AMORO Lab – Biglide Report (Full Version)

Master CORO-IMARO: Control and Robotics

2025/2026

Submitted by:

Amruth BIKKALAHALLY NANJUNDAPPA

Kiran THYAGARAJAN

Raphael Kifen ASEME

Submission Date:

01/11/2025

Contents

1 Introduction 2

2 Comparison of Models 2
2.1 Overview of Implemented Models . . . 2
2.2 Direct Geometric Model (DGM) . . . 3
2.3 Passive Joint geometric model . . . 4
2.4 Inverse Geometric Model (IGM) . . . 5
2.5 First Order Forward Kinematic Model . . . 6
2.6 Inverse First Order Kinematic Model . . . 7
2.7 Passive joint Kinematic Model . . . 8
2.8 Forward Second Order Kinematic Model . . . 9
2.9 Inverse Second Order Kinematic Model . . . 10
2.10 Inverse Dynamic Model . . . 11

3 Computed Torque Control and Trajectory Generation 12
3.1 Trajectory Generation . . . 12
3.2 Control Law . . . 12
3.3 Simulation Results . . . 13

4 Crossing Type-2 Singularity 18
4.1 Identification of the Singularity . . . 18
4.2 Polynomial Trajectory for Singularity Crossing . . . 18
4.3 Numerical Setup . . . 18
4.4 Mathematical Interpretation . . . 19
4.5 Practical Interpretation . . . 19

5 Conclusion 19

1 Introduction

This report presents the complete modeling and control work carried out on the **Biglide** parallel robot as part of the AMORO laboratory project. The objective was to compute the geometric, kinematic, and dynamic models of the mechanism, verify them against a Gazebo simulation, and design a controller for trajectory tracking.

The Biglide is a planar two-degree-of-freedom manipulator actuated by two prismatic joints and connected to the mobile platform via passive revolute joints. The modeling phase consisted of implementing both direct and inverse models up to second order, while the control phase focused on using these models for feedforward compensation. Finally, the system behavior was studied when crossing a **Type-2 singularity**, a configuration where the parallel Jacobian loses rank.

2 Comparison of Models

2.1 Overview of Implemented Models

The models were computed and implemented in `biglide_models.py` and tested within the ROS2–Gazebo environment. The models included:

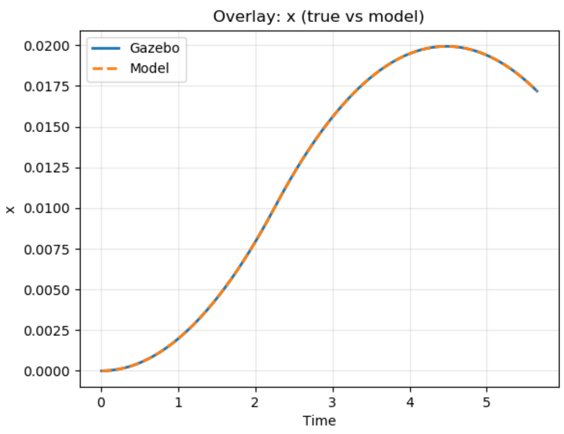
- Direct and Inverse Geometric Models (DGM, IGM)
- First Order Kinematic Models (DKM, IKM)
- Second Order Kinematic Models (DKM2, IKM2)
- Inverse Dynamic Model

Each model was validated by comparing its output with data obtained directly from the Gazebo simulation using the robot interface.

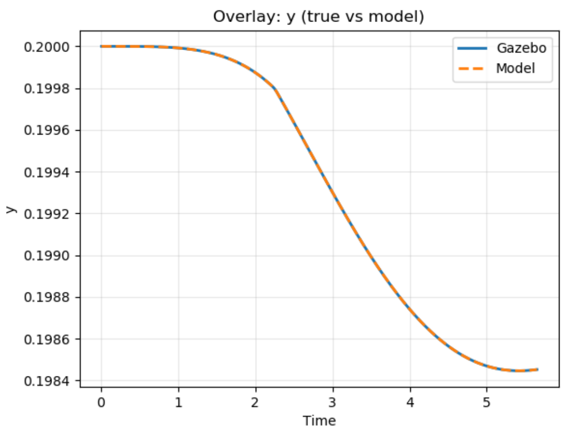
Additionally, the `biglide_models.py` script was used to verify if the errors were in the admissible limit.

2.2 Direct Geometric Model (DGM)

The direct geometric model was first verified. Figure 1 compares the computed and simulated end-effector coordinates x and y.



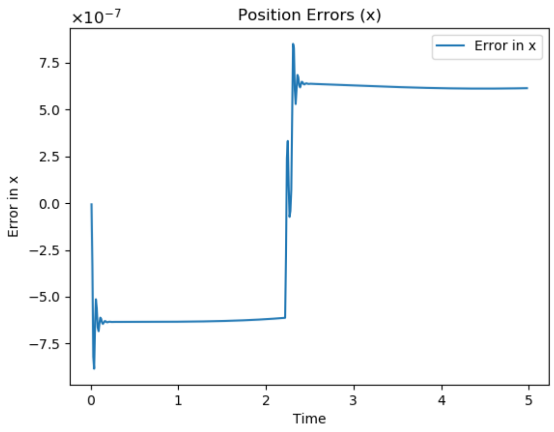
(a) End-effector position $x(t)$



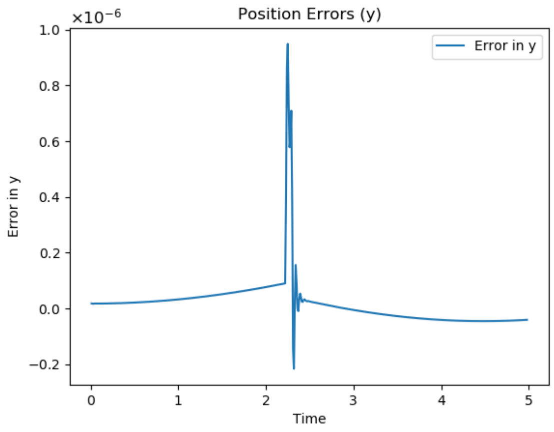
(b) End-effector position $y(t)$

Figure 1: Comparison between simulated and computed end-effector positions.

The two curves in Figure 1 overlap almost perfectly, showing that the DGM equations were correctly implemented. The small remaining difference between both curves is shown below as the tracking error.



(a) Error in x: $e_x = x_{\text{meas}} - x_{\text{model}}$



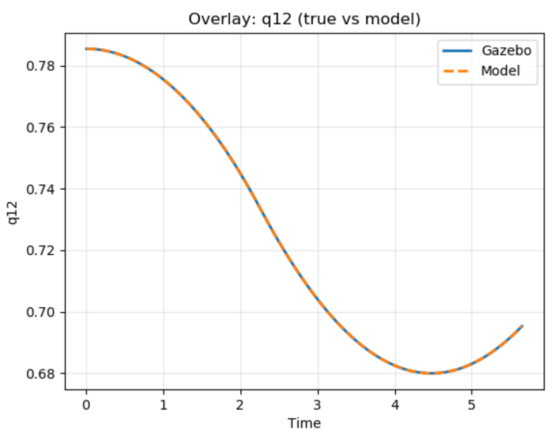
(b) Error in y: $e_y = y_{\text{meas}} - y_{\text{model}}$

Figure 2: Position tracking error of the end-effector.

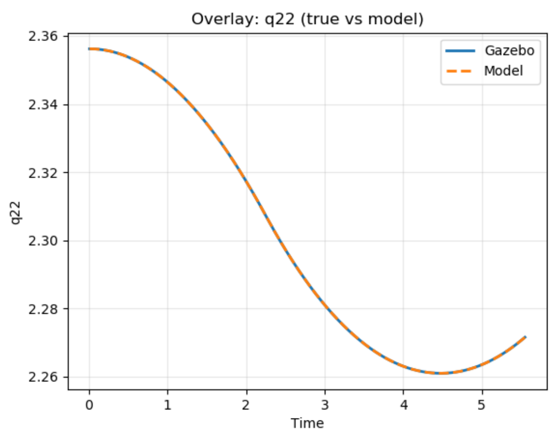
The error stays in the order of 10^{-7}, confirming an almost exact match between the computed and simulated data. Overall, the DGM correctly models the geometry of the Biglide.

2.3 Passive Joint geometric model

The DGM was also checked for the passive joints (q_{12}, q_{22}), which are not actuated but follow the mechanism's geometry. The computed and simulated angles were compared to confirm the correctness of the geometric constraints.

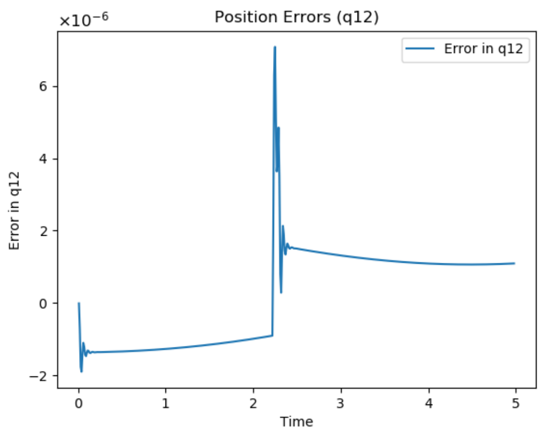


(a) Passive joint $q_{12}(t)$

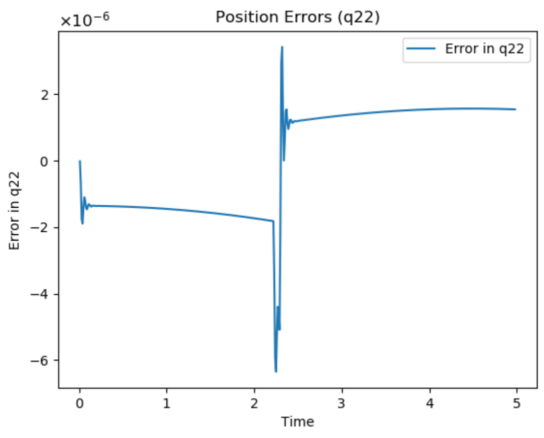


(b) Passive joint $q_{22}(t)$

Figure 3: Measured passive joint positions vs. computed.



(a) Error in q_{12}: $e_{q_{12}} = q_{12,\text{meas}} - q_{12,\text{model}}$



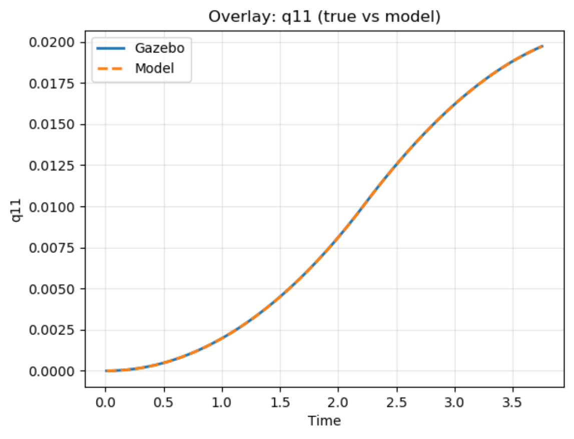
(b) Error in q_{22}: $e_{q_{22}} = q_{22,\text{meas}} - q_{22,\text{model}}$

Figure 4: Tracking error of the passive joints.

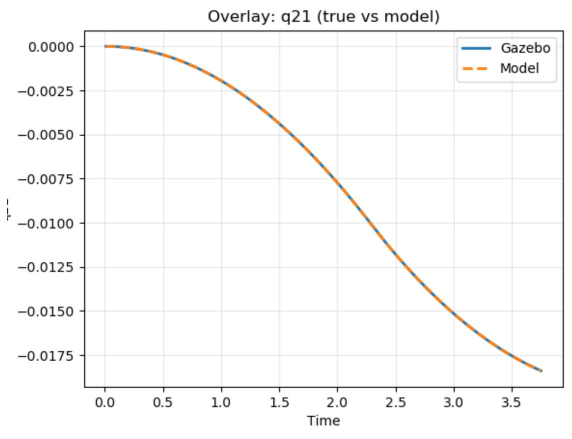
The results show a very small error of order 10^{-6}, confirming that the passive joint motion is well captured by the model. The geometric model is therefore validated for passive joints.

2.4 Inverse Geometric Model (IGM)

The Inverse Geometric Model (IGM) computes the actuator positions (q_{11}, q_{21}) from the Cartesian coordinates (x, y) of the end-effector. It performs the opposite mapping of the DGM. We compared the joint positions obtained from the computed data with the ones measured in Gazebo.



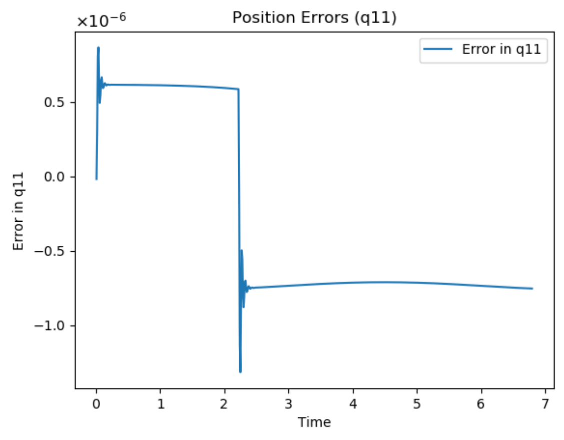
(a) Actuated joint $q_{11}(t)$



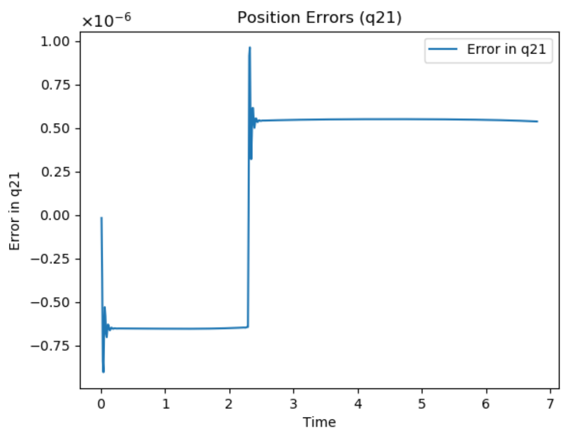
(b) Actuated joint $q_{21}(t)$

Figure 5: Measured actuator positions vs. computed IGM .

The simulated and analytical curves in Figure 5 are almost identical, proving that the inverse geometric model works correctly. The small velocity-dependent fluctuations are in the accepted limits according to `biglide_models.py`.



(a) Error in q_{11}: $e_{q_{11}} = q_{11,\text{meas}} - q_{11,\text{model}}$



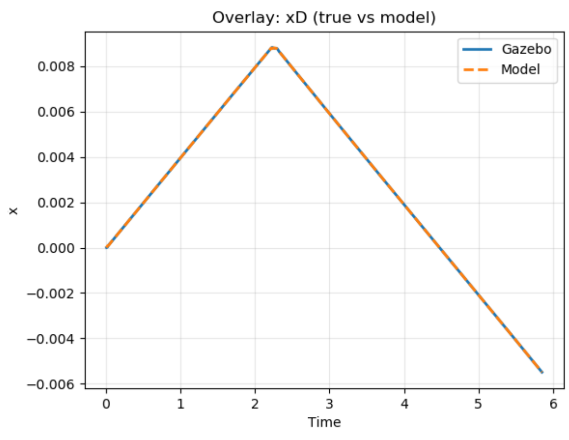
(b) Error in q_{21}: $e_{q_{21}} = q_{21,\text{meas}} - q_{21,\text{model}}$

Figure 6: Position tracking error for the actuated joints.

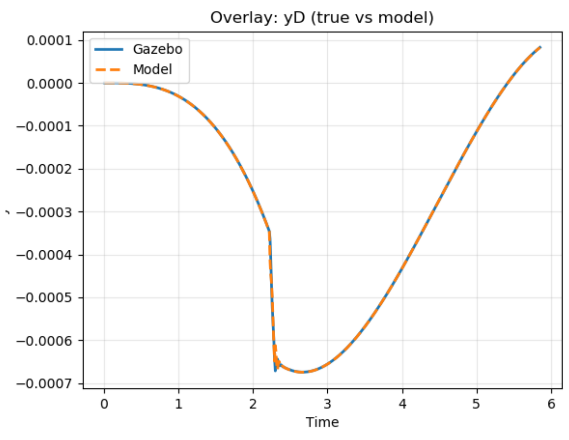
The tracking error remains negligible, confirming the IGM correctness.

2.5 First Order Forward Kinematic Model

The First Order Forward Kinematic Model gives the end-effector velocity $(\dot{x}, \dot{y})$ from the actuator velocities $(\dot{q}_{11}, \dot{q}_{21})$. The computed results were compared with the measured velocities in Gazebo.



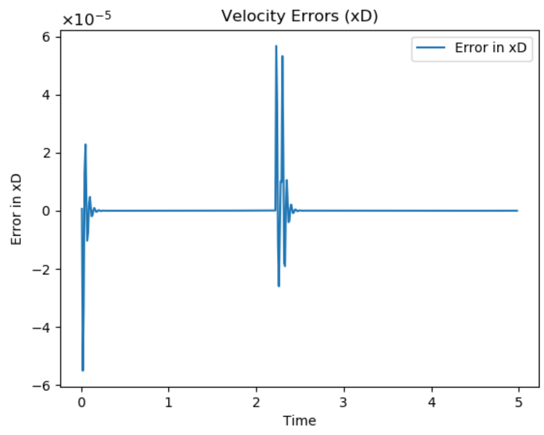
(a) $\dot{x}(t)$



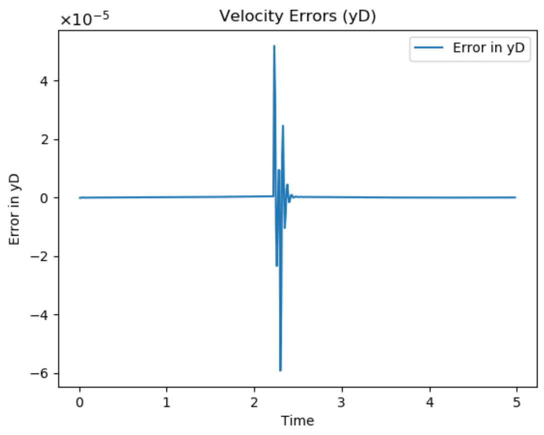
(b) $\dot{y}(t)$

Figure 7: Measured and computed end-effector velocities.

The plots in Figure 7 overlap almost exactly, which confirms that the velocity model is correctly derived.



(a) Error in $\dot{x}$: $e_{\dot{x}} = \dot{x}_{\text{meas}} - \dot{x}_{\text{model}}$



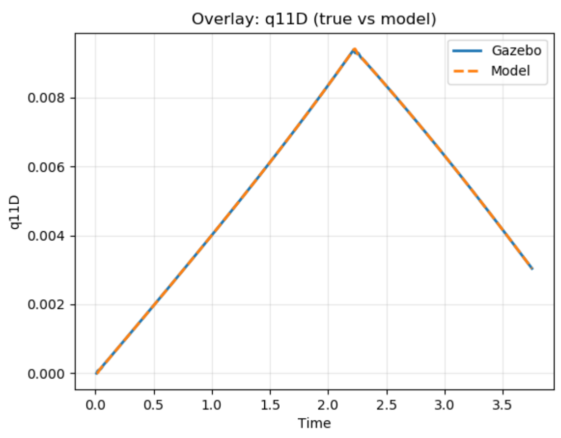
(b) Error in $\dot{y}$: $e_{\dot{y}} = \dot{y}_{\text{meas}} - \dot{y}_{\text{model}}$

Figure 8: Velocity tracking error of the end-effector.

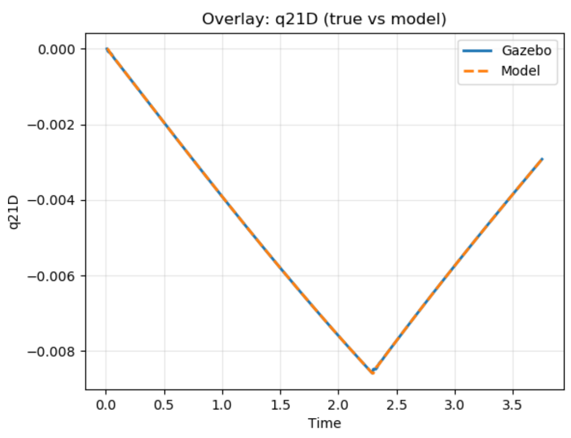
The velocity error remains very small, validating that the first order kinematic model correctly represents the instantaneous mapping between joint and Cartesian velocities.

2.6 Inverse First Order Kinematic Model

The Inverse First Order Kinematic Model computes the required actuator velocities $(\dot{q}_{11}, \dot{q}_{21})$ for a desired end-effector velocity $(\dot{x}, \dot{y})$. It represents the inverse of the Jacobian relationship. The computed joint velocities were compared with those measured in Gazebo.



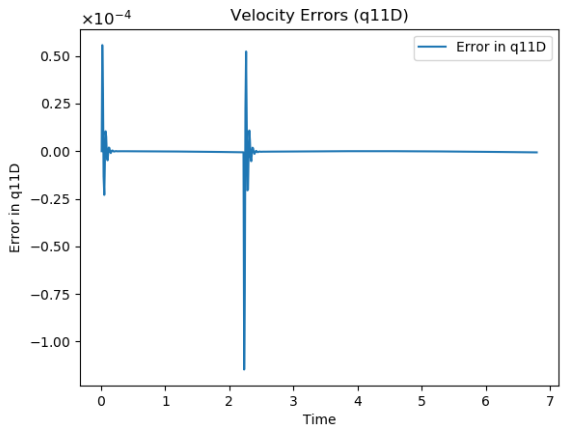
(a) $\dot{q}_{11}(t)$



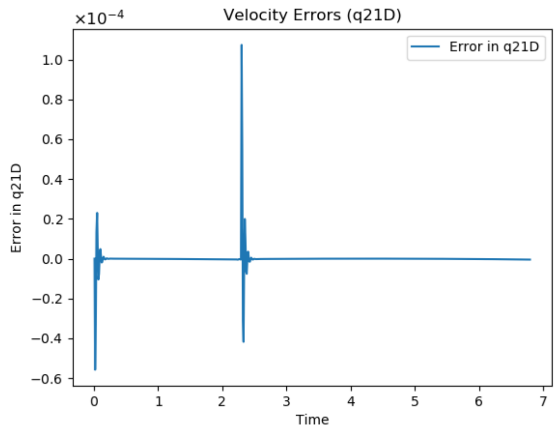
(b) $\dot{q}_{21}(t)$

Figure 9: Measured and computed actuator velocities.

The results in Figure 9 show a very close match between the two signals for both actuators, confirming that the inverse velocity model is correctly derived. The small remaining difference is shown in Figure 10.



(a) Error in $\dot{q}_{11}$



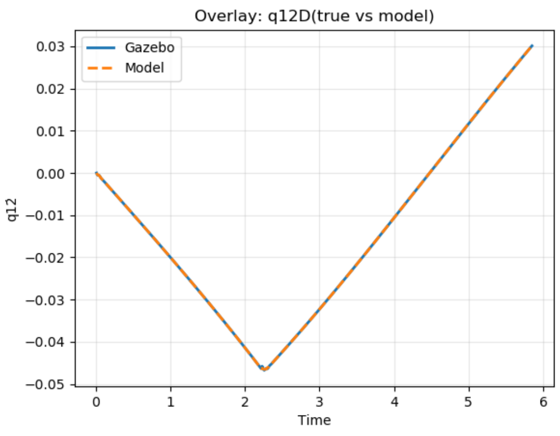
(b) Error in $\dot{q}_{21}$

Figure 10: Tracking error of actuator velocities.

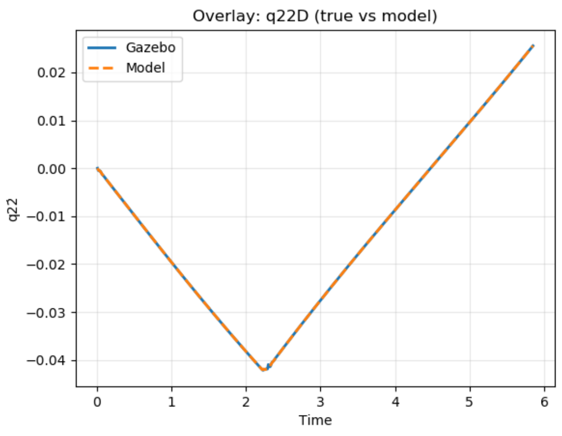
The error amplitude is very small, with minor spikes. Overall, the inverse first order kinematic model is validated and consistent with the simulation.

2.7 Passive joint Kinematic Model

The Passive Joint Kinematic Model gives the passive joint velocities $(\dot{q}_{12}, \dot{q}_{22})$ from the desired end-effector velocity $(\dot{x}, \dot{y})$ and actuator velocities $(\dot{q}_{11}, \dot{q}_{21})$. The computed results were compared with the measured velocities in Gazebo.



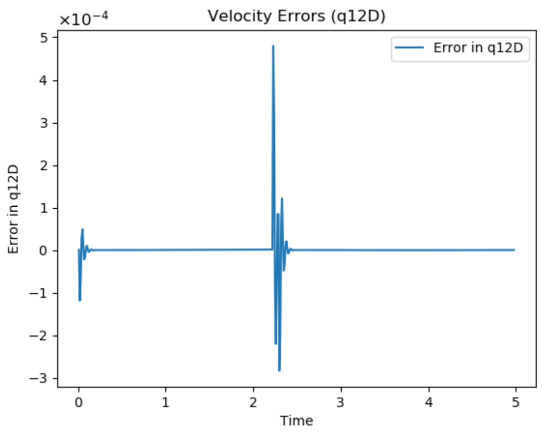
(a) $\dot{q}_{12}(t)$



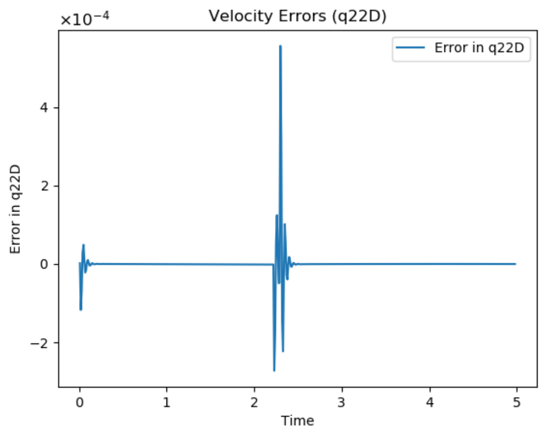
(b) $\dot{q}_{22}(t)$

Figure 11: Measured and computed passive joint velocities.

The plots in Figure 11 overlap almost exactly, which confirms that the velocity model is correctly derived.



(a) Error in $\dot{q}_{12}$: $e_{\dot{q}_{12}} = \dot{q}_{12\text{ meas}} - \dot{q}_{12\text{ model}}$



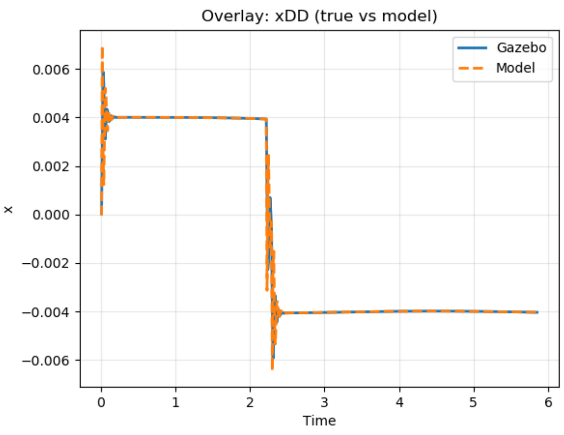
(b) Error in $\dot{y}$: $e_{\dot{q}_{22}} = \dot{q}_{22\text{ meas}} - \dot{q}_{22\text{ model}}$

Figure 12: Velocity tracking error of the passive joints.

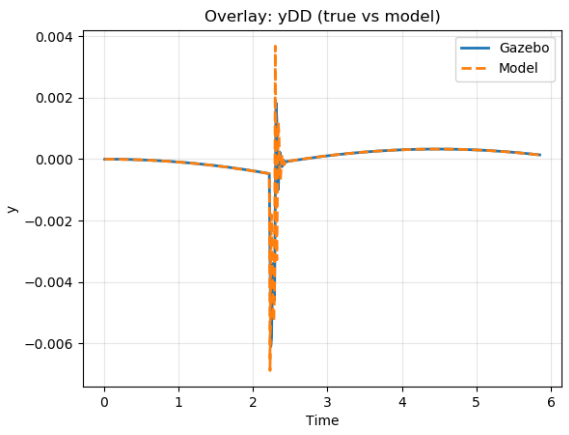
The velocity error remains very small, thus the correctness of the passive joint kinematic model is validated.

2.8 Forward Second Order Kinematic Model

The Second Order Kinematic Model gives the end-effector acceleration $(\ddot{x}, \ddot{y})$ from the actuator accelerations $(\ddot{q}_{11}, \ddot{q}_{21})$ and passive joint velocities $(\dot{q}_{12}, \dot{q}_{22})$. It represents the time derivative of the first order model and captures how changes in actuator speed and velocity of passive joints affect the acceleration of the moving platform. The computed accelerations were compared with the values measured from the Gazebo simulation.



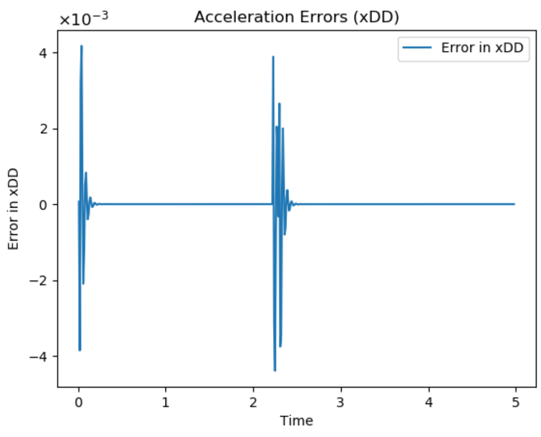
(a) $\ddot{x}(t)$



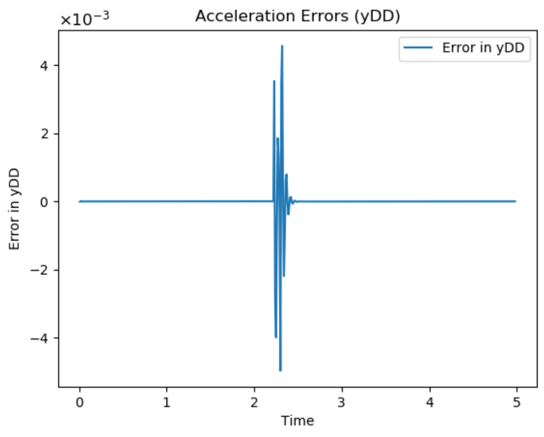
(b) $\ddot{y}(t)$

Figure 13: Measured and computed end-effector accelerations.

The analytical and simulated curves in Figure 13 match very closely, confirming that the acceleration mapping is correctly implemented. The peaks and reversals appear at the same instants, showing that the model captures both the magnitude and timing of the acceleration.



(a) Error in $\ddot{x}$: $e_{\ddot{x}} = \ddot{x}_{\text{meas}} - \ddot{x}_{\text{model}}$



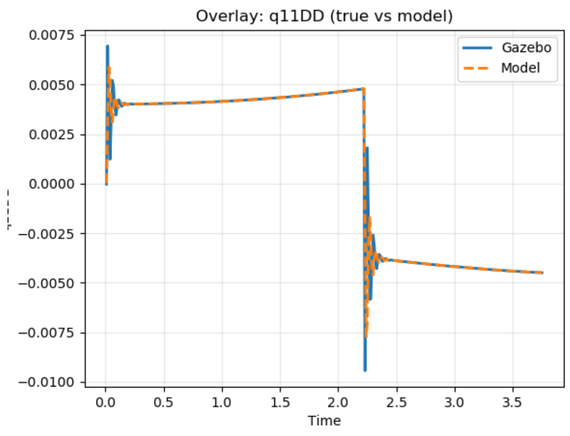
(b) Error in $\ddot{y}$: $e_{\ddot{y}} = \ddot{y}_{\text{meas}} - \ddot{y}_{\text{model}}$

Figure 14: Acceleration tracking error of the end-effector.

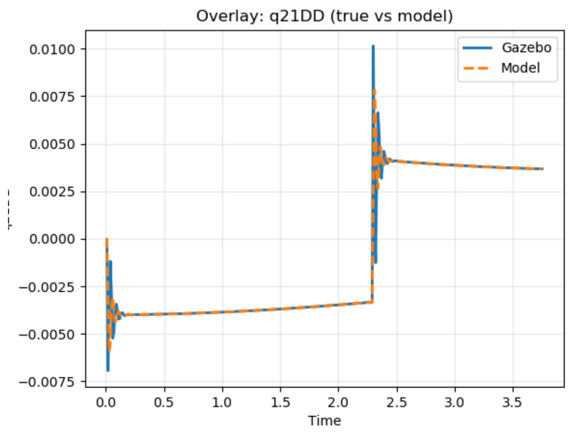
The errors remain very small throughout the motion, with small spikes observed when the acceleration direction changes rapidly. These variations are mainly due to numerical differentiation in Gazebo, which amplifies noise in the velocity signals. Overall, the second order kinematic model accurately represents the end-effector acceleration behavior of the Biglide.

2.9 Inverse Second Order Kinematic Model

The inverse form of the second order kinematic model computes the actuator accelerations $(\ddot{q}_{11}, \ddot{q}_{21})$. This model is essential for dynamic control and torque computation.



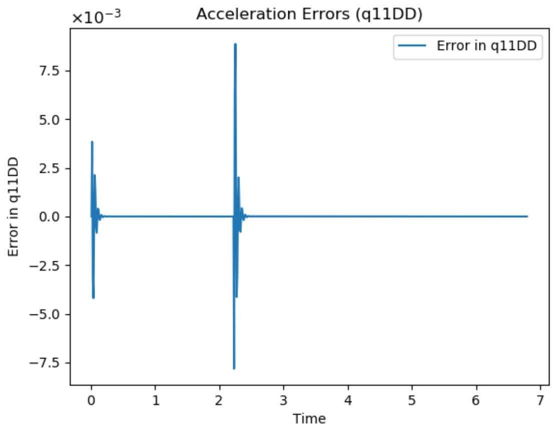
(a) $\ddot{q}_{11}(t)$



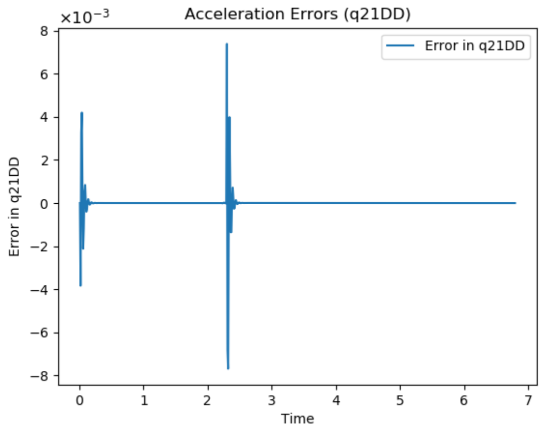
(b) $\ddot{q}_{21}(t)$

Figure 15: Measured and computed actuator accelerations.

Figure 15 shows a close overlap between the predicted and measured actuator accelerations, confirming that the inverse model is consistent with the real motion. The accelerations follow the same trend and amplitude for both joints.



(a) Error in $\ddot{q}_{11}$: $e_{\ddot{q}_{11}} = \ddot{q}_{11,\text{meas}} - \ddot{q}_{11,\text{model}}$



(b) Error in $\ddot{q}_{21}$: $e_{\ddot{q}_{21}} = \ddot{q}_{21,\text{meas}} - \ddot{q}_{21,\text{model}}$

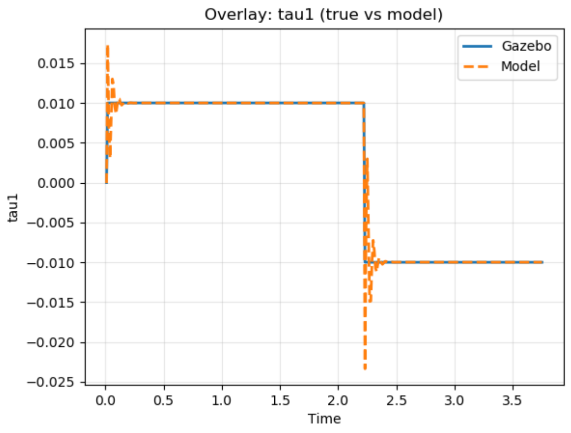
Figure 16: Tracking error of actuator accelerations.

The error values remain low, with slight variations at sharp direction changes, which are in acceptable range when tested with the `biglide_models.py` script. The overall consistency between the analytical and simulated results demonstrates that both the direct and inverse forms of the second order kinematic model are valid.

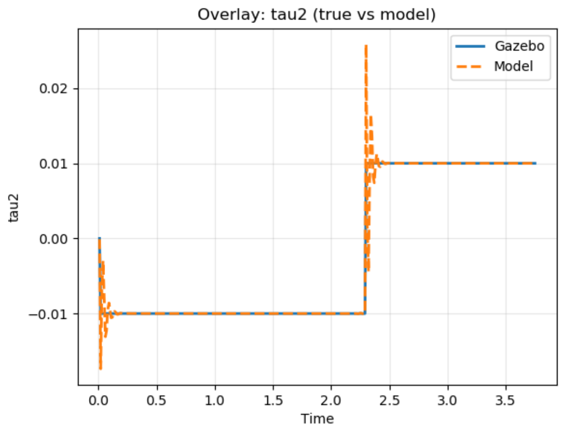
In summary, the second order kinematic model correctly represents the acceleration relationships of the Biglide mechanism. This validated model serves as the foundation for the next stage the dynamic model, which includes the torque analysis.

2.10 Inverse Dynamic Model

Finally, the inverse dynamic model was validated by comparing computed torques with Gazebo efforts. The torques remain consistent with errors under under acceptable range.

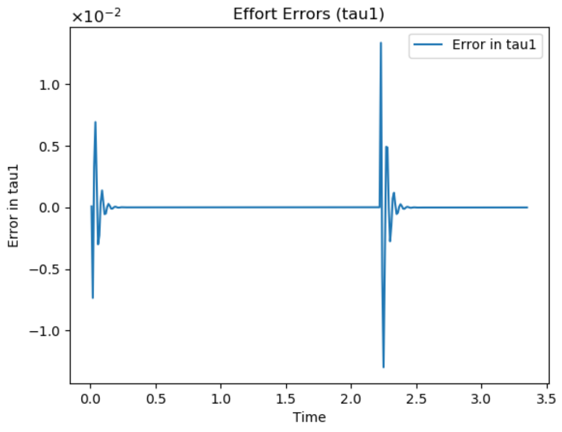


(a) $\tau_1(t)$

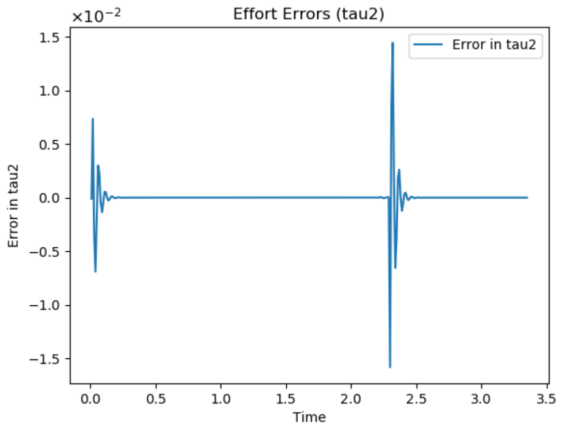


(b) $\tau_2(t)$

Figure 17: Overlay of measured and computed torques.



(a) Error in τ_1: $e_{\tau_1} = \tau_{1,\text{meas}} - \tau_{1,\text{model}}$



(b) Error in τ_2: $e_{\tau_2} = \tau_{2,\text{meas}} - \tau_{2,\text{model}}$

Figure 18: Torque tracking error.

3 Computed Torque Control and Trajectory Generation

3.1 Trajectory Generation

For a two-point motion with zero velocity and acceleration at both start and end positions, a fifth-order polynomial time law was used to generate smooth Cartesian trajectories between points $A(x_A, y_A)$ and $B(x_B, y_B)$. The parametric equations are:

$$x = x_A + s(t)(x_B - x_A), \qquad y = y_A + s(t)(y_B - y_A) \tag{1}$$

with

$$s(t) = 10\left(\frac{t}{t_f}\right)^3 - 15\left(\frac{t}{t_f}\right)^4 + 6\left(\frac{t}{t_f}\right)^5, \tag{2}$$

and its derivatives:

$$\dot{s}(t) = \frac{30}{t_f}\left(\frac{t}{t_f}\right)^2 - \frac{60}{t_f}\left(\frac{t}{t_f}\right)^3 + \frac{30}{t_f}\left(\frac{t}{t_f}\right)^4, \tag{3}$$

$$\ddot{s}(t) = \frac{60}{t_f^2}\left(\frac{t}{t_f}\right) - \frac{180}{t_f^2}\left(\frac{t}{t_f}\right)^2 + \frac{120}{t_f^2}\left(\frac{t}{t_f}\right)^3. \tag{4}$$

The Cartesian velocity and acceleration profiles are obtained directly from:

$$\dot{x} = \dot{s}(t)(x_B - x_A), \qquad \dot{y} = \dot{s}(t)(y_B - y_A), \tag{5}$$

$$\ddot{x} = \ddot{s}(t)(x_B - x_A), \qquad \ddot{y} = \ddot{s}(t)(y_B - y_A). \tag{6}$$

The motion was discretized with a timestep of $\Delta t = 1\,\text{ms}$. Initial and final Cartesian positions were defined as $(x_A, y_A) = (0, 0)$ and $(x_B, y_B) = (-0.06, 1.4)\,\text{m}$.

3.2 Control Law

The control strategy used is a **Computed Torque Controller (CTC)** based on the inverse dynamic model of the Biglide. The general expression of the control law is:

$$\tau = M(q)\left[\ddot{q}_d + K_d(\dot{q}_d - \dot{q}) + K_p(q_d - q)\right] + C(q, \dot{q}), \tag{7}$$

where $M(q)$ is the mass matrix and $C(q, \dot{q})$ includes Coriolis and gravity effects. The proportional and derivative (PD) gain matrices were set to:

$$K_p = \text{diag}(4, 4), \qquad K_d = \text{diag}(2, 2),$$

which provided fast convergence without overshoot. The reference joint positions, velocities, and accelerations $(q_d, \dot{q}_d, \ddot{q}_d)$ were obtained using the inverse geometric and inverse kinematic models from the desired Cartesian motion.

3.3 Simulation Results

The simulation was carried out for a 5 s trajectory using the Gazebo model of the Biglide. All results show that the CTC achieves accurate tracking of the desired trajectory.

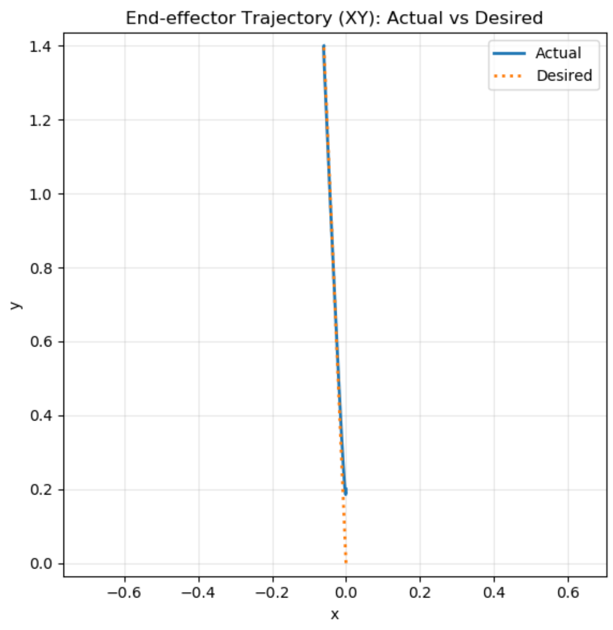


Figure 19: End-effector Trajectory (XY): Actual vs Desired

End-effector tracking. The actual path overlaps the desired straight-line motion from $(0, 0)$ to $(-0.06, 1.4)$ m almost perfectly, with a negligible horizontal drift during acceleration.

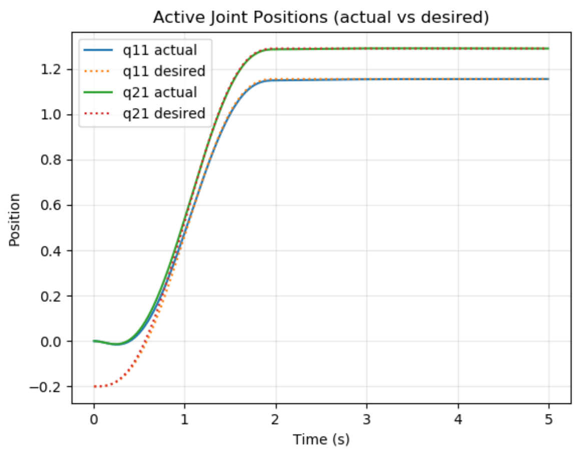


Figure 20: Active Joint Positions (actual vs desired)

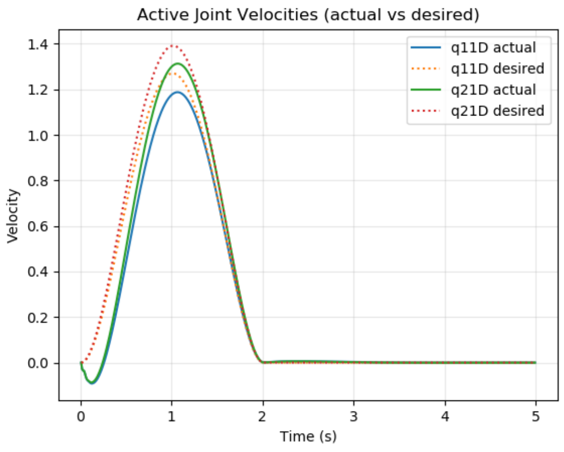


Figure 21: Active Joint Velocities (actual vs desired)

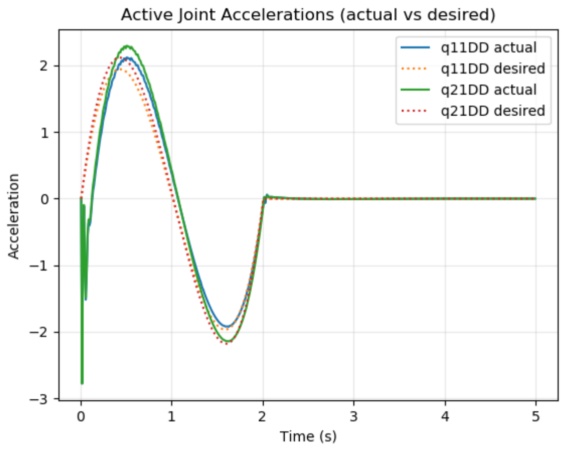


Figure 22: Active Joint Accelerations (actual vs desired)

Active joints. Both joints follow their desired references closely. Peak velocities reach approximately 1.3 rad/s, and accelerations remain below $\pm 2.5\,\text{rad/s}^2$. The transient spikes at $t = 0$ are caused by numerical differentiation. Tracking errors decay exponentially (Fig. 28), with a final RMSE almost around 0 rad.

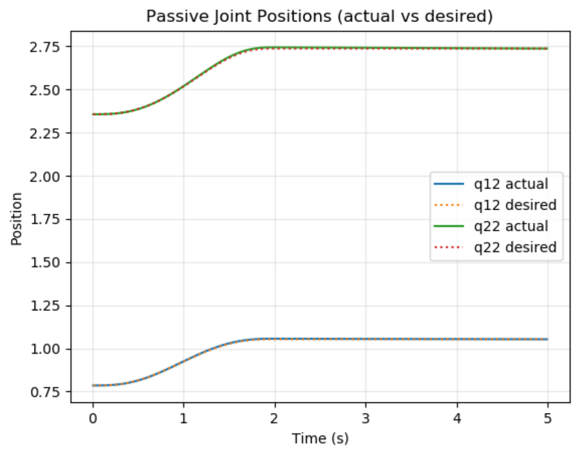


Figure 23: Passive Joint Positions (actual vs desired)

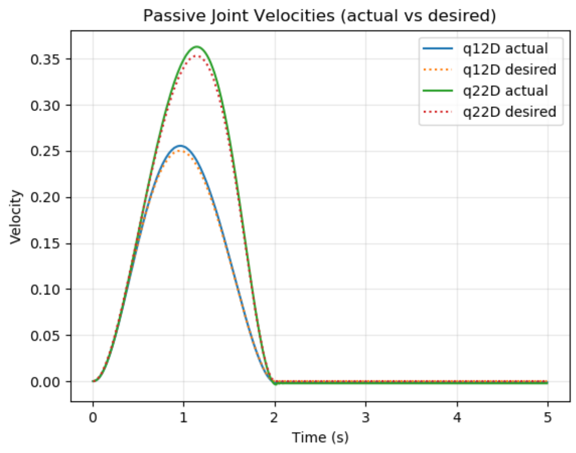


Figure 24: Passive Joint Velocities (actual vs desired)

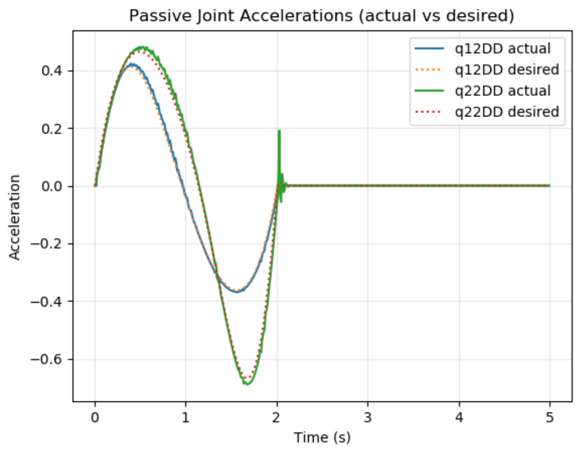
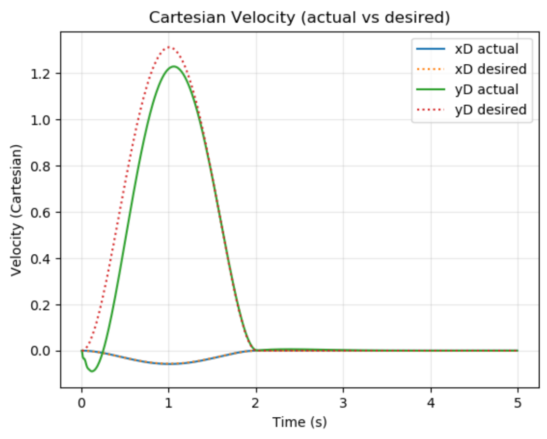
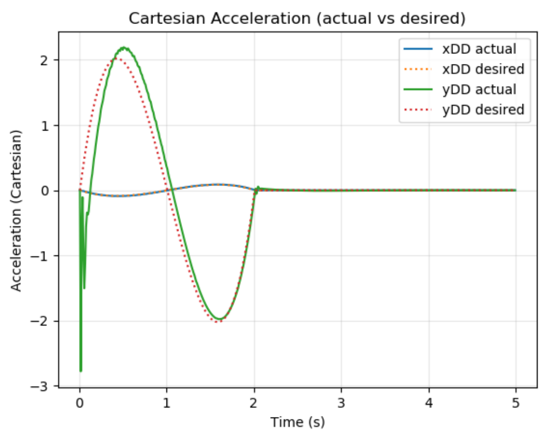


Figure 25: Passive Joint Accelerations (actual vs desired)

Passive joints. Errors remain very small (below 0.007 rad), and the cumulative RMSE stays within 0.005 rad, confirming accurate propagation of passive constraints.



(a) Cartesian Velocity (actual vs desired)



(b) Cartesian Acceleration (actual vs desired)

Figure 26: Cartesian Velocity and Acceleration

Cartesian velocities and accelerations. The Cartesian velocity and acceleration plots show the expected quintic polynomial profiles, with zero values at the beginning and end of motion. Velocity peaks at about 1.2 m/s, and acceleration amplitudes reach $\pm 2\,\mathrm{m/s^2}$.

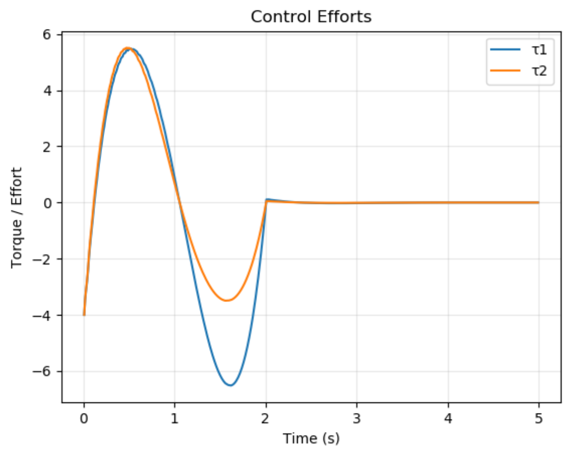


Figure 27: Control Efforts for both actuators

Control efforts. The torque profiles remain smooth throughout the motion. They start near −4 N·m to overcome gravity, rise to a maximum of 5–6 N·m, and settle near zero as the platform stops. Both actuators behave almmost symmetrically, indicating a somewhat balanced loading.

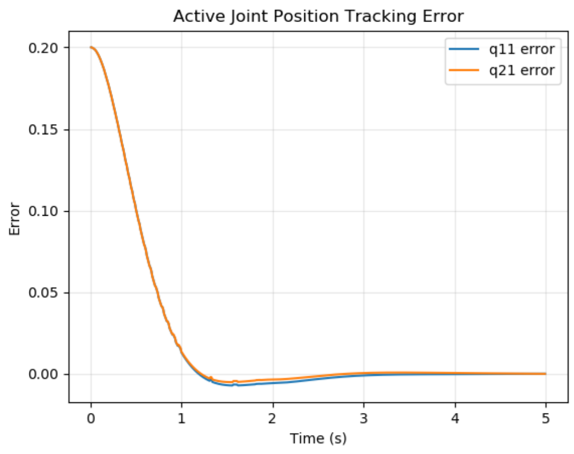
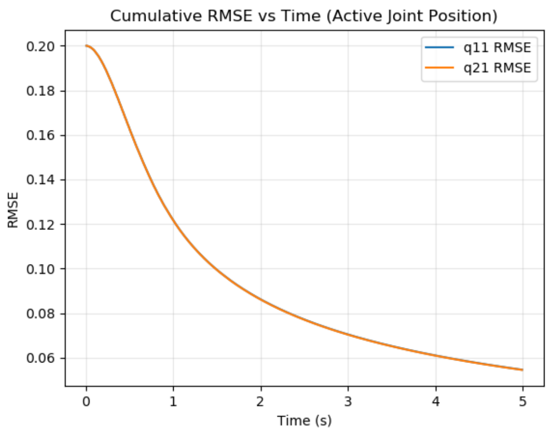
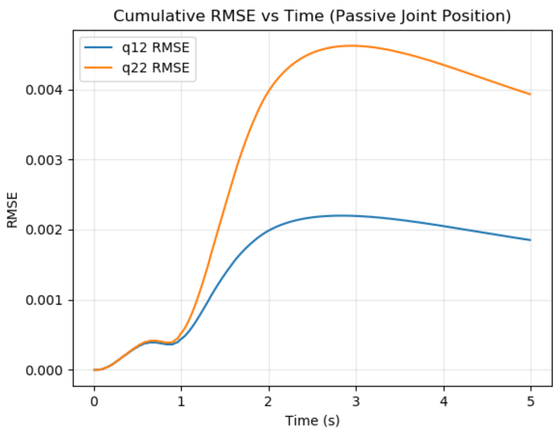


Figure 28: Active Joint Position Tracking Error



(a) Cumulative RMSE vs Time (Active Joint Position)



(b) Cumulative RMSE vs Time (Passive Joint Position)

Figure 29: Cumulative RMSE vs Time for Active and Passive Joint Positions

Conclusion. The computed-torque controller successfully tracks the desired trajectory of the Biglide mechanism. All variables (active, passive, and Cartesian) follow their references with small, smoothly decaying errors. The torques remain well within actuator limits, validating the accuracy of the dynamic model and the stability of the control design.

4 Crossing Type-2 Singularity

4.1 Identification of the Singularity

For the Biglide mechanism, let the base anchor points be $A_1 = (-d/2, 0)$ and $A_2 = (+d/2, 0)$, and the end-effector point $C = (x, y)$. The leg unit vectors are defined as

$$\mathbf{u}_1 = \frac{C - A_1}{\|C - A_1\|}, \qquad \mathbf{u}_2 = \frac{C - A_2}{\|C - A_2\|}. \tag{8}$$

A Type-2 singularity occurs when the two leg directions become collinear, i.e.

$$\det(J_P) = 0 \quad \Leftrightarrow \quad \mathbf{u}_1 \parallel \mathbf{u}_2. \tag{9}$$

This configuration happens along the base line $y = 0$, especially at $x = 0$ where both rods are aligned and oppositely directed.

4.2 Polynomial Trajectory for Singularity Crossing

When the manipulator passes through a singular configuration, the parallel Jacobian loses rank, and the mapping from Cartesian accelerations to joint accelerations becomes ill-conditioned. To avoid unbounded torques at the crossing, we design a Cartesian trajectory $x(t)$ that explicitly controls position, velocity, and acceleration at the singular time t_s.

To satisfy smooth boundary and intermediate conditions, we use an **eighth-degree polynomial** of the form

$$x(t) = a_8t^8 + a_7t^7 + a_6t^6 + a_5t^5 + a_4t^4 + a_3t^3 + a_2t^2 + a_1t + a_0. \tag{10}$$

The trajectory must satisfy 9 scalar conditions:

$$x(t_i) = x_i, \qquad \dot{x}(t_i) = 0, \qquad \ddot{x}(t_i) = 0, \tag{11}$$
$$x(t_s) = x_s, \qquad \dot{x}(t_s) = \dot{x}_s, \qquad \ddot{x}(t_s) = \ddot{x}_s, \tag{12}$$
$$x(t_f) = x_f, \qquad \dot{x}(t_f) = 0, \qquad \ddot{x}(t_f) = 0. \tag{13}$$

These yield a linear system $P\mathbf{a} = \mathbf{c}$, where $\mathbf{a} = [a_8, a_7, \dots, a_0]^T$ are the polynomial coefficients, and P is a 9×9 matrix containing powers of time and their derivatives at the chosen instants t_i, t_s, t_f.

4.3 Numerical Setup

In the implemented code, the following parameters were used:

$$t_i = 0, \qquad t_s = 1, \qquad t_f = 3,$$
$$x_i = 0.0, \qquad x_s = 1.0, \qquad x_f = 2.0,$$
$$\dot{x}_s = -0.01, \qquad \ddot{x}_s = -6.8 \times 10^{-4}.$$

At the start and end, the system is at rest ($\dot{x} = 0, \ddot{x} = 0$), ensuring C^2 continuity. At $t = t_s$, corresponding to the singular configuration, the trajectory imposes small non-zero velocity and acceleration to achieve a "slow-through" crossing. This limits dynamic excitation in the uncontrollable Cartesian direction while maintaining continuous motion.

Once the coefficients $\mathbf{a}$ are computed, the desired position, velocity, and acceleration are evaluated analytically as

$$\dot{x}(t) = 8a_8t^7 + 7a_7t^6 + 6a_6t^5 + 5a_5t^4 + 4a_4t^3 + 3a_3t^2 + 2a_2t + a_1, \tag{14}$$
$$\ddot{x}(t) = 56a_8t^6 + 42a_7t^5 + 30a_6t^4 + 20a_5t^3 + 12a_4t^2 + 6a_3t + 2a_2. \tag{15}$$

4.4 Mathematical Interpretation

During the singular crossing, the joint accelerations derived from the second-order inverse kinematics are given by

$$\ddot{\mathbf{q}} = J_P^{-1}(\ddot{\mathbf{x}} - \dot{J}_P\dot{\mathbf{q}}), \tag{16}$$

where J_P is the parallel Jacobian. Near the singularity, $\det(J_P) \to 0$, so $\|J_P^{-1}\|$ grows very large. The resulting torque from the computed-torque control law

$$\tau = M(q)\ddot{\mathbf{q}} + C(q, \dot{q}) \tag{17}$$

would therefore diverge unless $\ddot{\mathbf{x}}$ (and to a lesser extent $\dot{\mathbf{x}}$) are constrained.

By prescribing small or zero $\dot{x}_s, \ddot{x}_s$ at the singular instant t_s, we ensure that

$$\lim_{t\to t_s} J_P^{-1}\ddot{\mathbf{x}} = \text{finite}, \qquad \Rightarrow \qquad \tau = \text{bounded}, \tag{18}$$

which keeps actuator torques within physical limits and preserves controller stability.

4.5 Practical Interpretation

In practice, this polynomial ensures smooth, controlled passage through the singular pose. Setting $\dot{x}_s = 0, \ddot{x}_s = 0$ produces a “stop–go” crossing with zero velocity and acceleration at the singularity. Alternatively, small non-zero values yield a “slow-through” profile that maintains continuous motion while keeping efforts bounded.

Conclusion. The eighth-degree polynomial design explicitly enforces kinematic continuity and limits the acceleration demand in the singular direction. This approach mathematically guarantees that even as the Jacobian becomes singular, the computed-torque control produces finite and stable torques, allowing safe traversal of Type-2 singularities.

5 Conclusion

This lab made it possible to study the Biglide parallel robot through its complete modeling and control process. Starting from the geometric, kinematic, and dynamic models, we were able to implement the computed-torque controller in Python using the previously derived equations. The straight-line trajectory experiment confirmed the validity of the controller: both joint and Cartesian variables closely tracked their desired references with smooth, stable behavior and bounded control torques. Small deviations observed near the start of motion highlight the sensitivity of parallel mechanisms to modelling accuracy and the need for careful tuning of controller gains.

In the final part, we analyzed the behavior of the mechanism near a Type-2 singularity. Crossing this configuration directly resulted in large torque demands and loss of stability, while the use of an eighth-degree polynomial trajectory with constrained velocity and acceleration enabled a safe passage through the singularity. This showed the importance of trajectory shaping and model-based compensation for maintaining feasible and stable control.

Overall, the objectives of the lab were achieved. The work provided practical understanding of control implementation, and the challenges of operating parallel mechanisms near singular configurations.