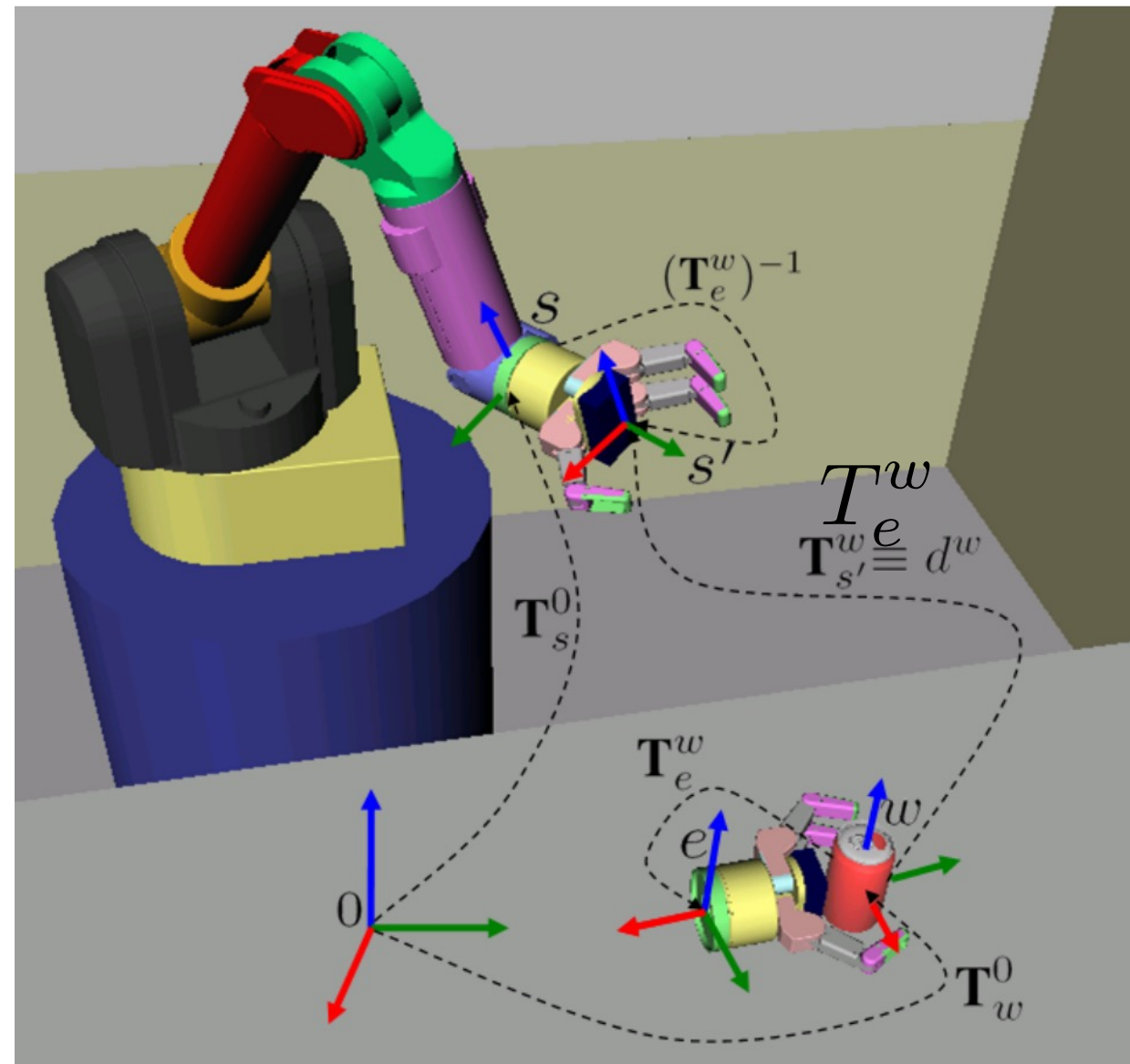
- If there exists a parameterization of the constraint, we can directly generate samples

- Used to generate samples in a desired goal region

Task Space Regions

- TSRs describe end-effector constraints as subsets of $SE(3)$
- TSRs are specifically designed to be used with sampling-based planners
- Applications:
 - How to grasp an object
 - How to manipulate an object with constraints

Task Space Regions



T_w^0 Transform from the origin to the TSR frame w

T_e^w End-effector offset in the coordinates of w

B^w 6 x 2 matrix of bounds in coordinates of w

End-Effector Offsets

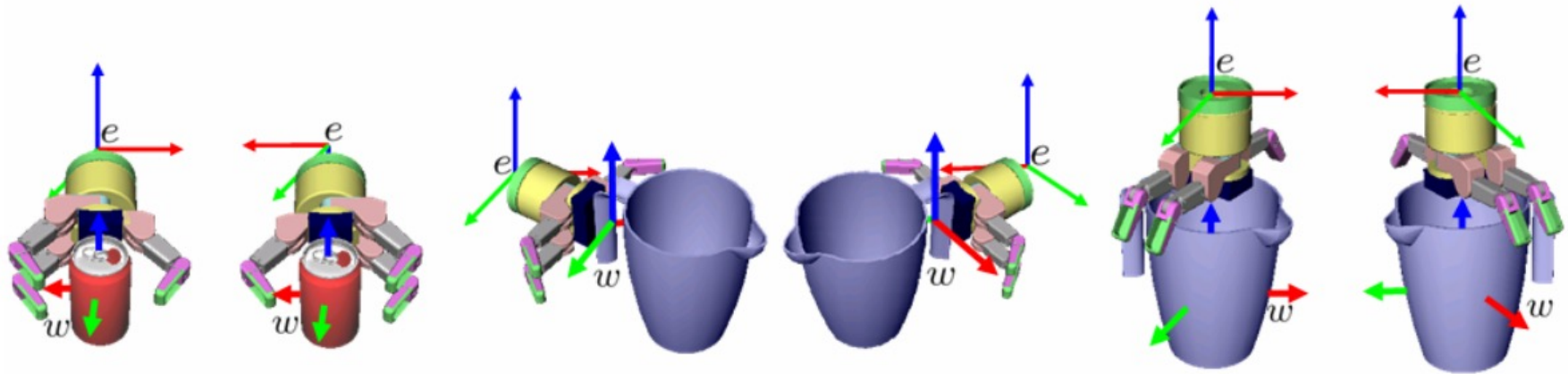
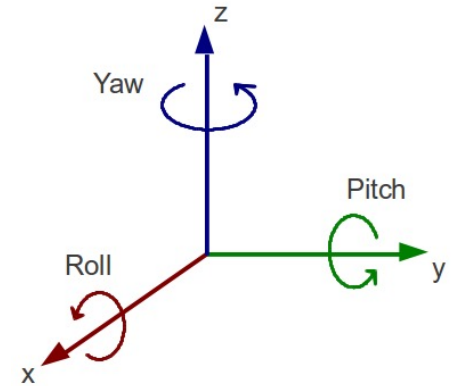


Figure 4: The w and e frames used to define end-effector goal TSRs for a soda can and a pitcher.

Matrix of Bounds

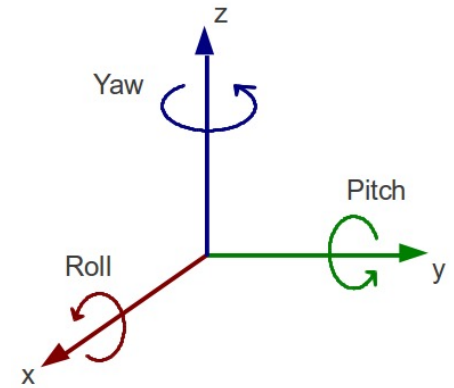
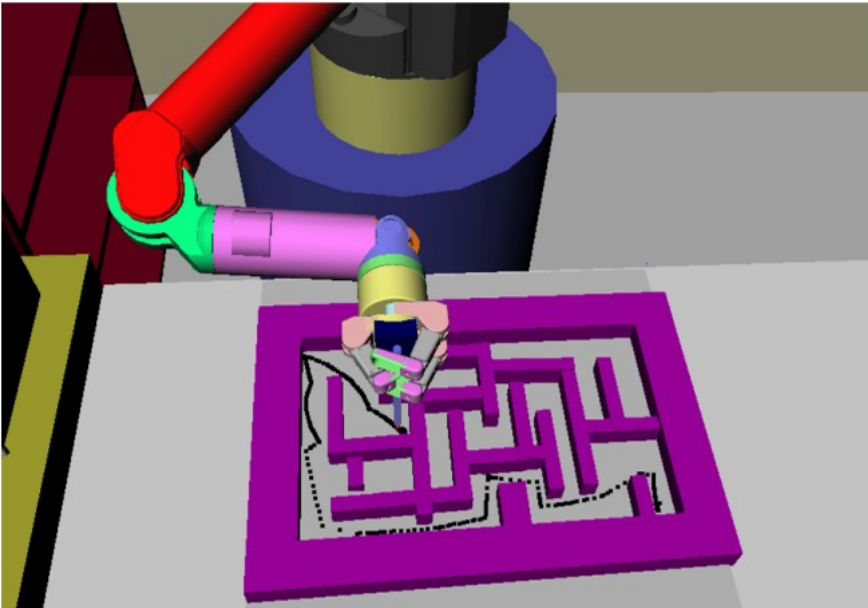


$$\mathbf{B}^w = \begin{bmatrix} x_{min} & x_{max} \\ y_{min} & y_{max} \\ z_{min} & z_{max} \\ \psi_{min} & \psi_{max} \\ \theta_{min} & \theta_{max} \\ \phi_{min} & \phi_{max} \end{bmatrix}$$

The first 3 rows bound the allowable translation along the x , y , and z axes (in meters) and the last three (Roll-Pitch-Yaw) bound the allowable rotation in those axes

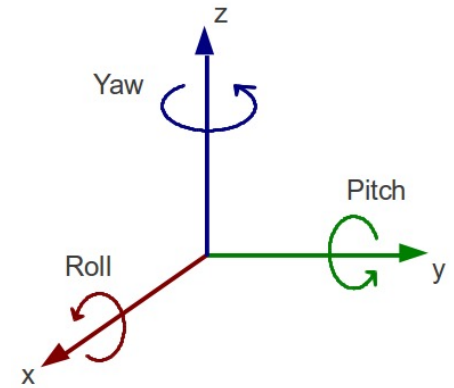
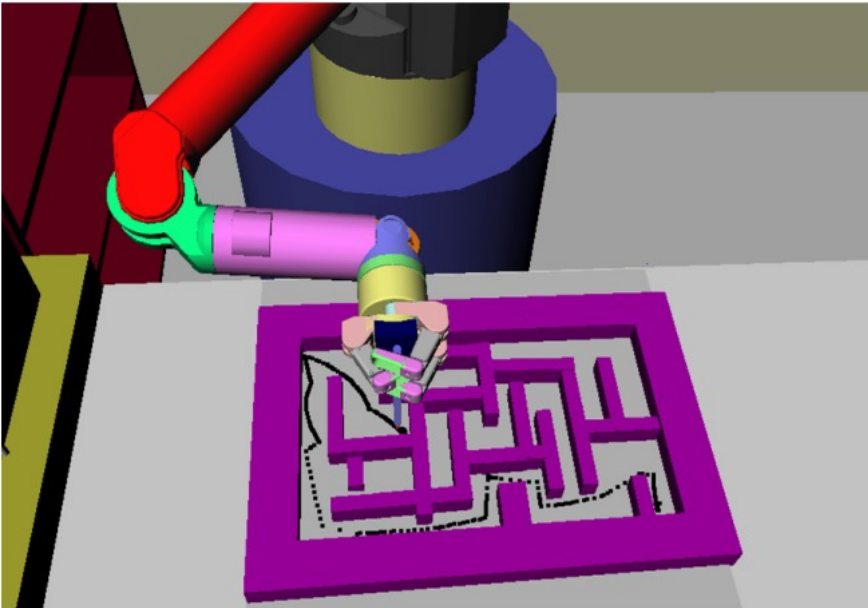
Matrix of Bounds Examples: Maze

“The robot’s end-effector should always touch the table”



Matrix of Bounds Examples: Maze

“The robot’s end-effector should always touch the table”



$$B^w = \begin{bmatrix} -\infty & \infty \\ -\infty & \infty \\ 0 & 0 \\ -\alpha & \alpha \\ -\alpha & \alpha \\ -\pi & \pi \end{bmatrix}$$

Matrix of Bounds Examples

YCB Task: HERB Stacks Cups

Vinitha Ranganeni
Jennifer King
Rachel Holladay
Siddhartha Srinivasa

The Robotics Institute
Carnegie Mellon University

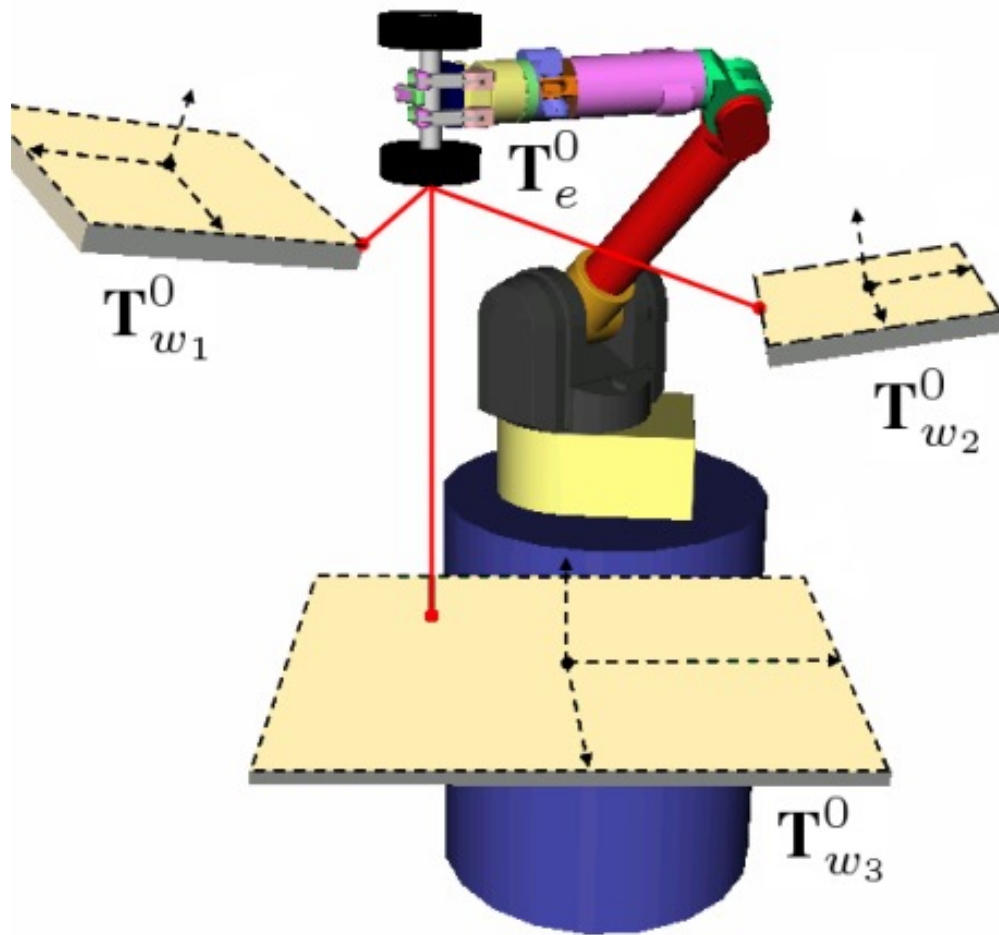


Matrix of Bounds Examples: Cup



$$B^w = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ -\alpha & \alpha \\ 0 & 0 \\ 0 & 0 \\ -\pi & \pi \end{bmatrix}$$

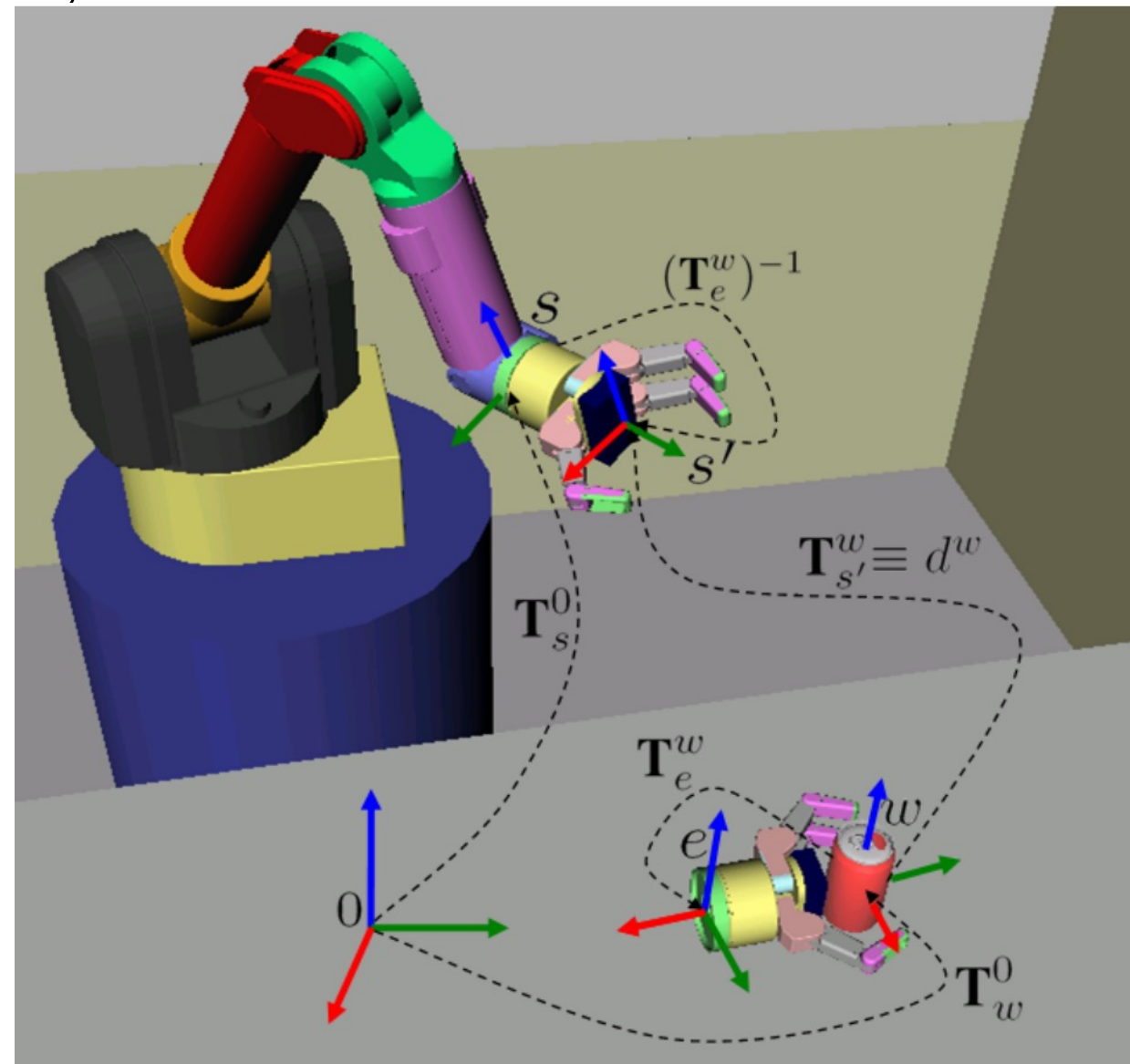
Matrix of Bounds Examples: Sliding Surfaces



$$B^w = \begin{bmatrix} -\frac{l}{2} & \frac{l}{2} \\ -\frac{w}{2} & -\frac{w}{2} \\ \alpha & \alpha \\ 0 & 0 \\ 0 & 0 \\ -\pi & \pi \end{bmatrix}$$

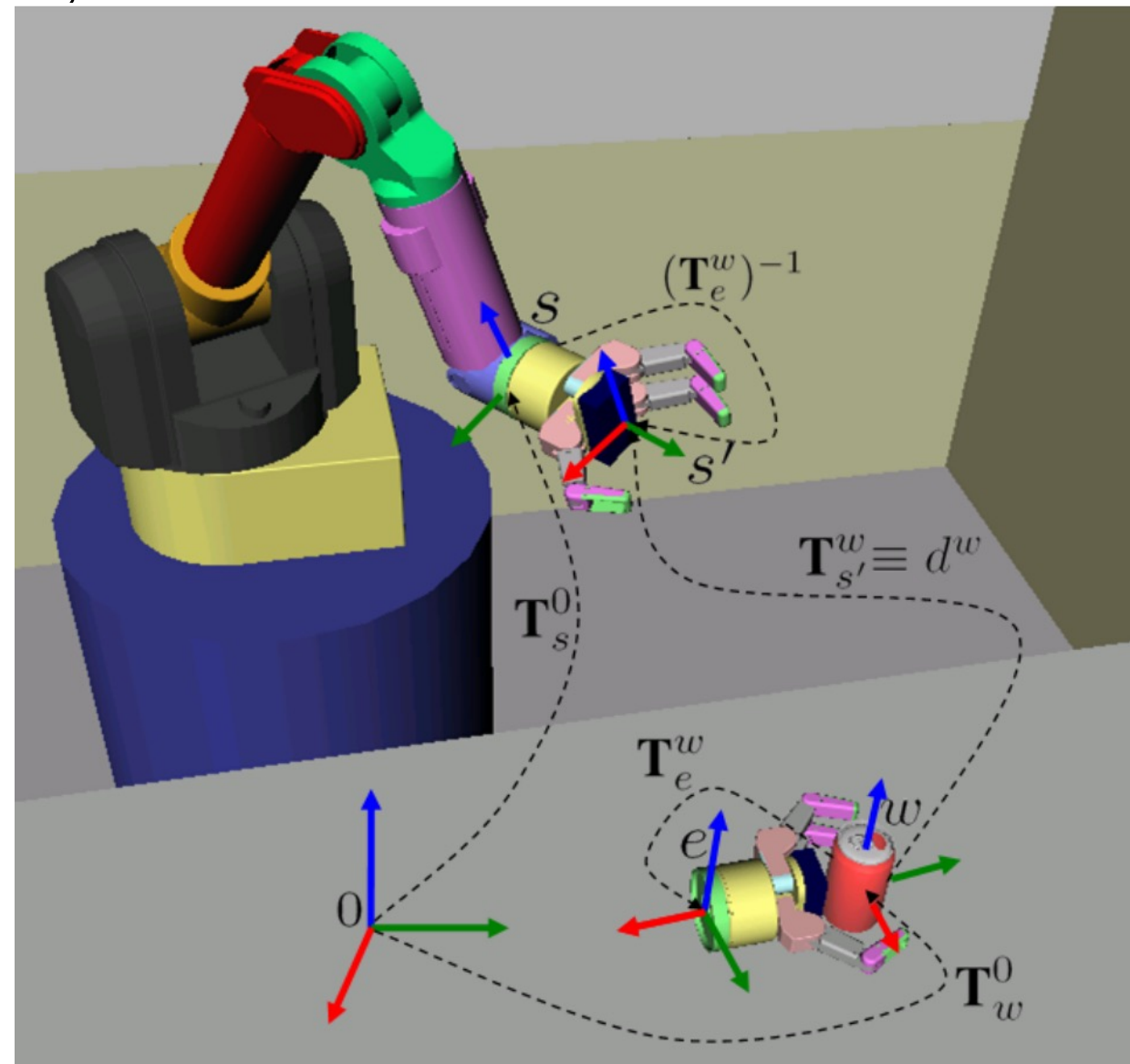
Transformation to World Coordinate System

1. Sample s from B^w
2. Convert 6D vector to transformation T_s^w in SE(3)

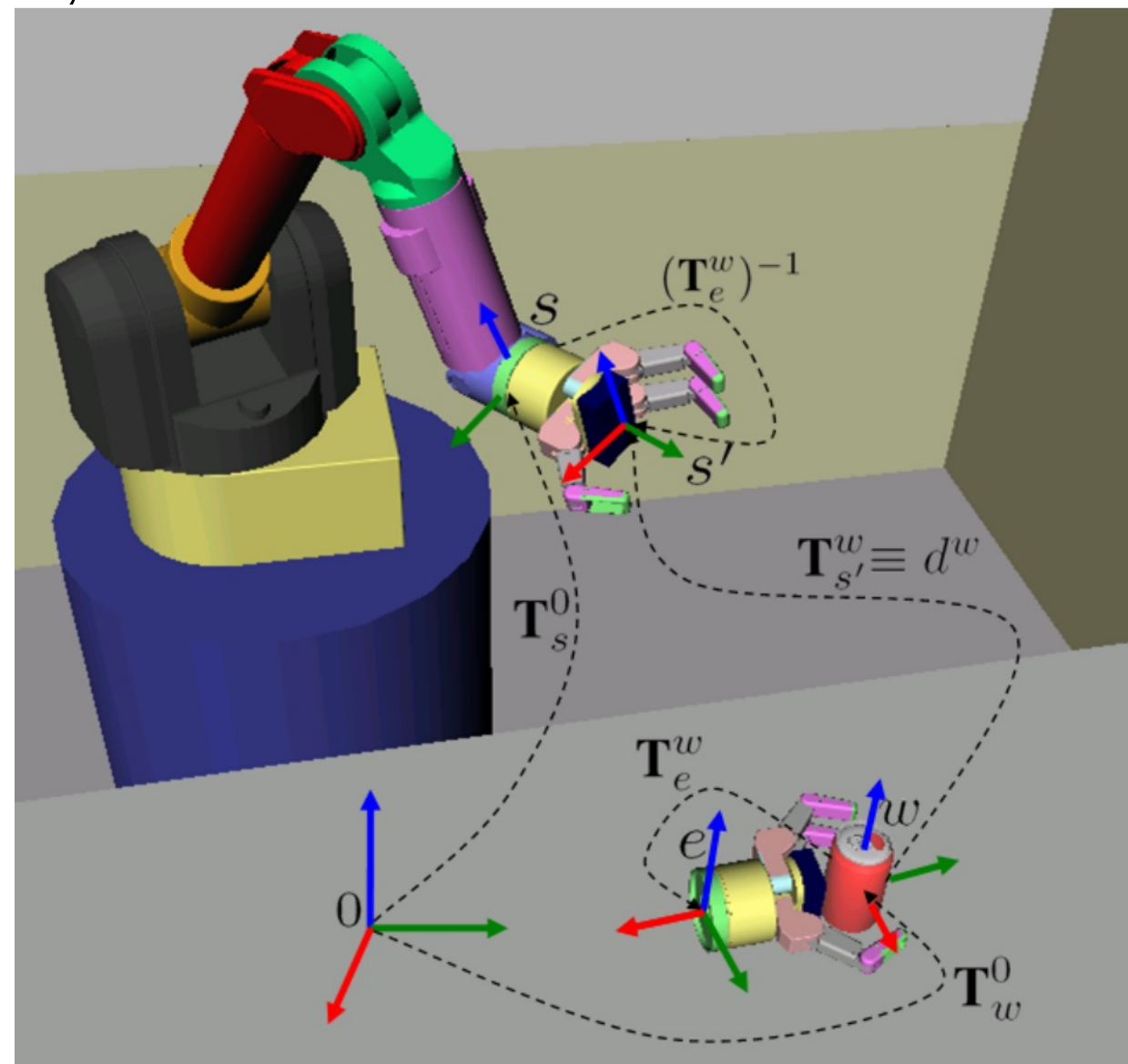


Transformation to World Coordinate System

1. Sample s from B^w
2. Convert 6D vector to transformation T_s^w in SE(3)
3. Convert from object to world coordinate systems



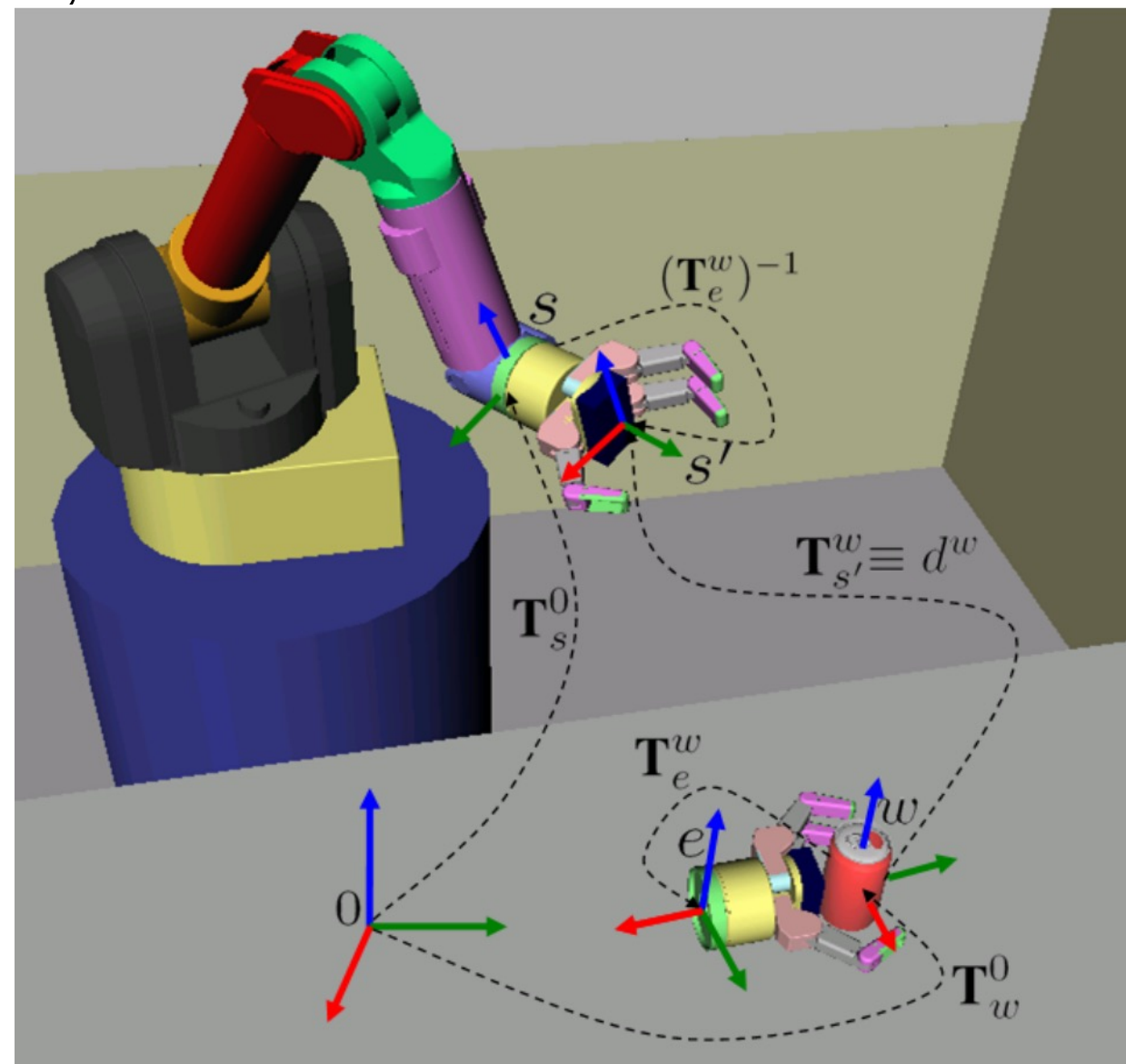
Transformation to World Coordinate System



1. Sample s from B^w
2. Convert 6D vector to transformation T_s^w in SE(3)
3. Convert from object to world coordinate systems

$$T_s^0 = T_w^0 T_s^w$$

Transformation to World Coordinate System

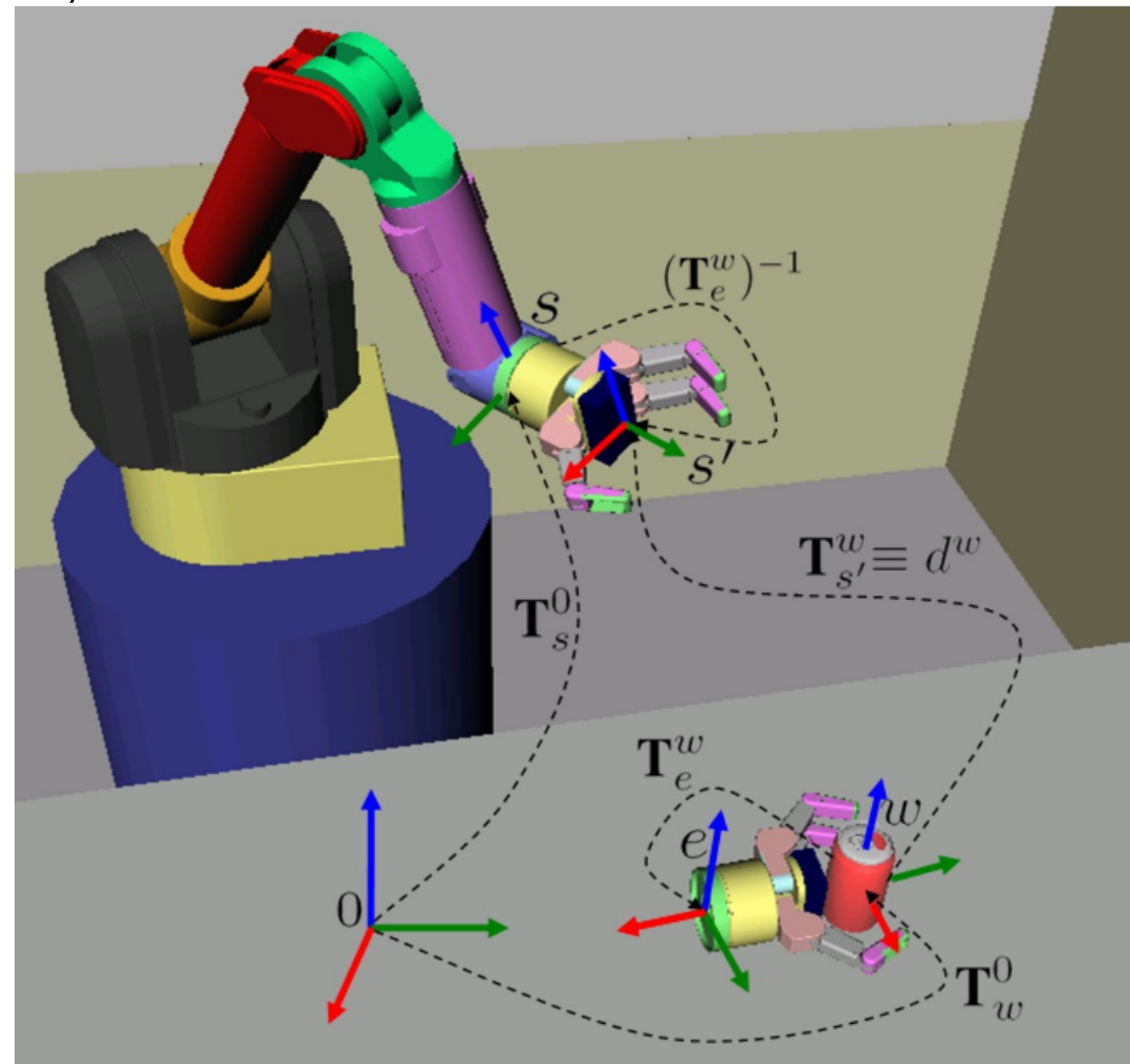


1. Sample s from B^w
2. Convert 6D vector to transformation T_s^w in SE(3)
3. Convert from object to world coordinate systems

$$T_s^0 = T_w^0 T_s^w$$

4. Apply end-effector offset transform T_e^w

Transformation to World Coordinate System



$$T_{s'}^0 = T_w^0 T_s^w T_e^w$$

↓ IK

qG

Building Constrained Paths

Rejection Sampling

- Generate a sample $q \in Q$
- Check if $C(q) = 0$
- If not, deem q invalid

Building Constrained Paths

Rejection Sampling

- Generate a sample $q \in Q$
- Check if $C(q) = 0$
- If not, deem q invalid

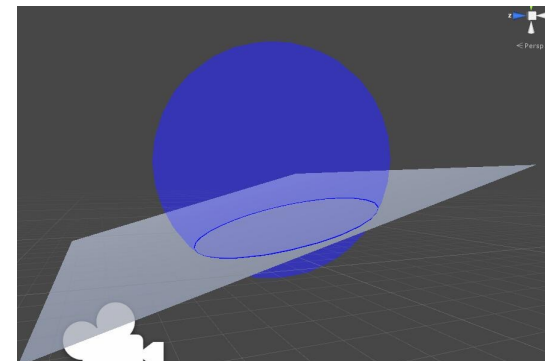
What are limitations of this strategy?

Building Constrained Paths

Rejection Sampling

- Generate a sample $q \in Q$
- Check if $C(q) = 0$
- If not, deem q invalid

If constraint manifold is of lower dimension than Q , the probability of sampling from the manifold is 0!



Building Constrained Paths

Projection Strategy

- Generate a sample $q \in Q$
- Project q to the constraint manifold
- Need to specify $C(q)$ as some type of distance to the manifold

Building Constrained Paths

Projection Strategy

- Generate a sample $q \in Q$
- Project q to the constraint manifold
- Need to specify $C(q)$ as some type of distance to the manifold

How can we project to the manifold?

Building Constrained Paths

Projection Strategy

- Generate a sample $q \in Q$
- Project q to the constraint manifold
- Need to specify $C(q)$ as some type of distance to the manifold

Gradient Descent!

Building Constrained Paths

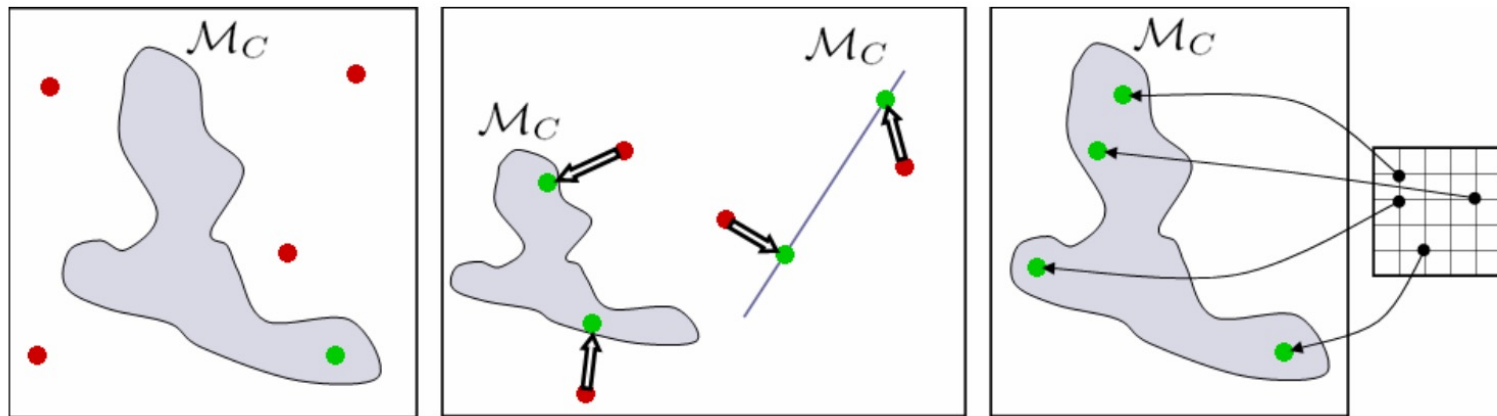
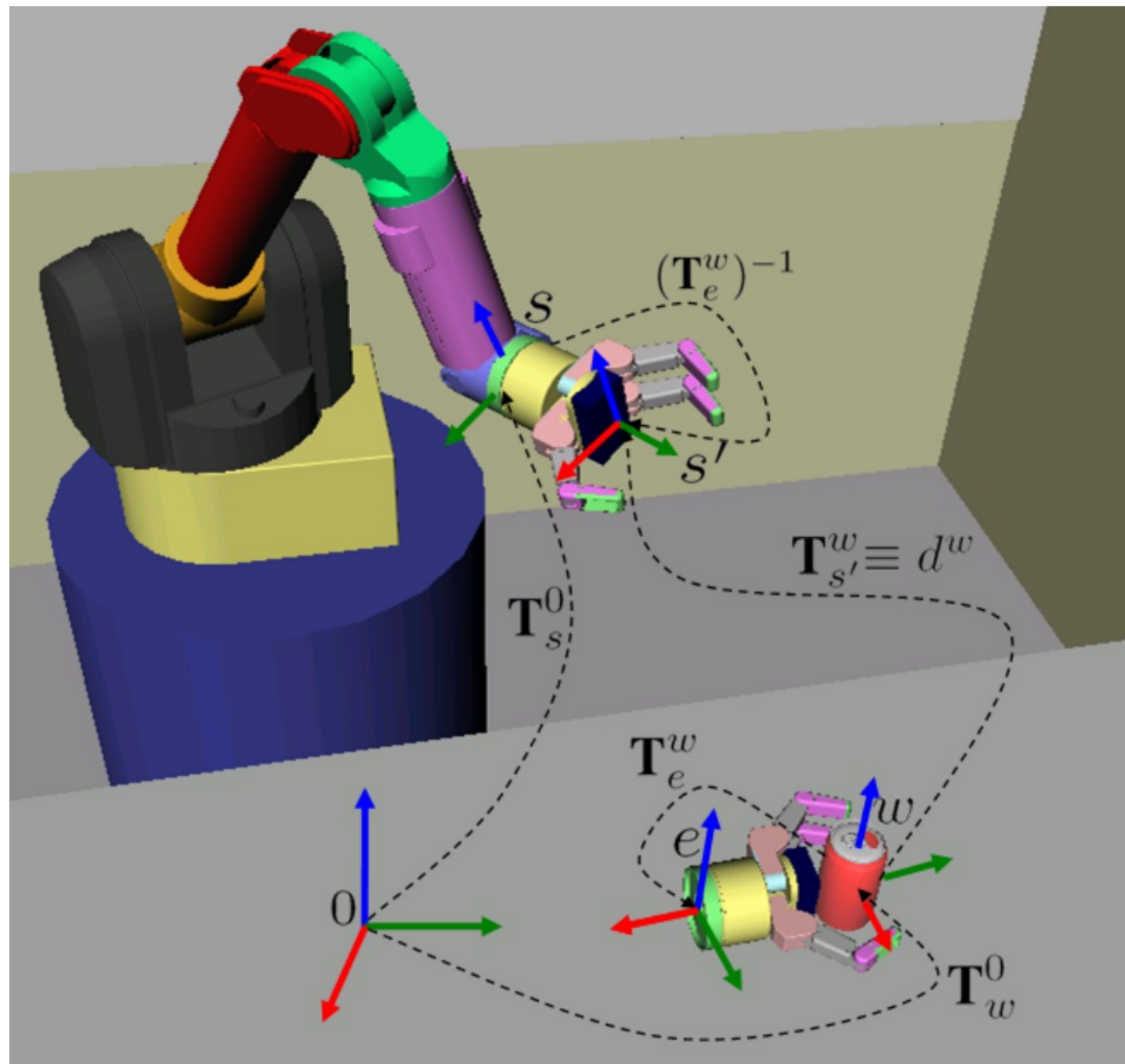


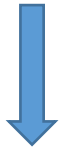
Figure 3: The three sampling strategies used in our framework. Red dots represent invalid samples and green dots represent valid ones. (Left) Rejection sampling (Center) Projection sampling (Right) Direct sampling from a parameterization of the constraint

Transformation to World Coordinate System



Projection Strategy

$$T_{s'}^w = (T_w^0)^{-1} T_s^0 (T_e^w)^{-1}$$



d^w

6D vector (conversion
from rotation matrix to
Euler Angles)

Projection Strategy

$$T_{s'}^w = (T_w^0)^{-1} T_s^0 (T_e^w)^{-1}$$

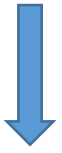


d^w 6D vector (conversion
from rotation matrix to
Euler Angles)

$$d^w = \begin{bmatrix} \mathbf{t}_{s'}^w \\ \arctan 2(\mathbf{R}_{s'_{32}}^w, \mathbf{R}_{s'_{33}}^w) \\ - \arcsin(\mathbf{R}_{s'_{31}}^w) \\ \arctan 2(\mathbf{R}_{s'_{21}}^w, \mathbf{R}_{s'_{11}}^w) \end{bmatrix}$$

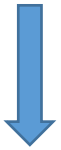
Projection Strategy

$$T_{s'}^0 = T_w^0 T_s^w T_e^w$$



6D vector (conversion
from rotation matrix to
Euler Angles)

$$d^w$$



$$\Delta x$$

$$\Delta \mathbf{x}_i = \begin{cases} d_i^w - \mathbf{B}_{i,1}^w & \text{if } d_i^w < \mathbf{B}_{i,1}^w \\ d_i^w - \mathbf{B}_{i,2}^w & \text{if } d_i^w > \mathbf{B}_{i,2}^w \\ 0 & \text{otherwise} \end{cases}$$

Projection Strategy

Algorithm 1: \mathbf{J}^+ Projection(q)

```
1 while true do
2    $\Delta \mathbf{x} \leftarrow \text{DisplacementFromTSR}(q)$ ;
3   if  $\|\Delta \mathbf{x}\| < \epsilon$  then
4     return  $q$ ;
5   end
6    $\mathbf{J} \leftarrow \text{GetJacobian}(q)$ ;
7    $\Delta q_{error} \leftarrow \mathbf{J}^T (\mathbf{J}\mathbf{J}^T)^{-1} \Delta \mathbf{x}$ ;
8    $q \leftarrow (q - \Delta q_{error})$ ;
9 end
```

Planning with TSRs as Goal Sets

- Motion Planning Problem

$$\tau : q \in M_{c_i} \forall q \in \tau(s), \forall s \in [0, 1], i \in \{1, \dots, n\}$$

Planning with TSRs as Goal Sets

- Motion Planning Problem

$$\tau : q \in M_{c_i} \forall q \in \tau(s), \forall s \in [0, 1], i \in \{1, \dots, n\}$$

- We specify: $\{C(q) = \text{DistanceToTSR}(q), \quad s = [1]\}$

Planning with TSRs as Goal Sets

1. Sample directly the TSR
2. Pass directly the sampled pose to an IK solver
3. Rank IK solutions, e.g., based on proximity to the robot
4. Run RRT with small goal precision for highest-ranked goal solution
5. If fail, go to step 4 with next candidate goal solution

Planning with TSRs as Pose Constraints

Planning with TSRs as Pose Constraints

1. Run standard RRT (or variant)
2. When ``steering'' towards the newly sampled point, project to constrained manifold

Constrained Bi-Directional RRT

Algorithm 2: CBiRRT2(Q_s, Q_g)

```
1  $T_a$ .Init( $Q_s$ );  $T_b$ .Init( $Q_g$ );
2 while TimeRemaining() do
3    $T_{goal} = \text{GetBackwardTree}(T_a, T_b)$ ;
4   if size( $T_{goal}$ ) = 0 or rand(0, 1) <  $P_{sample}$  then
5     AddRoot( $T_{goal}$ );
6   else
7      $q_{rand} \leftarrow \text{RandomConfig}()$ ;
8      $q_{near}^a \leftarrow \text{NearestNeighbor}(T_a, q_{rand})$ ;
9      $q_{reach}^a \leftarrow \text{ConstrainedExtend}(T_a, q_{near}^a, q_{rand})$ ;
10     $q_{near}^b \leftarrow \text{NearestNeighbor}(T_b, q_{reached}^a)$ ;
11     $q_{reach}^b \leftarrow \text{ConstrainedExtend}(T_b, q_{near}^b, q_{reach}^a)$ ;
12    if  $q_{reach}^a = q_{reach}^b$  then
13       $P \leftarrow \text{ExtractPath}(T_a, q_{reach}^a, T_b, q_{reach}^b)$ ;
14      return ShortenPath( $P$ );
15    else
16      Swap( $T_a, T_b$ );
17    end
18  end
19 end
20 return  $\emptyset$ ;
```

Constrained Bi-Directional RRT

