THE OHIO STATE UNIVERSITY

ME7752 Final Project

Autumn 2021

Inverse Kinematics and Trajectory Generation of Block "O" with dVRK Robot

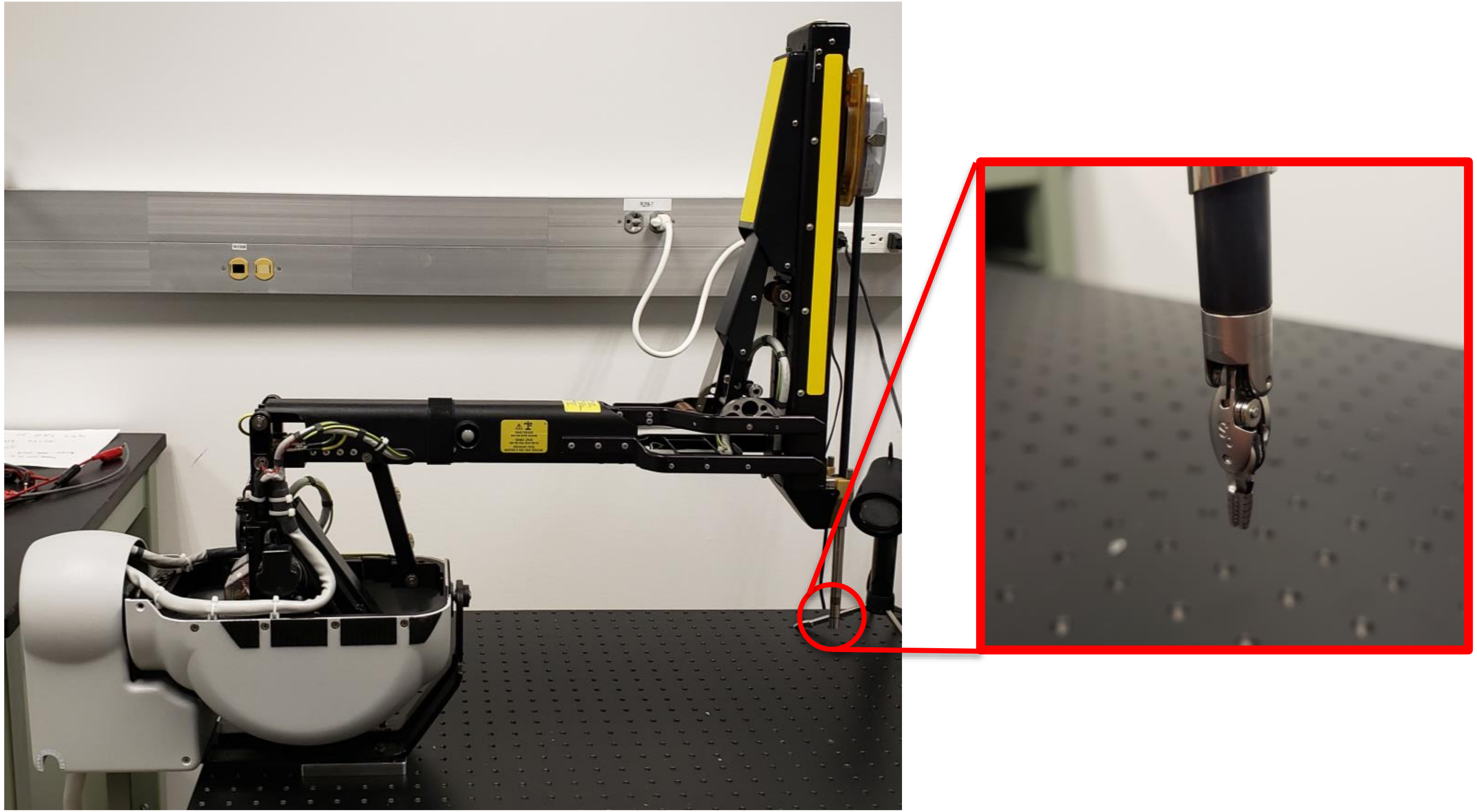
Overview

- Introduction
- Forward and Inverse Kinematics Using PoE
- Trajectory Generation of the Block "O"
- Software & Hardware
- Results & Conclusions

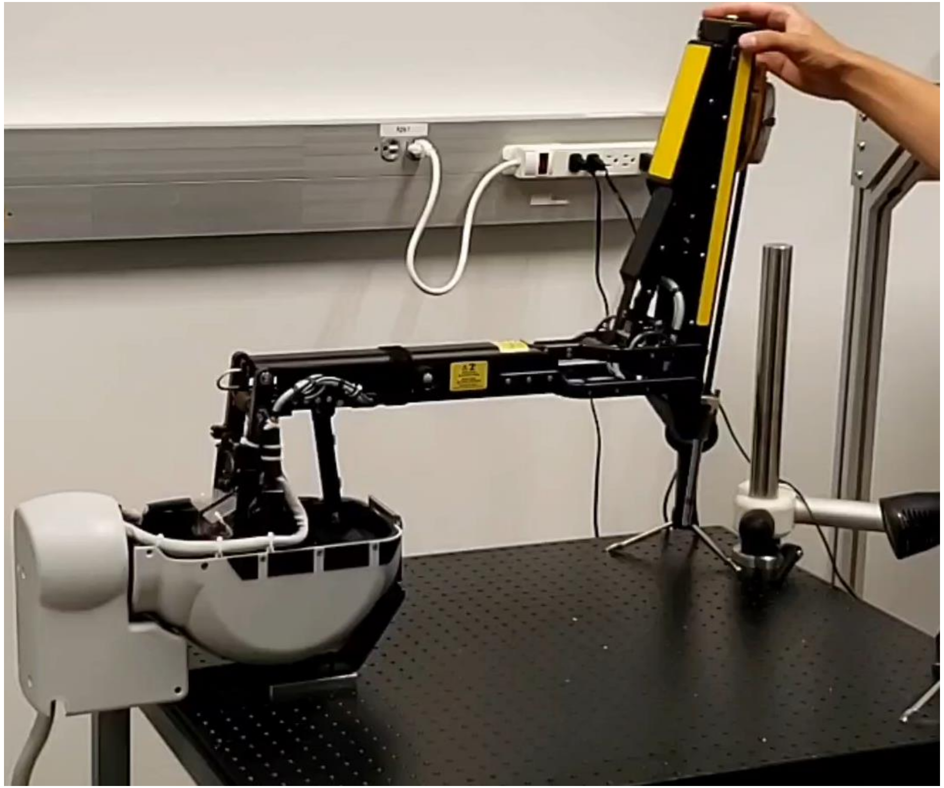
daVinci Research Kit (dVRK)

- Intuitive Surgical, Inc.
- **PSM** – “Patient Side Manipulator”
- **MTM** – “Master Tool Manipulator”

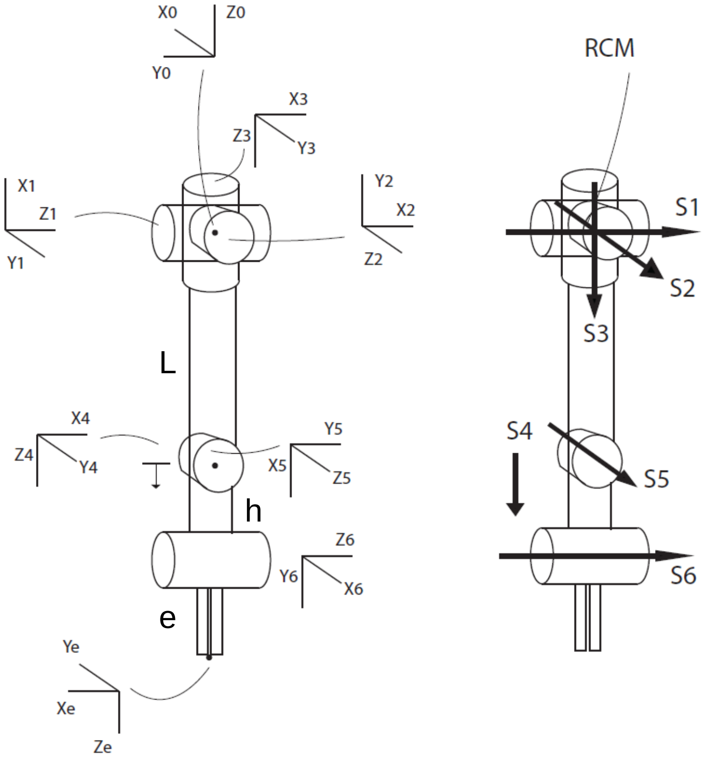
PSM Robot



PSM Movement



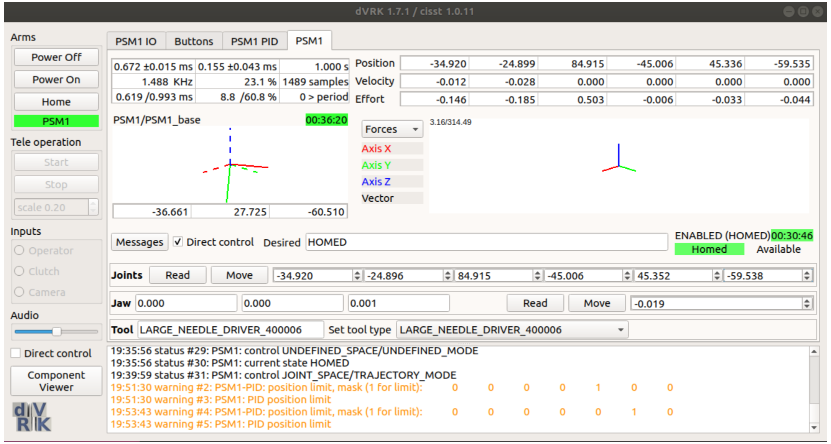
Forward Kinematics

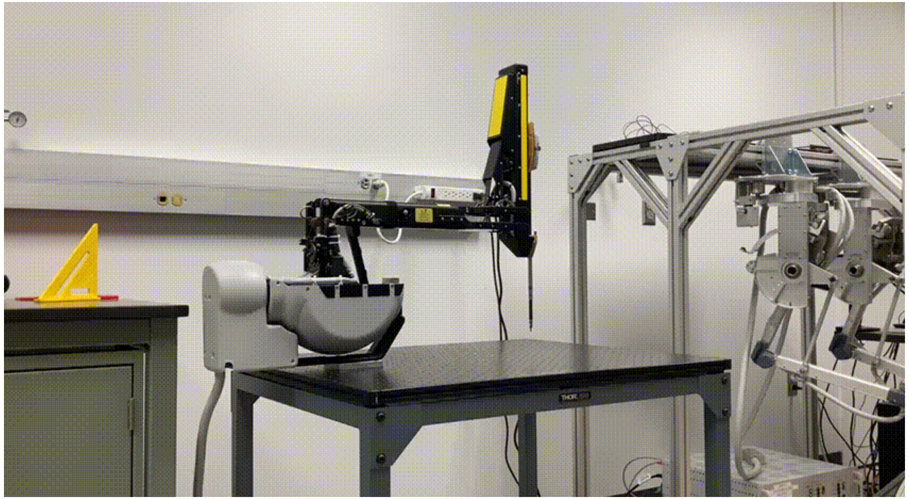


S_i	ω_{si}	v_{si}
1	(0 -1 0)	(0 0 0)
2	(-1 0 0)	(0 0 0)
3	(0 0 -1)	(0 0 0)
4	(0 0 0)	(0 0 -1)
5	(-1 0 0)	(0 L 0)
6	(0 -1 0)	(-(h+L) 0 0)

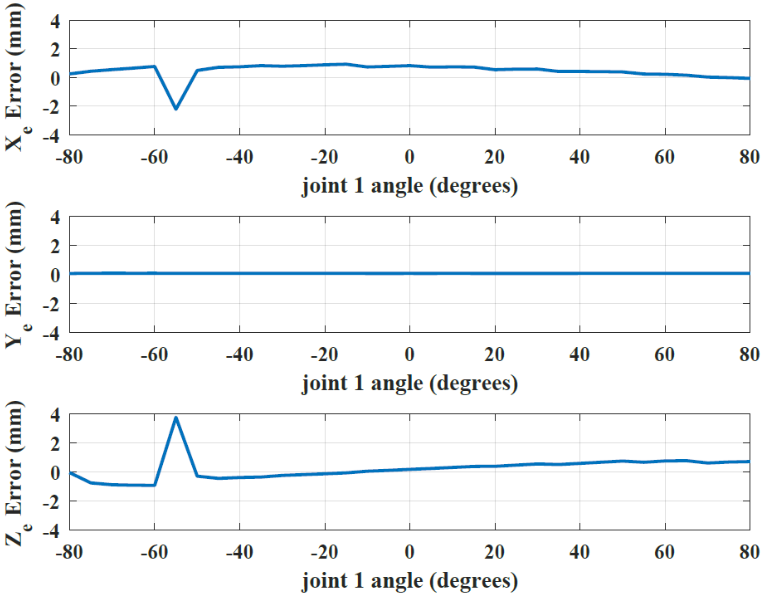
$$T_{0e} = e^{[S_1]\theta_1} e^{[S_2]\theta_2} \ldots e^{[S_6]\theta_6} M$$

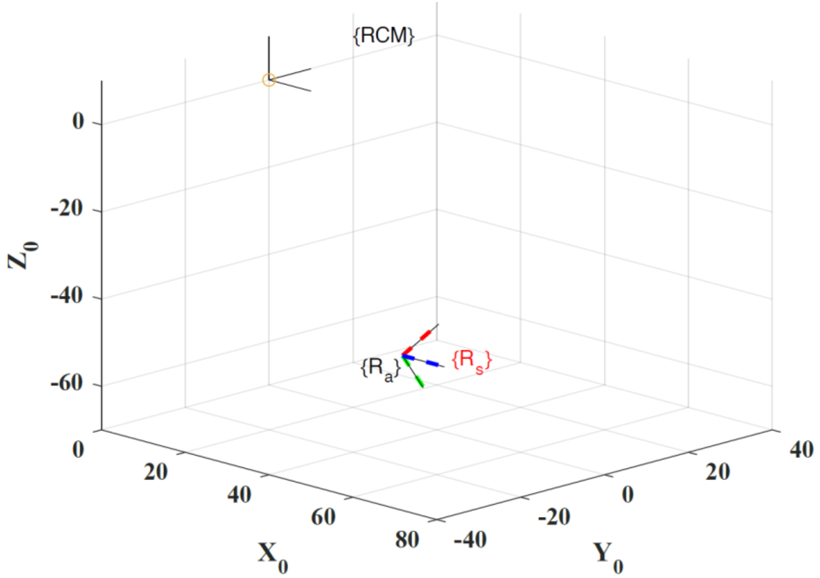
FK Validation Methods





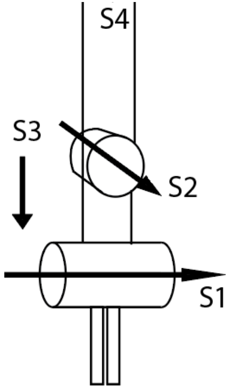
FK Validation Results





Closed-Form Inverse Kinematics

$$\theta_1 = atan2(px, pz - e)$$
$$\theta_2 = atan2(-py, px * s1 + pz * c1 - c1 * e - h)$$
$$\theta_3 = -L - h * c2 - py * s2 + (pz - e) * c1 * c2 + px * c2 * s1$$
$$\theta_4 = atan2(-Rot(1,2) * c1 + Rot(3,2) * s1, Rot(2,2) * c2 + Rot(3,2) * c1 * s2 + Rot(1,2) * s1 * s2)$$
$$\theta_5 = atan2(Rot(3,1) * (s1 * s4 + c1 * c4 * s2) - Rot(1,1) * (c1 * s4 - c4 * s1 * s2) + Rot(2,1) * c2 * c4, \\ -Rot(3,1) * c1 * c2 + Rot(2,1) * s2 - Rot(1,1) * c2 * s1)$$
$$\theta_6 = atan2(-Rot(3,1) * c1 * c2 + Rot(2,1) * s2 - Rot(1,1) * s1 * c2, Rot(3,3) * c1 * c2 - Rot(2,3) * s2 + Rot(1,3) * c2 * s1)$$



$$\theta_{IK} = [20, \quad 22, \quad 1.5, \quad 15, \quad -150, \quad 25.71]$$

$$\theta_{actual} = [20, \quad 22, \quad 1.5, \quad 15, \quad 30, \quad 25.71]$$

Virtual Base Frame Here

Path Generation

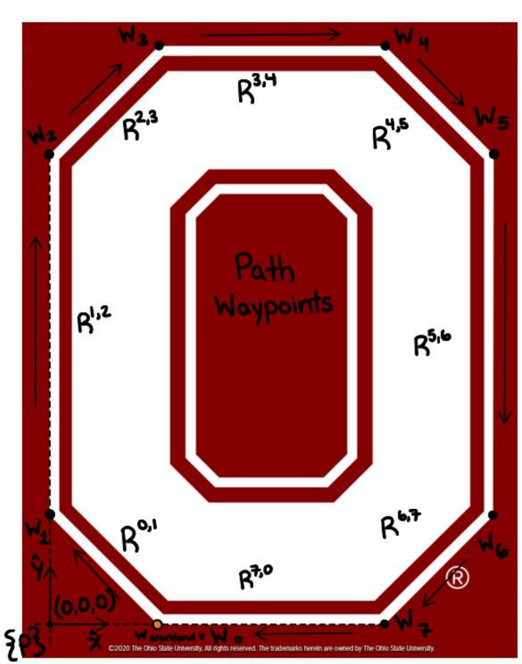
Block "O" Coordinates

- 8 positional waypoints

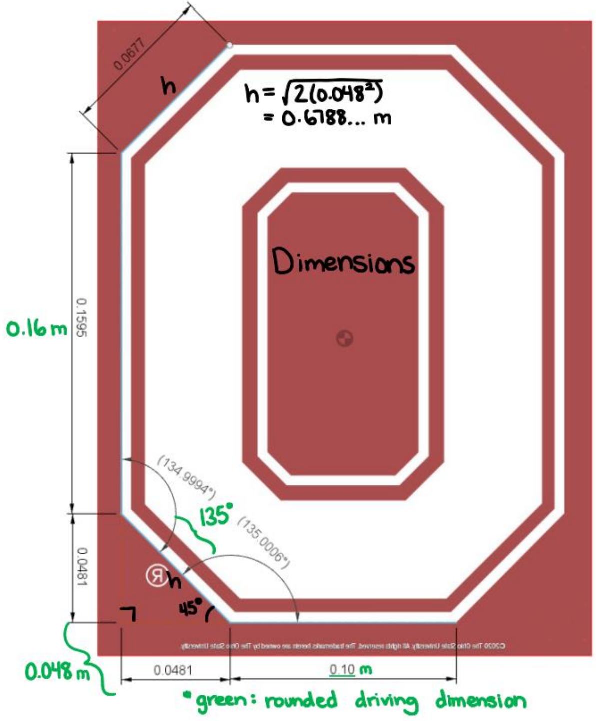
Block "O" Modulation

- **c** – scaling factor
- $\mathbf{R_p}$ – reorientation
- $\mathbf{(x_f,y_f,z_f)}$ – origin offset
- $\mathbf{\{W_m\}}$ – set of new waypoints

$\{W_m\} = R_p[c*\{w_0,w_1,\ldots,w_7\}]+(x_f,y_f,z_f)$



W_i: Waypoint$_i$, {p}: path plane orientation
$R^{i,j}$: orientation of EE on path from $W_i \rightarrow W_j$

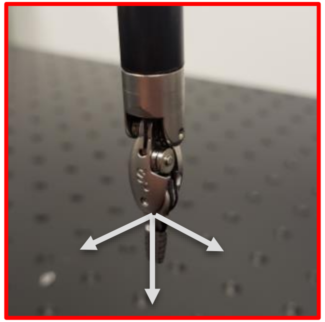


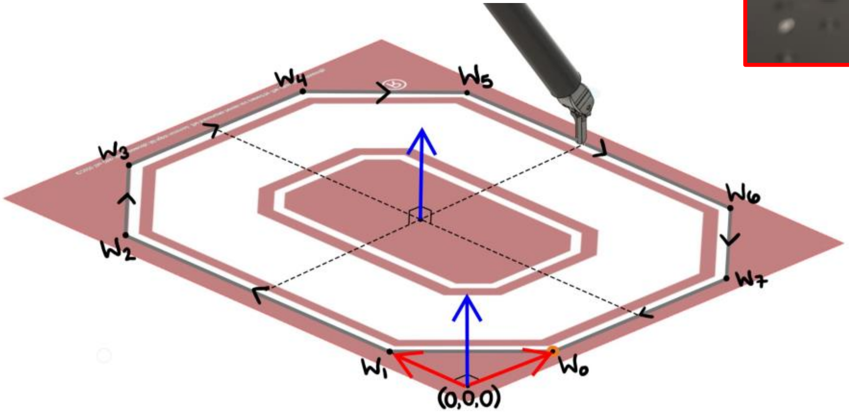
W_i	W_0	W_1	W_2	W_3	W_4	W_5	W_6	W_7
(x,y,z) [m]	(0.048, 0,0)	(0,0.048 ,0)	(0, 0.208,0)	(0.048, 0.256,0)	(0.148, 0.256, 0)	(0.196, 0.208,0)	(0.196, 0.048,0)	(0.148,0,0)

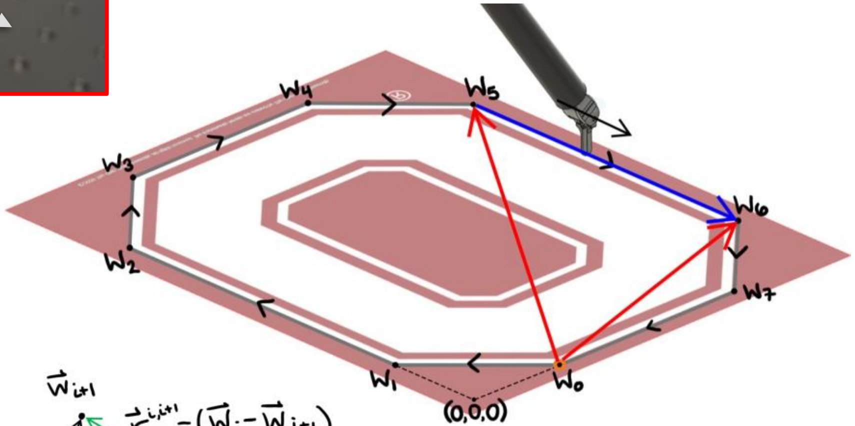
Path Generation

Waypoint Orientations

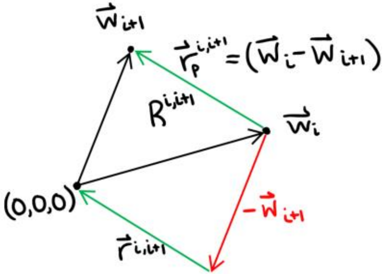
- Normal to plane (constant)
- Parallel to velocity







$$\hat{r}_n = \frac{\vec{W}_0 \times \vec{W}_1}{\|\vec{W}_0 \times \vec{W}_1\|} = \frac{\vec{r}^{\,n}}{\|\vec{r}^{\,n}\|}, \text{ where } \vec{r}^{\,n} = \vec{W}_0 \times \vec{W}_1$$



$$\hat{r}_p^{\,i,i+1} = \frac{(\vec{W}_i - \vec{W}_{i+1})}{\|(\vec{W}_i - \vec{W}_{i+1})\|} = \frac{\vec{r}^{\,i,i+1}}{\|\vec{r}^{\,i,i+1}\|}$$

Path Generation

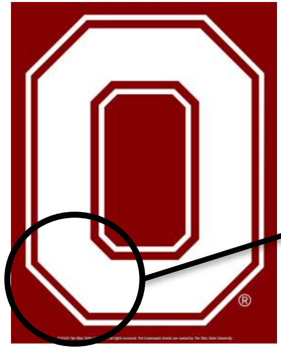
List of Waypoints

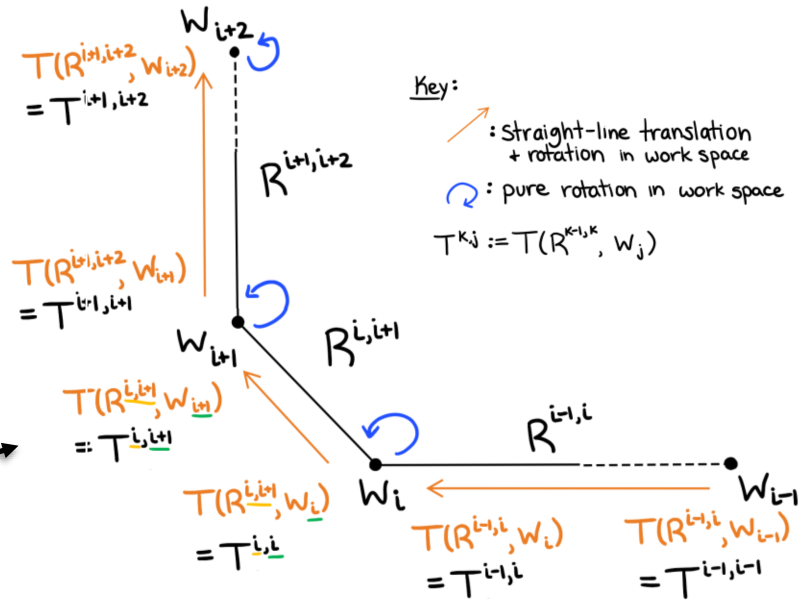
$$T^{K,j} := T(R^{K,K+1}, W_j)$$

- where path is ordered list of transformations

Block "O" Path $:= [T^{0,0}, T^{0,1}, T^{1,1}, T^{1,2}, \dots T^{7,7}, T^{7,8}]$, where

$R^{7,8} = R^{7,0}$
$+\ W_8 = W_0$





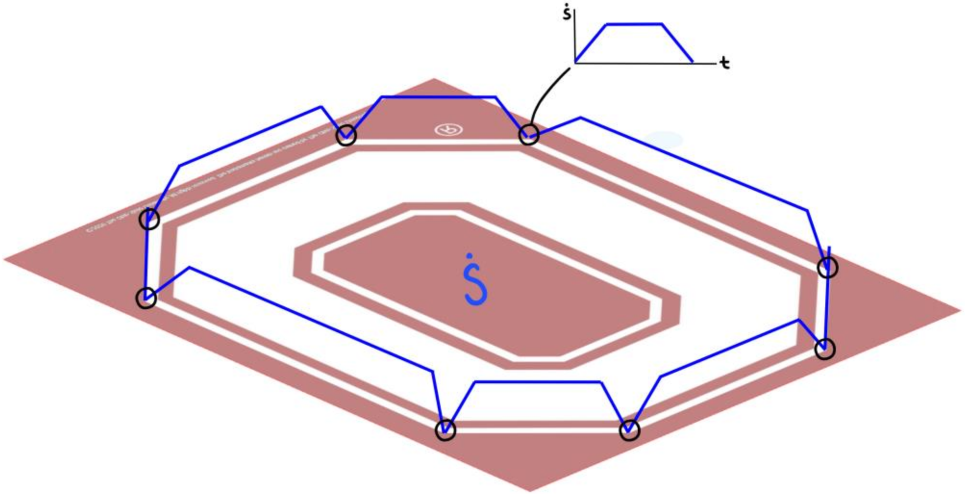
Path from $W_0, W_1, \dots W_8$:

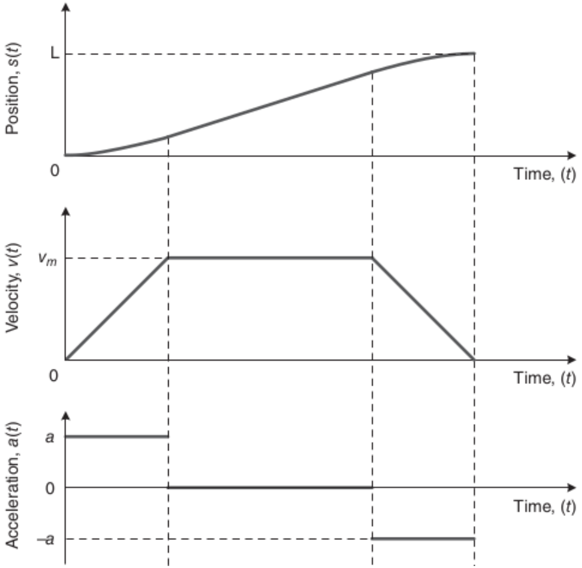
$$\underbrace{T^{0,0}}_{\text{start}} \rightarrow T^{0,1} \circlearrowright T^{1,1} \rightarrow T^{1,2} \circlearrowright T^{2,2} \dashrightarrow T^{6,7} \circlearrowright T^{7,7} \rightarrow \underbrace{T^{7,8}}_{\text{end}} = T^{7,0}$$

Time-Scaling

Trapezoidal Time-Scaling

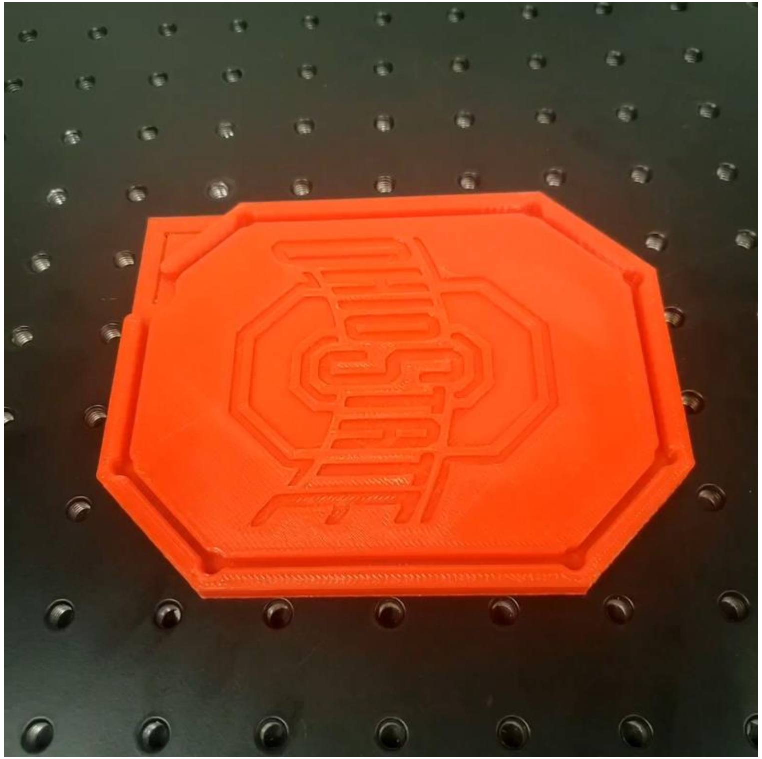
- v = 0.005 [m/s], a = 0.02 [m/s^2]
- v = π / 12 [rad/s], a = π / 6 [rad/s^2]
- Straight-line interpolation in task space
- 200 Hz

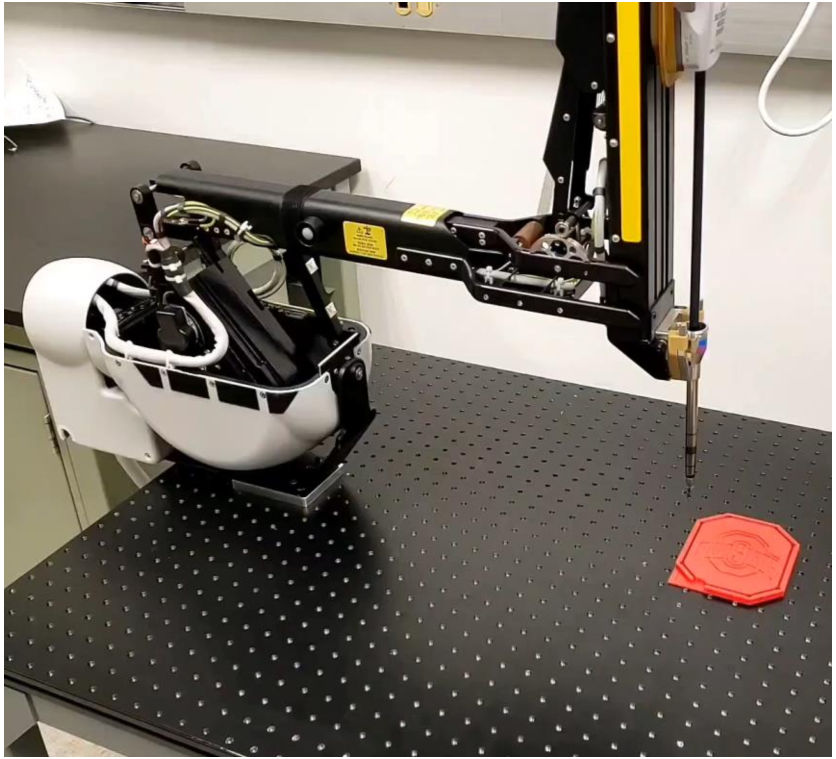




$$p(s) = p_{\text{start}} + s(p_{\text{end}} - p_{\text{start}}),$$
$$R(s) = R_{\text{start}} \exp(\log(R_{\text{start}}^{\mathrm{T}} R_{\text{end}})s)$$

Demonstration



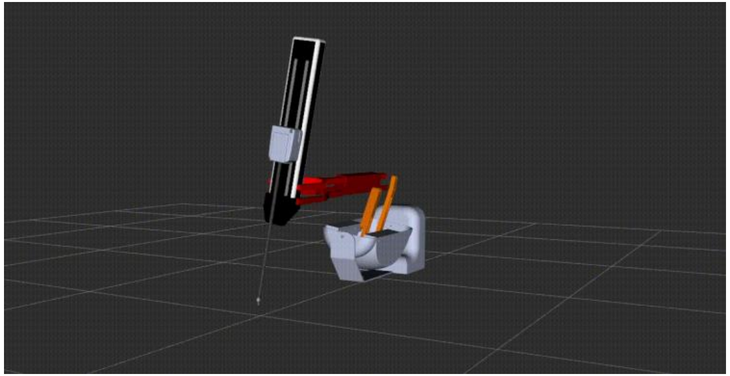


Hardware and software

dVRK for dummies: Hardware and software integration for PSM

- Installing dVRK in personal machine:

https://github.com/ferdous-alam/DVRK_robot_project/blob/main/Documentation/Installing_dvrk.md



dVRK for dummies: Hardware and software integration for PSM

Surgical Assistant Workstation (SAW) package
(built on *cisst* libraries)

↓

Components

↓

Implements interfaces to devices

PSM software architecture

High-level control ↔ sawROS PSM

↑

Low-level control (i.e. PID)

↑

Hardware interfaces (I/O)

Hardware and software

dVRK for dummies: Hardware and software integration for PSM

Surgical Assistant Workstation
(SAW) package
(built on *cisst* libraries)

Components

Implements interfaces to
devices

Component based development layer

Wrapper

- C++ library that enables direct access to the raw I/O data
- C++ classes: provides more convenient API

Hardware and software

dVRK for dummies: Hardware and software integration for PSM

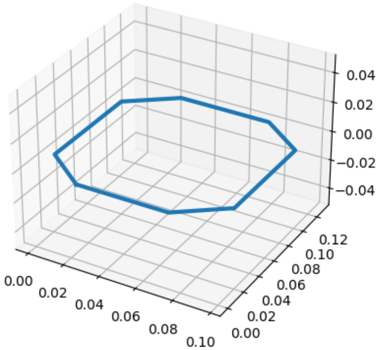
PSM software architecture

Component based development layer → High-level control → forward kinematics, inverse kinematics, trajectory generation, gripper control

Surgical Assistant Workstation (SAW) package (built on *cisst*)

Path planning using LSPB time scaling

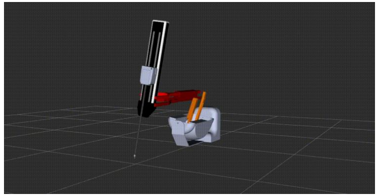
LSPB
time-scaling



Trajectory
generation

dVRK-ROS
python

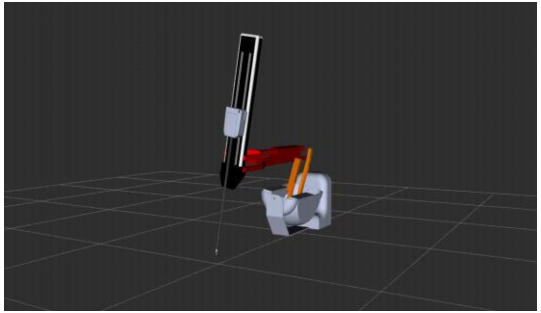
Higher-level
control



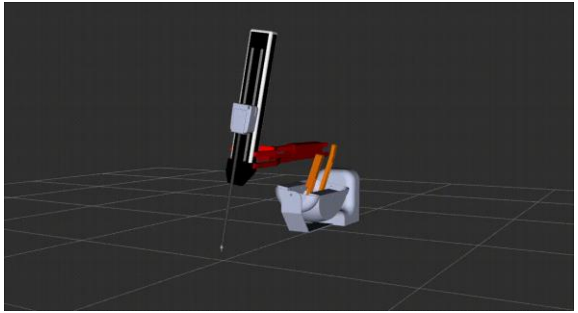
Hardware and software

Path planning at different velocities

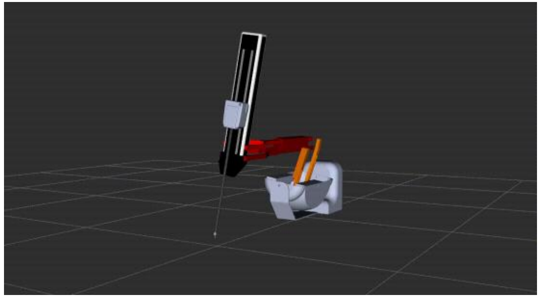
500 Hz



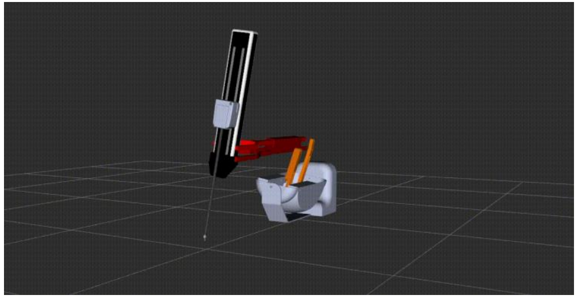
5 MHz



2 MHz



10 MHz

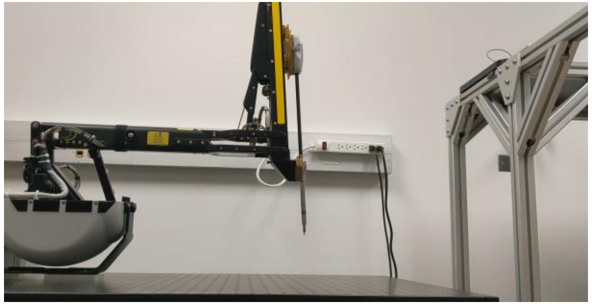


Hardware and software

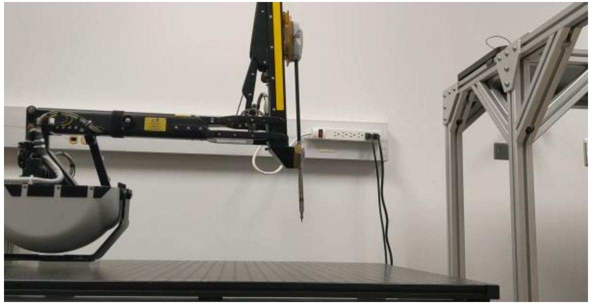
Path planning at different velocities

Sampling rate = 200 Hz

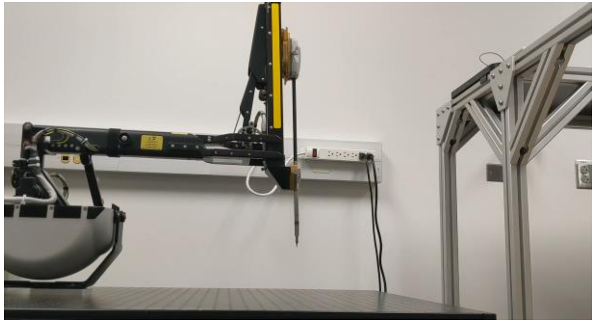
$v = 5mm/s, a = 10mm/s^2$



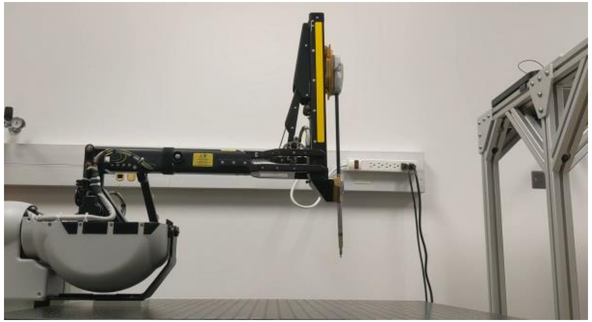
$v = 7.5mm/s, a = 14mm/s^2$



$v = 10mm/s, a = 20mm/s^2$



$v = 20mm/s, a = 40mm/s^2$



Thank you!

Questions?

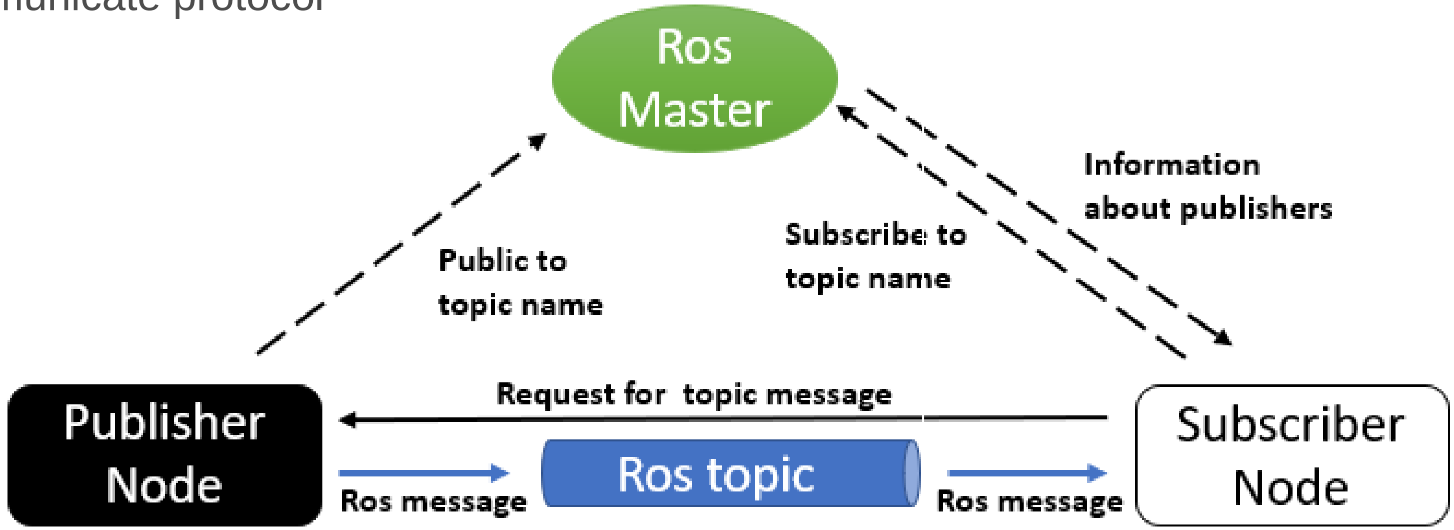
Trajectory Generation

- Block "O" Modulation
- Generate corner transformations
- Time-scaling loop, 200 Hz
- Compute entire trajectory

```
def main():
    # ***** BLOCK "O" MODIFICATION *****
    modified_waypoints_ordered = modify_blockO()
    # ***** GENERATE CORNER-WAYPOINT TRANSFORMATIONS *****
    waypoint_transformations = generate_waypoint_transformations(modified_waypoints_ordered)
    # ***** GENERATE ALL TRAPEZOIDAL TIME-SCALED TRANSFORMATIONS *****
    trajectory = generate_trapezoidal_transformations(waypoint_transformations, print_data=False)
    # ***** OUTPUT TRAJECTORY TO dVRK ROBOT *****
    return modified_waypoints_ordered, waypoint_transformations, trajectory
```

ROS & Python

- "Middleware"
- Communicate protocol



https://trojrobert.github.io/hands-on-introdution-to-robot-operating-system(ros)/