Drawing with Serial Robot Arm

Yu Hang He, Surya Prakash Senthil Kumar, & Qize Tao

Spring 2023 Final Project Presentation

ME 6407 Robotics

Introduction

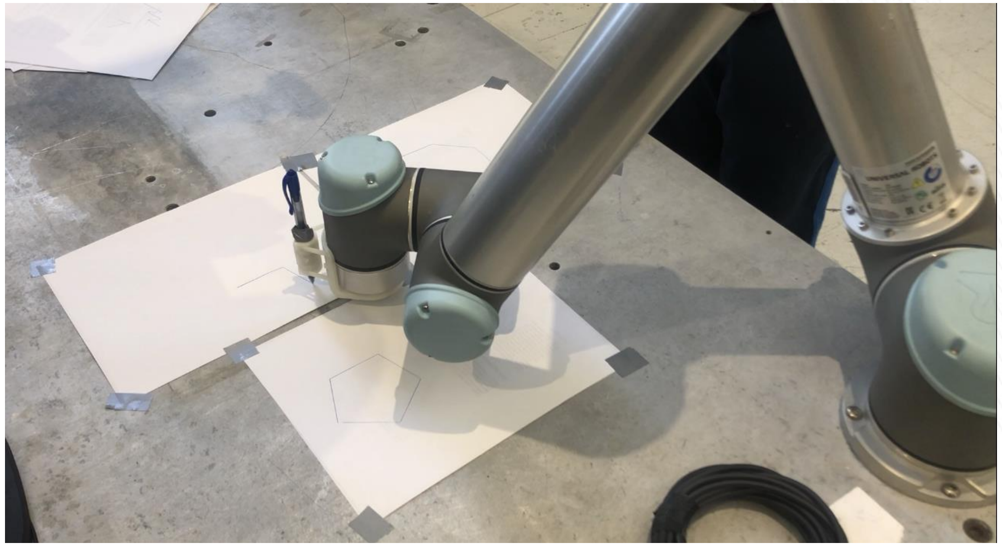
- Our objective is to program a serial chain robot (UR5) to draw an object in task space by getting the waypoints from an arbitrary image.
- Tasks involve Computing Kinematics, Image Processing and Trajectory generation.
- Simulation of UR5 robot in MATLAB demonstrating drawing complex shapes with three time scaling techniques.
- Implementation in real hardware demonstrating drawing of simple polygons (Pentagons, Star and Octagon).

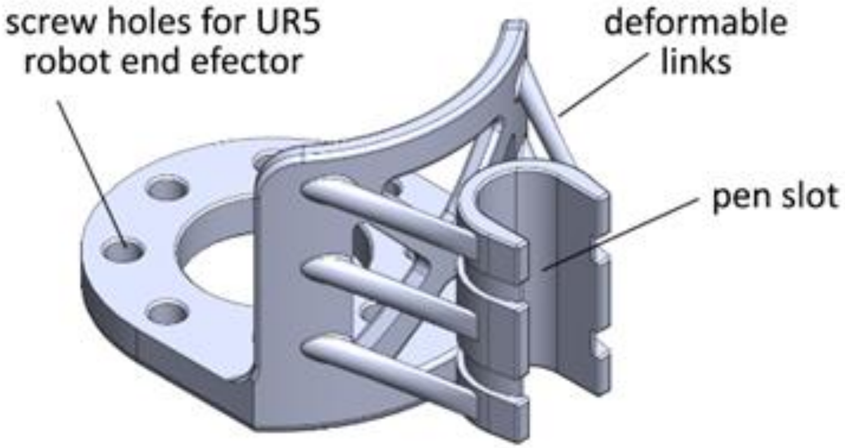
Motivation

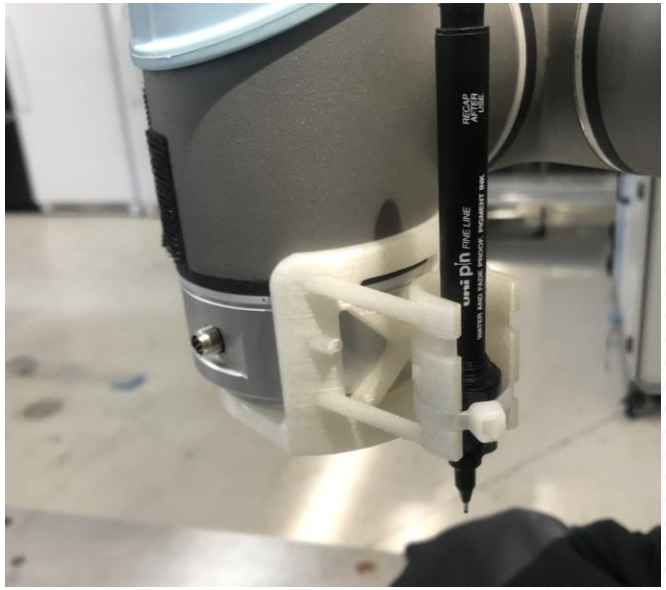
- Motivation:
  - Negative perception of AI image generation
  - No physical drawing process
  - Improve perception through physical drawing process with robot manipulator

Physical Setup

- UR-5 serial robot arm (6-DoF)
- Compliant pen holder
    - Maintains pen contact without force control



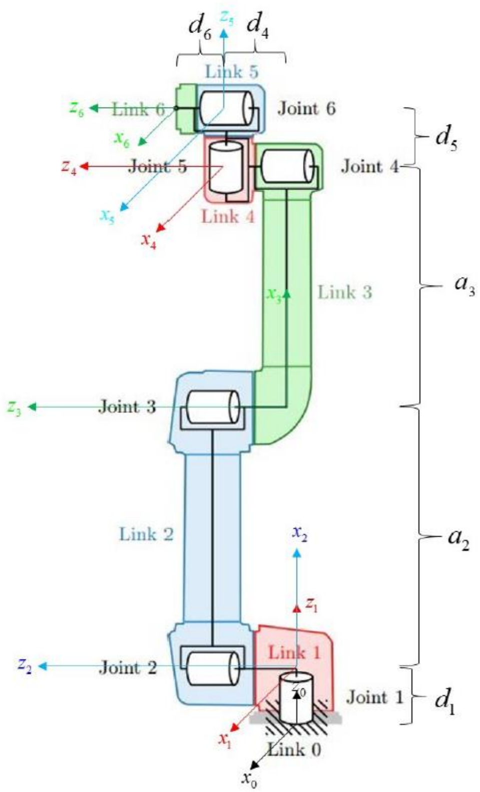




Georgia Tech
CREATING THE NEXT

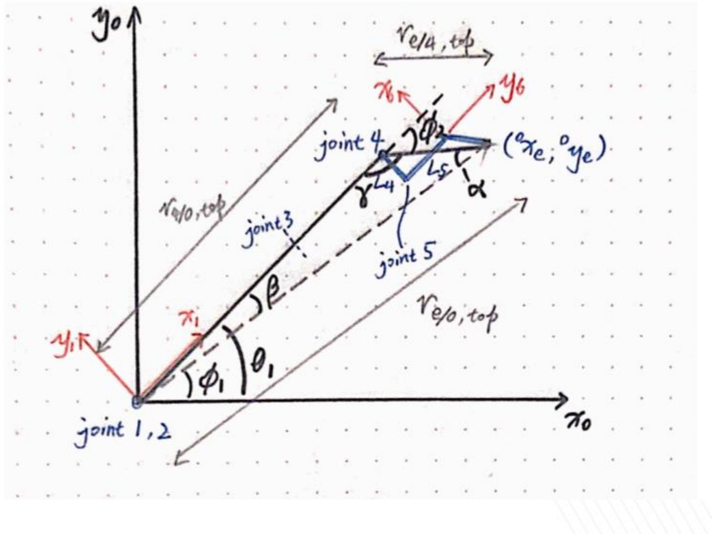
Kinematics

- Pen always point downward
  - joint 5 = π/2, joint 6 = 0
  - 6-DoF --> 4-DoF
- RDA with variable end effector size
  - Unique analytical solution in elbow-up configuration
  - Link length measured from CAD files

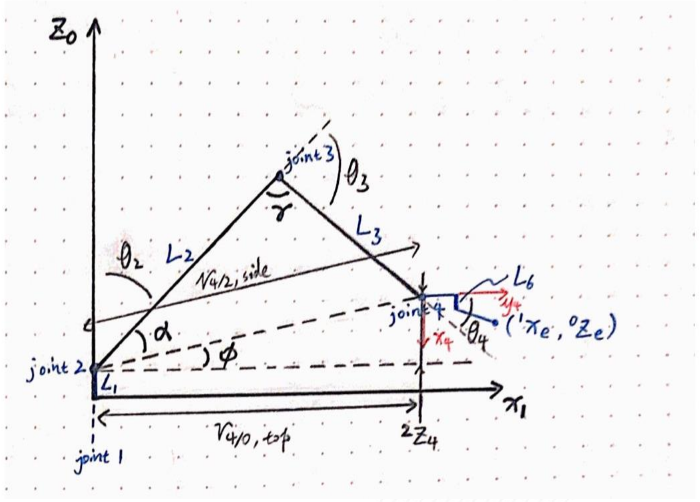


UR-5 schematics [2]

top view (ground frame)

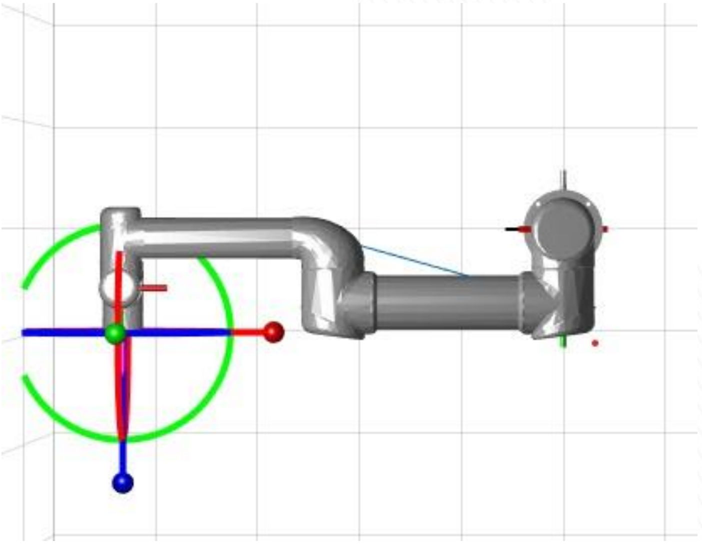


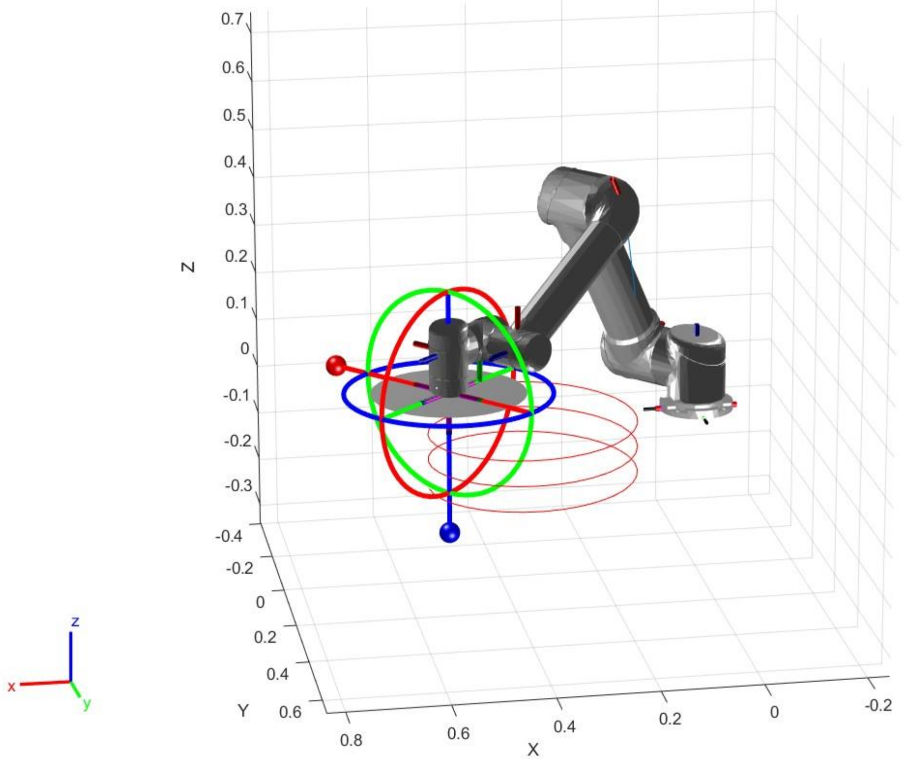
side view (link-1 frame)



Simulation in MATLAB

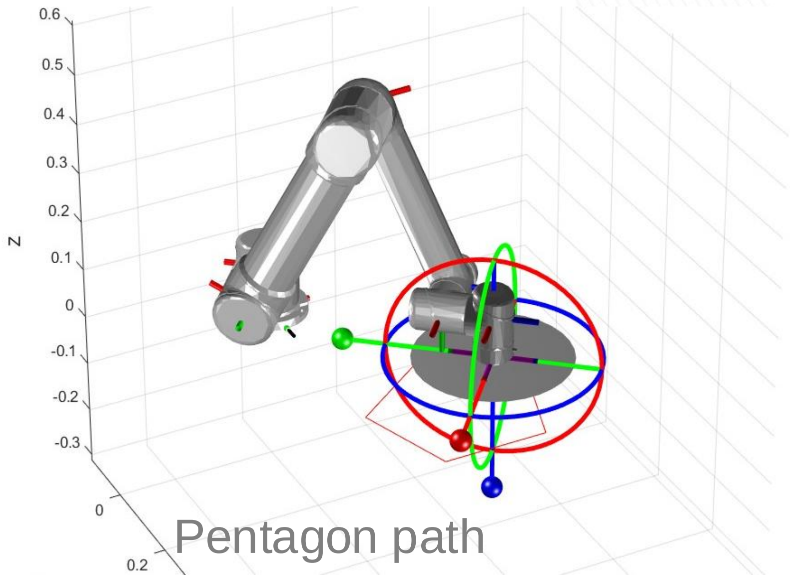
- MATLAB's [Robotic System Toolbox]
- Provided inverse kinematics solver leads to infeasible IK solutions for the physical robot.
- Integrated our RDA for the UR5 with the model.

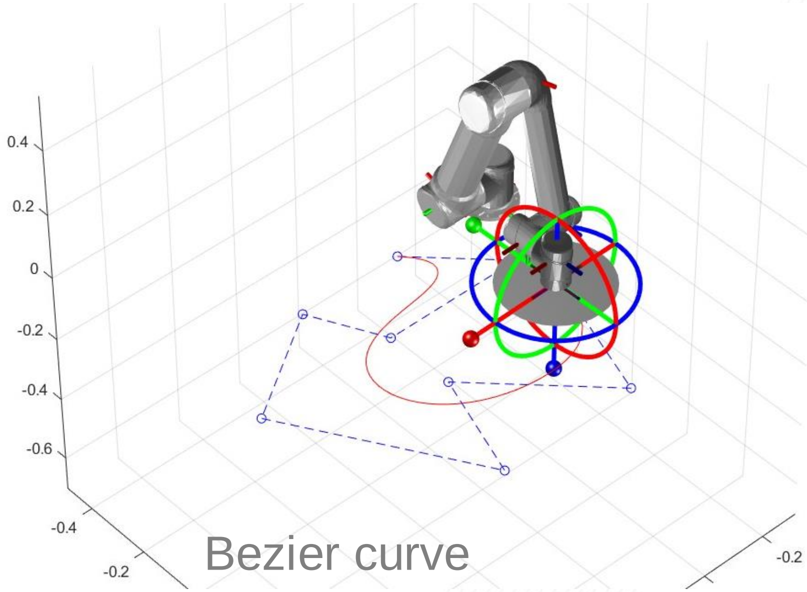




Trajectory Generation and Motion Planning

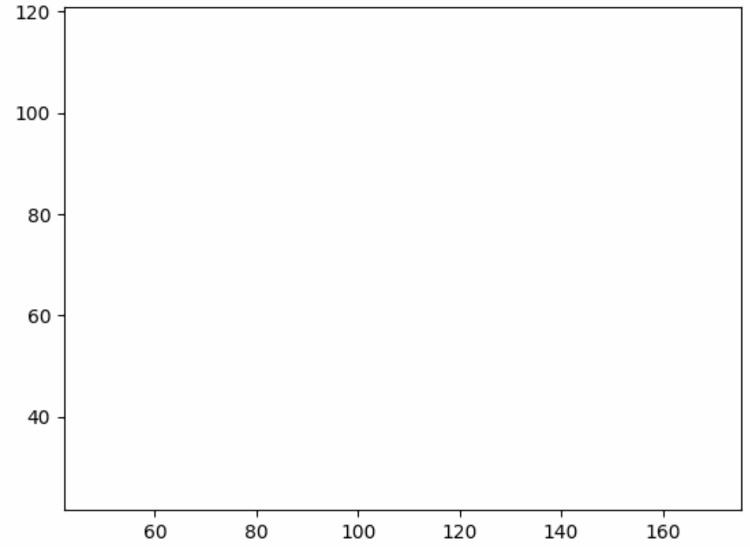
- Time scaling
  - Linear (given time)
  - 3rd order polynomial (given time)
  - Trapezoidal (given v_{max}, a)
- Path
  - Arbitrary polygon
  - Arch
  - Bézier curve

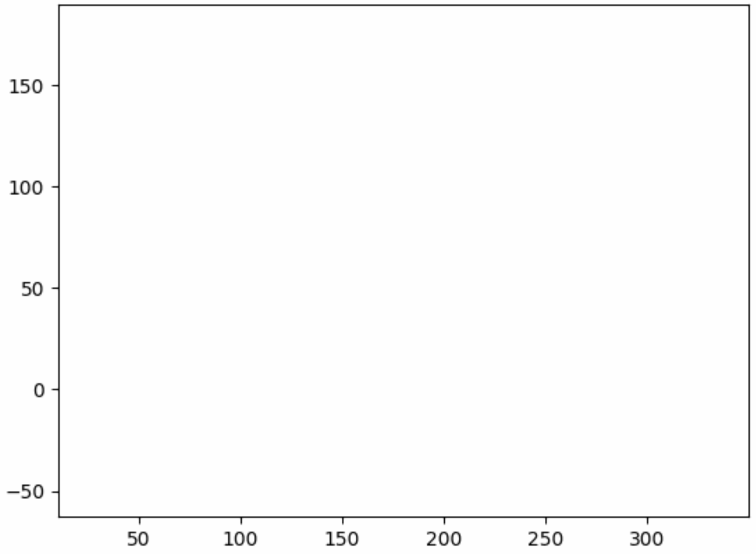




SVG Path Conversion

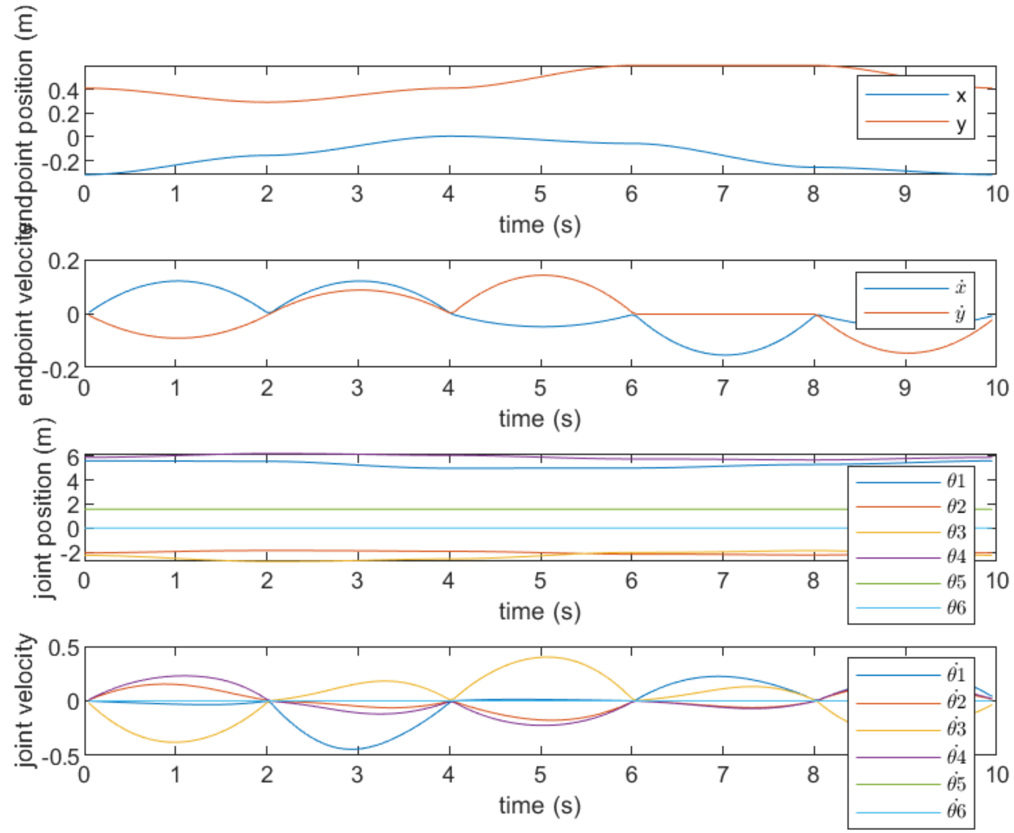
- Investigate vector image for generate more complex trajectory
- Parse SVG path format
- Discretize path description for trajectory generation
- Validate the trajectory in MATLAB simulation



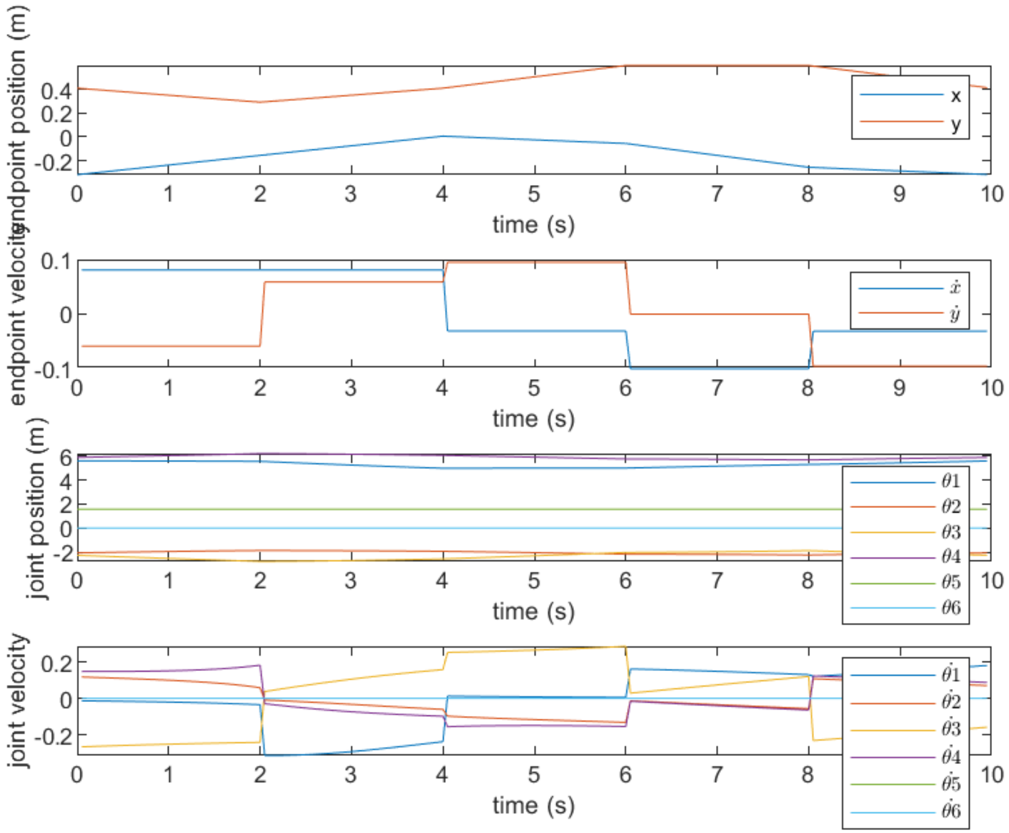


Georgia Tech
CREATING THE NEXT

Results – Simulation

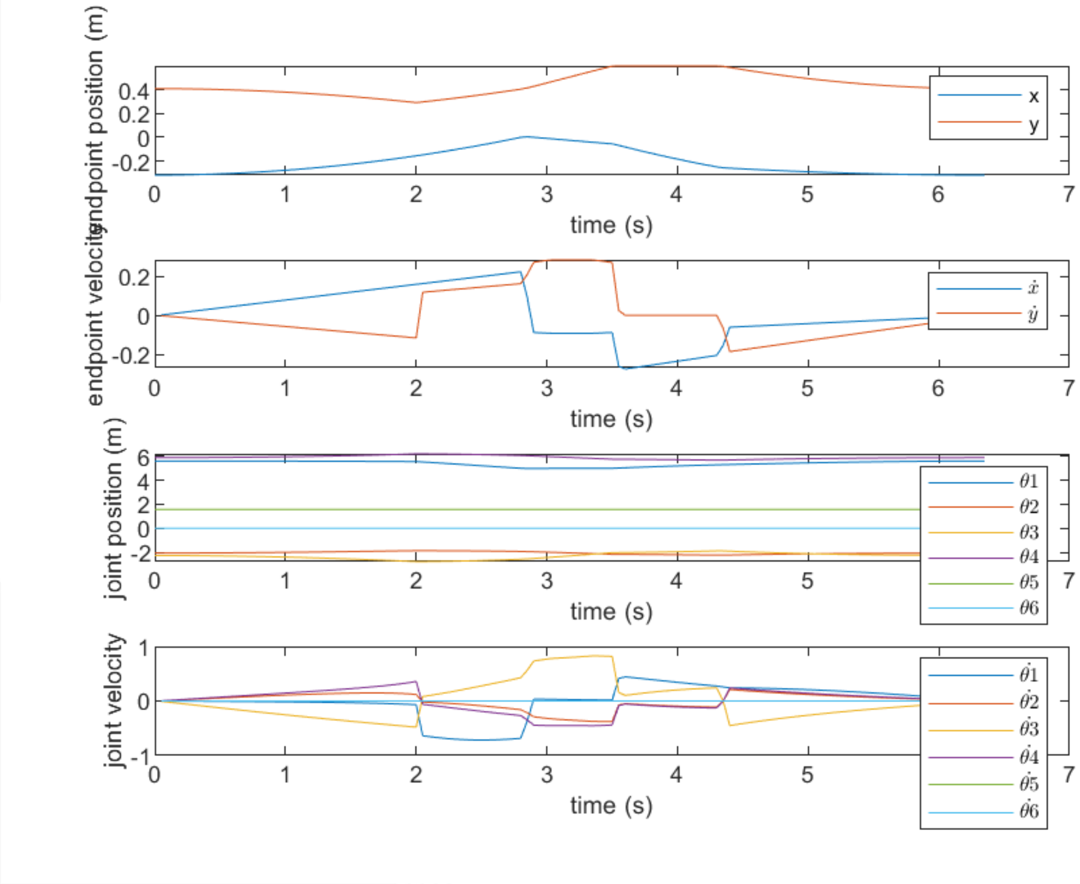


Polynomial time scaling - pentagon

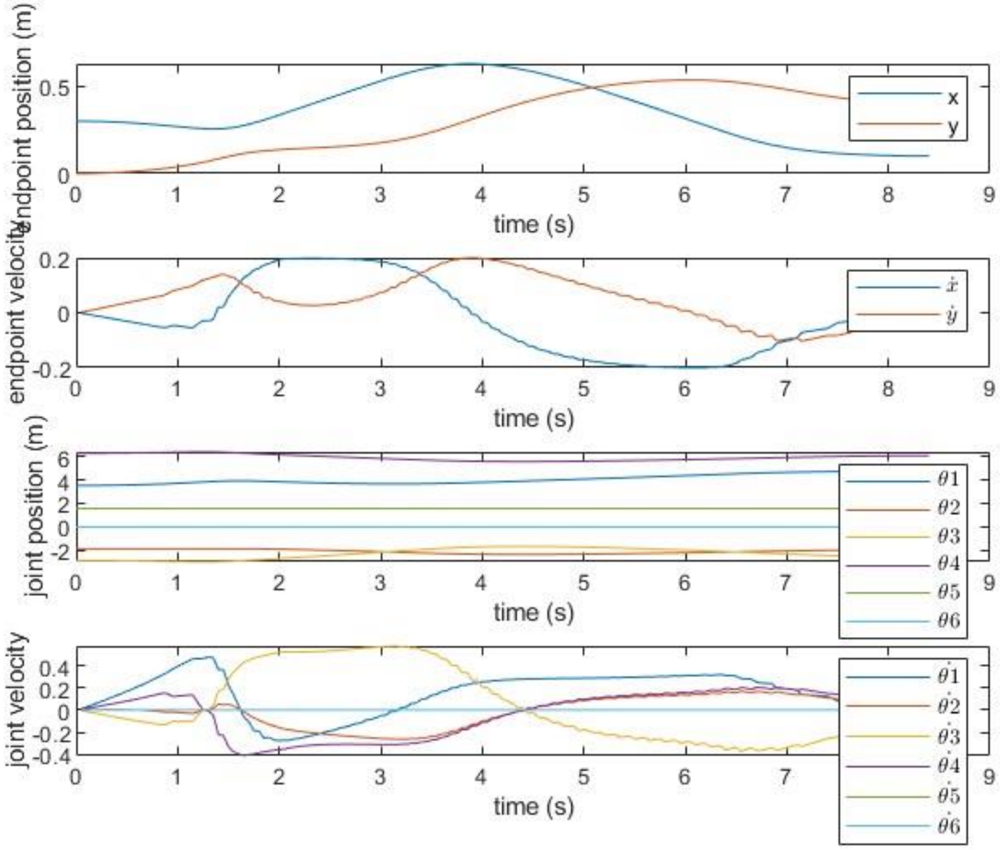


Linear time scaling - pentagon

Results – Simulation



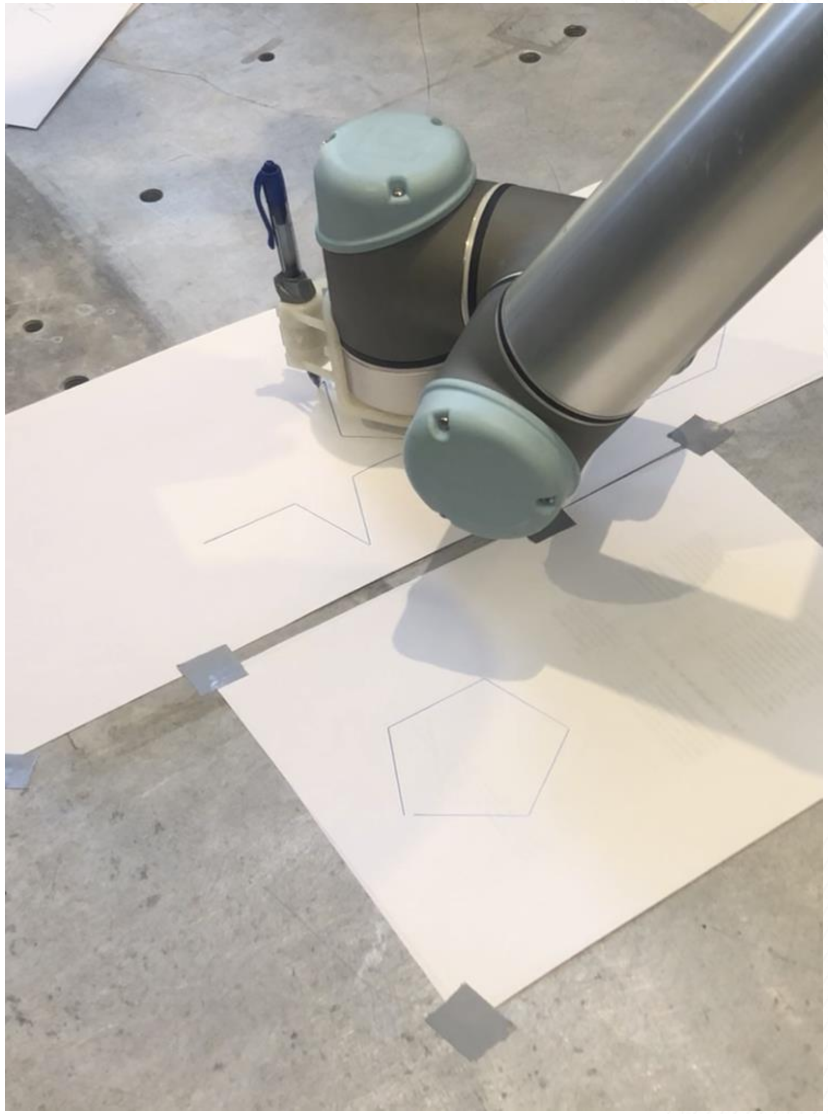
Trapezoidal time scaling – pentagon



Trapezoidal time scaling – curve

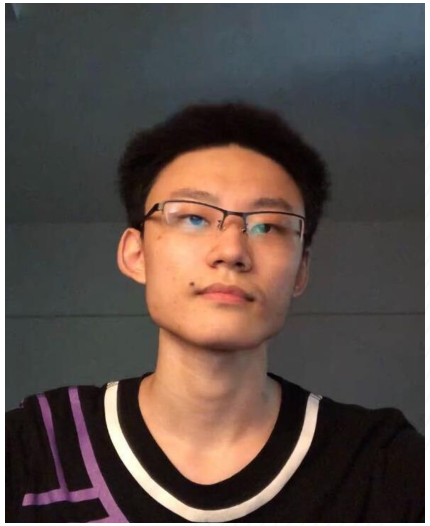
Results – Physical Robot

- Implement the MATLAB codes in python
- Communicate with physical robot through ROS and UR Driver
- Performed regular polygon drawing under linear and polynomial time scaling



Conclusion

- Trajectory transfer between simulation and physical can be challenging
- Failed simulation with ROS Gazebo
- Complicated art is difficult to process using CV technique
- Testing the trajectory on the physical robot earlier rather than later



Demo Video

Simulation

Appendix: RDA Derivation

Given:

Link lengths: $L_1 = 89.2mm$, $L_2 = 425mm$, $L_3 = 392.25mm$, $L_4 = 110mm$, $L_5 = 94.75mm$ and $L_6 = 81.5mm$

Desired end point position: $\overrightarrow{{}^0p_e} =< {}^0x_e, {}^0y_e, {}^0z_e >$

End point position in frame 6 (end effector size): $\overrightarrow{{}^6p_e} =< {}^6x_e, {}^6y_e, {}^6z_e >$

1)

$$r_{e/0,top} = \sqrt{{}^0x_e^2 + {}^0y_e^2} \qquad \text{(end point to origin distance on } x-y \text{ plane)}$$

$$r_{e/4,top} = \sqrt{({}^6x_e - L_4)^2 + ({}^6y_e + L_5)^2} \qquad \text{(end point to joint 4 distance on x-y plane)}$$

$$\phi_1 = atan2({}^0y_e, {}^0x_e)$$

$$\phi_2 = atan2(L_4 - {}^6x_e, {}^6y_e + L_5)$$

$$\gamma_1 = \pi - |\phi_2|$$

$$\alpha_1 = \arcsin\left(\sin\gamma_1 \cdot \frac{r_{e/4,top}}{r_{e/0,top}}\right) \qquad \text{(law of sine)}$$

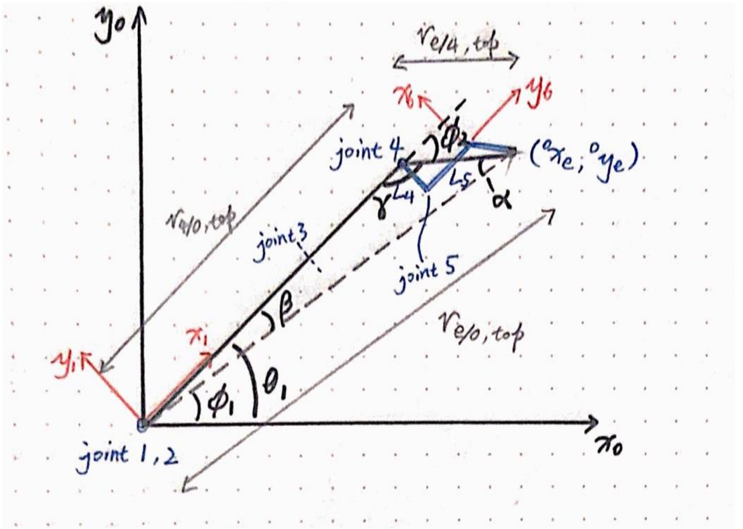
$$\beta_1 = |\phi_2| - \alpha_1$$

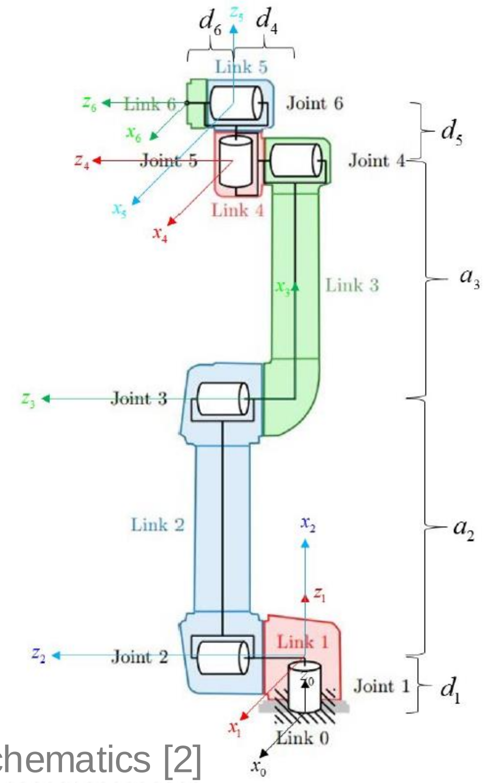
$$\begin{aligned}\theta_1 &= \phi_1 + sign(\phi_2)\beta_1 \\ &= \phi_1 + \phi_2 - sign(\phi_2)(\alpha_1)\end{aligned} \qquad \text{solution for } \theta_1$$

$$r_{4/0,top} = \sqrt{r_{e/0,top}^2 + r_{e/4,top}^2 - 2r_{e/0,top}r_{e/4,top}\cos\beta_1} \qquad \text{(joint 4 to origin distance on x-y plane)}$$

for theta_2, 3 and 4 calculation

top view (ground frame)





UR-5 schematics [2]

Appendix: RDA Derivation

2)

$$^0z_4 = {^0z_e} + {^6z_6} + L_6 \qquad \text{(joint 4 to ground height)}$$

$$^2z_4 = {^0z_4} - L_1 \qquad \text{(joint 2 and 4 height difference)}$$

$$r_{4/2,side} = \sqrt{r^2_{4/0,top} + {^2z_4^2}} \qquad \text{(joint 2 to 4 distance on } x_1 - z_0 \text{ plane)}$$

$$\phi = atan2(^2z_4, r_{4/0,top})$$

$$\alpha_2 = \arccos\left(\frac{L_2^2 + r^2_{4/2,side} - L3^2}{2L_2 r_{4/2,side}}\right) \qquad \text{(law of cosine)}$$

$$\gamma_2 = \arccos\left(\frac{L_2^2 + L_3^2 - r^2_{4/2,side}}{2L_2L_3}\right) \qquad \text{(law of cosine)}$$

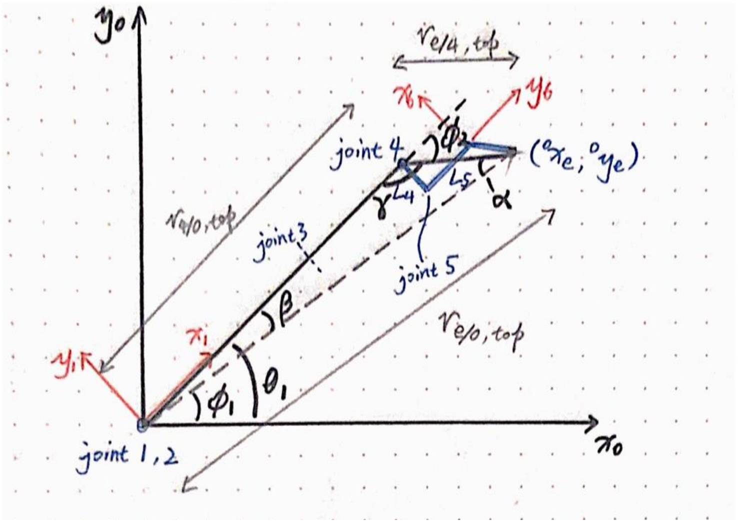
$$\theta_2 = -(\pi/2 - \alpha_2 - \phi) \qquad \text{solution for } \theta_2$$

$$\theta_3 = -(\pi - \gamma_2) \qquad \text{solution for } \theta_3$$

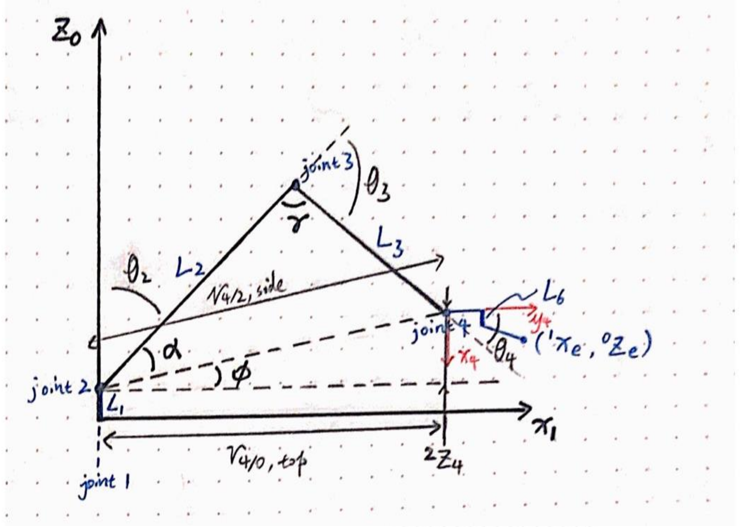
$$\theta_4 = -\pi/2 - \theta_2 - \theta_3 \qquad \text{solution for } \theta_4$$

$\theta_5 = \pi/2$ and $\theta_6 = 0$ at all time.

top view (ground frame)



side view (link-1 frame)



References

[1] J. Villalobos, I. Y. Sanchez and F. Martell, "Statistical comparison of Denavit-Hartenberg based inverse kinematic solutions of the UR5 robotic manipulator," *2021 International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME)*, Mauritius, Mauritius, 2021, pp. 1-6.

[2] P. M. Kebria, S. Al-wais, H. Abdi and S. Nahavandi, "Kinematic and dynamic modelling of UR5 manipulator," *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Budapest, Hungary, 2016.

[3] J. Arruti, M. Ottavianelli, A. Solari, P. Monzón, P. Musé and J. P. Oliver, "A drawing robot pipeline with artist-inspired execution," *2021 IEEE URUCON*, Montevideo, Uruguay, 2021, pp. 461-466.

[4] R. B. Venkataramaiyer, S. Kumar and V. P. Namboodiri, "Can I teach a robot to replicate a line art," *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, Snowmass, CO, USA, 2020, pp. 1922-1930.

[5] F. Gao, J. Zhu, Z. Yu, P. Li and T. Wang, "Making Robots Draw A Vivid Portrait In Two Minutes," *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Las Vegas, NV, USA, 2020, pp. 9585-9591.

[6] S. Calinon, J. Epiney and A. Billard, "A humanoid robot drawing human portraits," *5th IEEE-RAS International Conference on Humanoid Robots, 2005.*, Tsukuba, Japan, 2005, pp. 161-166.

[7] G. Jean-Pierre and Z. Saïd, "The artist robot: A robot drawing like a human artist," *2012 IEEE International Conference on Industrial Technology*, Athens, Greece, 2012, pp. 486-491.

[8] T. Chettibi, M. Haddad, H. E. Lehtihet and W. Khalil, "Suboptimal Trajectory Generation for Industrial Robots using Trapezoidal Velocity Profiles," *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Beijing, China, 2006, pp. 729-735.

[9] J. Lin, N. Somani, B. Hu, M. Rickert and A. Knoll, "An Efficient and Time-Optimal Trajectory Generation Approach for Waypoints Under Kinematic Constraints and Error Bounds," *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Madrid, Spain, 2018, pp. 5869-5876.

[10] Foit, Krzysztof. "Controlling the movement of the robot's effector on the plane using the SVG markup language." *Advanced Materials Research*. Vol. 837. Trans Tech Publications Ltd, 2014.