Advanced Control Systems Project

Master's degree in Computer Engineering for Robotics and Smart Industry

ACADEMIC YEAR: 2021-2022

AUTHOR'S: Antonino Parisi

October 2021

Summary

1 Robot Kinematics

In figure 1 is shown the robot which is a RPR(Rotative-Prismatic-Rotative) robot.

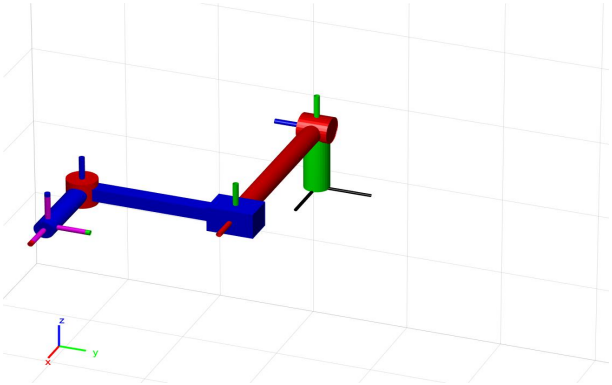

Figure 1: Robot visualization with Matlab toolbox

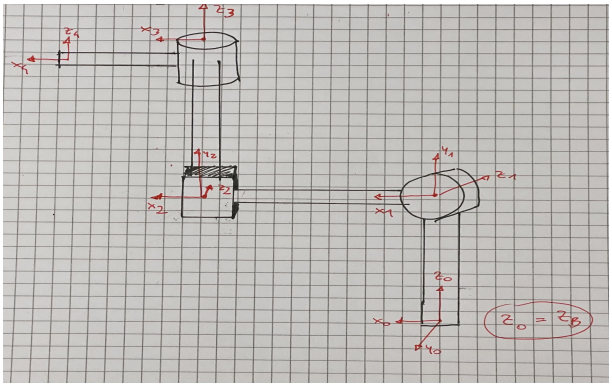

Figure 2: Robot frames

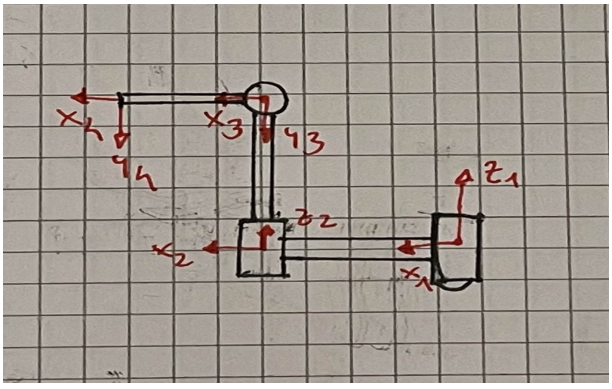

Figure 3: Robot frames - top view

1.1 Denavit Hartenberg Table

The reference frame attached to each joint follows the Denavit Hartenberg convention, as it is shown by the table 1. The first and last line are only constant for the base reference frame and for the end-effector.

	a_i	α_i	d_i	θ_i
0-1	0	$\pi/2$	a1	0
1-2	a_2	0	0	θ_1^*
2-3	0	$-\pi/2$	a3+d1*	0
3-4	0	0	a3	θ_2^*

Table 1: Denavit Hartenberg Table of RPR robot

1.2 Direct Kinematics

$$T_0^2 = \begin{bmatrix} cos(\theta_1) & -sin(\theta_1) & 0 & a2 * cos(\theta_1) \\ 0 & 0 & -1 & 0 \\ sin(\theta_1) & cos(\theta_1) & 0 & a1 + sin(\theta_1) \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1}$$

$$T_0^3 = \begin{bmatrix} cos(\theta_1) & 0 & -sin(\theta_1) & a2 * cos(\theta_1) \\ 0 & 1 & 0 & -a3 - d1 \\ sin(\theta_1) & 0 & cos(\theta_1) & a1 + a2 * sin(\theta_1) \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2}$$

$$T_0^4 = \begin{bmatrix} cos(\theta_1)*\cos(\theta_2) & -cos(\theta_1)*\sin(\theta_2) & -sin(\theta_1) & a2 * cos(\theta_1) + a4*\cos(\theta_1)*\cos(\theta_2) \\ sin(\theta_1) & cos(\theta_2) & 0 & a4 * sin(\theta_2) \text{ -d1 -a3} \\ sin(\theta_1)*\cos(\theta_1) & -sin(\theta_1)*\sin(\theta_2) & cos(\theta_1) & a1 + a2 * sin(\theta_1) + a4*\cos(\theta_2)*\sin(\theta_1) \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3}$$

1.3 Inverse Kinematics

Here we apply squaring and summing to find a closed form to obtain all joint variables through cartesian coordinates.

The values a_1, a_2, a_3, a_4 are the link lenghts, θ_1, θ_2, d_1 are the joint variables, pwx, pwy, pwz are the coordinates of EE;

1.3.1 Theta 2

$$pwx^2 + (pwz - a1)^2 = (cos(\theta_1)(a2 + a4cos(\theta_2))^2 + (sin(\theta_1)(a2 + a4cos(\theta_2))^2$$

$$cos(\theta_2) = \frac{\sqrt{pwx^2 + (pwz - a1)^2} - a2}{a4}$$

$$sin(\theta_2) = \pm\sqrt{1 - cos(\theta_1)^2}$$

$$\theta_2 = atan2(sin(\theta_2), cos(\theta_2))$$

1.3.2 D 1

$$pwy = a4 * sin(t2) - d1 - a3$$

$$d1 = a4 * sin(t2) - a3 - pwy$$

1.3.3 Theta 1

$pwx = a2 * cos(\theta_1) + a4 * cos(\theta_1) * cos(\theta_2)$
$pwz = a1 + a2 * sin(\theta_1) + a4 * cos(\theta_2) * sin(\theta_1)$
$pwx^2 + pwy^2 = (a2 * cos(\theta_1) + a4 * cos(\theta_1) * cos(\theta_2))^2 + (a1 + a2 * sin(\theta_1) + a4 * cos(\theta_2) * sin(\theta_1))^2$
$sin(\theta_1) = -\frac{(a4^2 * cos(\theta_2)^2 + a1^2 + a2^2 + 2 * a2 * a4 * cos(\theta_2) - pwx^2 - pwz^2)}{(2 * a1 * a2 + 2 * a1 * a4 * cos(\theta_2))}$
$cos(\theta_1) = \pm\sqrt{1 - sin(\theta_1)^2}$
$\theta_1 = atan2(sin(\theta_1), cos(\theta_1))$

1.4 Jacobians

1.4.1 Geometric Jacobian

$$\mathrm{J} = \begin{bmatrix} z1 \times (p4 - p1) & z2 & z3 \times (p4 - p3) \\ z1 & 0 & z3 \end{bmatrix} = \begin{bmatrix} a4 * sin(\theta_2) \text{ - d1 - a3} & 0 & a4 * cos(\theta_1) * \sin(\theta_2) \\ -a2 * cos(\theta_1) \text{ - a4} * \cos(\theta_1) * \cos(\theta_2) & -1 & -a4 * cos(\theta_2) \\ 0 & 0 & a4 * sin(\theta_1) * \sin(\theta_2) \\ 0 & 0 & -sin(\theta_1) \\ 0 & 0 & 0 \\ 1 & 0 & cos(\theta_1) \end{bmatrix} \tag{4}$$

1.4.2 Analytical Jacobian

To compute the analytical jacobian is exploited the notation of siciliano's text book.

$$J_a(q) = \frac{\partial k(q)}{\partial q} \tag{5}$$

the method to compute it is reported in the following steps

- Extract the rotation matrix from kinematics
- compute ϕ, θ, ψ as is reported in the book
- $\phi = atan2(r_{2,3}, r_{1,3})$
- $\theta = atan2(\sqrt{r_{1,3}^2 + r_{2,3}^2}, r_{3,3})$
- $\psi = atan2(r_{3,2}, -r_{3,1})$
- then compute the partial derivative for each angle in base of the actual q.

this is the analytical jacobian obtained

$$\begin{bmatrix} -a2 * sin(\theta_1) \text{ - a4*cos}(\theta_2)\text{*sin}(\theta_1) & 0 & -a4 * cos(\theta_1)\text{*sin}(\theta_2) \\ 0 & -1 & a4 * cos(\theta_2) \\ a2 * cos(\theta_1) \text{ + a4*cos}(\theta_1)\text{*cos}(\theta_2) & 0 & -a4 * sin(\theta_1)\text{*sin}(\theta_2) \\ 0 & 0 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (6)$$

the analitic jacobian is the same if we compute it with the alternative way,

$$J_a = T(\phi)^{-1} J(q) \quad (7)$$

2 Lagrangian approach

The full equation of Lagrangian approach is describes as follows:

$$B(q)\ddot{q} + C(q, \dot{q})\dot{q} + G(q) = \tau \quad (8)$$

and in the following section is explained how to compute the matrices B,C and G.

2.1 B matrix

The B matrix is inertia matrix, has some properties:

- is symmetric definite(all eigs are positive)
- configuration dependent
- bii (q) is the moment of inertia at Joint i axis when the other joints are blocked
- $bii(q) = bii > 0$
- bij is the effect of acceleration of Joint j on Joint i

to compute the matrix B we have to compute all the partial Jacobians and shift the inertia of i-th link to the edge of the link.

$$PL_1 = \begin{bmatrix} -a2s_1 & 0 & 0 \\ 0 & 0 & 0 \\ a2c_1 & 0 & 0 \end{bmatrix} \tag{9}$$

$$PL_2 = \begin{bmatrix} -a2s_1 & 0 & 0 \\ 0 & -1 & 0 \\ a2c_1 & 0 & 0 \end{bmatrix} \tag{10}$$

$$PL_3 = \begin{bmatrix} a4s_2 - d1 - a3 & 0 & 0 \\ -a2 * c_1 - a4c_1c_2 & -1 & 0 \\ a4c_1s_2 & -a4c_2c_1^2 - a4c_2s_1^2 & a4s_1s_2 \end{bmatrix} \tag{11}$$

$$WL_1 = \begin{bmatrix} 0 & 0 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{12}$$

$$WL_2 = \begin{bmatrix} 0 & 0 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{13}$$

$$WL_3 = \begin{bmatrix} 0 & 0 & -sin(t1) \\ -1 & 0 & 0 \\ 0 & 0 & cos(t1) \end{bmatrix} \tag{14}$$

$$\tag{15}$$

after the computation of the partial Jacobians is mandatory to compute also the kinetic energy related to the link bodies which the robot is composed of. The kinetic energy of a rectangle can be expressed as follows:

$$I_{Ci} = \begin{bmatrix} \frac{1}{12}m(b^2 + c^2) & 0 & 0 \\ 0 & \frac{1}{12}m(a^2 + c^2) & 0 \\ 0 & 0 & \frac{1}{12}m(a^2 + b^2) \end{bmatrix} \tag{16}$$

and the same for the hollow cylinder:

$$I_{Ci} = \begin{bmatrix} \frac{1}{2}m(b^2 + c^2) & 0 & 0 \\ 0 & \frac{1}{2}m(3(a^2 + c^2) + h^2) & 0 \\ 0 & 0 & \frac{1}{2}m(3(a^2 + c^2) + h^2) \end{bmatrix} \tag{17}$$

and then shift the kinetic energy to the edge of his geometry limits by the Steiner Theorem:

Theorem 1 *Let I_C be the inertia matrix with respect to a reference frame Σ with origin on the center of mass. The inertia I with respect to another reference frame obtained translating C by the vector $r \in$ is given by:*
$I = I_C + m(r^T rI - rr^T)$

now we can compute the B matrix and also the value of kinetic energy.

$$\frac{1}{2}\dot{q}^T \left[\sum_{i=1}^{n} (M_{li}(J_P^{li})^T J_P^{li} + J_O^{li} R_i I_{li}^i R_i^T J_O^{li} \right] \dot{q} \tag{18}$$

$$\frac{1}{2}\dot{q}^T B(q)\dot{q} \tag{19}$$

Some remarks:

- $\tau \geq 0$
- $\tau = 0 \iff \dot{q} = 0$

2.2 C matrix

The computation of C matrix is related to B matrix; the C matrix is related to centrifugal and Coriolis effects. To compute these terms we use Christoffel's coefficents, the terms inside the C matrix are reported as h_{ijk} because of the Chr. coefficients.

Remarks:

- the general formula for joint i is

$$C_i = \sum_{j=1}^{n} \sum_{k=1}^{n} h_{i,j,k} \dot{q}_k \dot{q}_j \tag{20}$$

- $h_{i,j,j}\dot{q}_j^2$ is centrifugal effect induced on Joint i by velocity of Joint j
- $h_{i,j,k}\dot{q}_j\dot{q}_k$ is Coriolis effect induced on Joint i by velocities of Joints j and k
- we can rewrite the previous equation as follows:

$$C_i = \sum_{j=1}^{n} c(q, \dot{q})\dot{q}_j \tag{21}$$

  because of the Christoffel symbols of the first type

- the choice of c_{ij} is not unique

to compute the C matrix we consider this following formula:

$$C_{i,j} = \sum_{j=1}^{n} \sum_{k=1}^{n} \frac{1}{2} \left(\frac{\partial b_{i,j}(q)}{\partial q_k} \dot{q}_k \dot{q}_j + \frac{\partial b_{i,k}(q)}{\partial q_j} - \frac{\partial b_{j,k}(q)}{\partial q_i} \right) \dot{q}_k \dot{q}_j \tag{22}$$

to clarify how to compute these two different derivatives we explain it,

- the $b_{i,j}$, $b_{i,k}$ and $b_{j,k}$ is referring to the (i,j), (i,k) or (j,k) element of the B matrix
- the q_k,q_i and q_j at the denominator is referring to the joint variable, in this case

$$(q_k, q_j, q_i) \in [\theta_1, d_1, \theta_2] \tag{23}$$

- in the end the computation of $c_{i,j}$ is just a matter of derivative computation with the correct joint variable

2.3 G matrix

The G matrix is obtained by differentiating the potential energy by each joint variable, here is reported the formulas:

$$G = -\sum_{j=1}^{n} m_{lj} g_0^T J_{pi}^{lj}(q) \tag{24}$$

3 Recursive Newton-Eulero

The newton-eulero is recursive formulation for the dynamic model of a robot, to compute each torque and moment we have two phases : forward and backward.

3.1 Legend of words

Collection of used symbols.

- m_i is the mass of augmented link i
- I_i is the inertia tensor of augmented link
- I_{mi} is the moment of inertia of the rotor i
- $r_{i1,Ci}$ is the vector from origin of frame (i 1) to centre of mass C_i
- r_{i,C_i} vector from origin of frame i to centre of mass C_i
- $r_{i-1,i}$ vector from origin of frame (i-1) to origin frame i
- $\dot{p}_{C_i}$ linear velocity of centre of mass Ci
- $\dot{p}_i$ linear velocity of origin of Frame i
- ω_i angular velocity of link i
- ω_{m_i} angular velocity of rotor
- $\ddot{p}_{C_i}$ linear acceleration of centre of mass C_i
- $\ddot{p}_i$ linear acceleration of origin of Frame i

- $\dot{\omega}_i$ angular acceleration of link i
- $\dot{\omega}_{m_i}$ angular acceleration of rotor i
- g_0 gravity acceleration
- f_i force exerted by Link i 1 on Link i
- $-f_{i+1}$ force exerted by Link i + 1 on Link i
- μ_i moment exerted by Link i 1 on Link i with respect to origin of Frame (i 1)
- $-\mu_{i+1}$ moment exerted by Link i + 1 on Link i with respect to origin of Frame i

equations The Newton-Eulero formulation consider the translational motion of the CoM and the rotational motion taking into account forces and moments. The linear forces are considered in the Newton formulas

$$f_i - f_{i+1} + m_i g_0 = m_i \ddot{p}_{C_i} \tag{25}$$

and torques are considered in Euler formulas

$$\mu_i + f_i x + r_{i-1 g_0 = m_i \ddot{p}_{C_i}} \tag{26}$$

3.2 Forward equations

The forward equations are needed to compute $\omega_i^i, \dot{\omega}_i^i, \ddot{p}_i^i, \ddot{p}_{C_i}^i, \dot{\omega}_{m_i}^{i-1}$;

$$w_i^i = \begin{cases} (\mathrm{R}_i^{i-1})^T \ \omega_{i-1}^{i-1} & \text{prismatic} \\ (\mathrm{R}_i^{i-1})^T \ (\omega_{i-1}^{i-1} + \dot{\theta}_i z_0) & \text{rotational} \end{cases} \tag{27}$$

$$\dot{w}_i^i = \begin{cases} (\mathrm{R}_i^{i-1})^T \ \omega_{i-1}^{i-1} & \text{prismatic} \\ (\mathrm{R}_i^{i-1})^T \ (\dot{\omega}_{i-1}^{i-1} + \ddot{\theta}_i z_0) + \dot{\theta}_i \omega_{i-1}^{i-1} x z 0 & \text{rotational} \end{cases} \tag{28}$$

$$\ddot{p}_i^i = \begin{cases} (\mathrm{R}_i^{i-1})^T \ (\ddot{p}_{i-1}^{i-1} + \ddot{d}_i z 0) + 2\dot{d}_i w_i^i \times ((R_i^{i-1})^T z 0) + \dot{\omega}_i^i \times r_{i-1,i}^i + \omega_i^i \times (\omega_i^i \times r_{i-1,i}^i) & \text{prismatic} \\ (\mathrm{R}_i^{i-1})^T \ \ddot{p}_{i-1}^{i-1} + \dot{\omega}_i^i \times r_{i-1,i}^i + \omega_i^i \times (\omega_i^i \times r_{i-1,i}^i) & \text{rotational} \end{cases} \tag{29}$$

$$\ddot{p}_{C_i}^i = \ddot{p}_i^i + \dot{\omega}_i^i \times r_{i,C_i}^i + \omega_i^i \times (\omega_i^i \times r_{i,C_i}^i) \tag{30}$$

$$\dot{w}_i^i = \dot{\omega}_{i-1}^{i-1} + k_{ri} \ddot{q}_i z_{m_i}^{i-1} + k_{ri} \dot{q}_i \dot{\omega}_{i-1}^{i-1} \times z_{m_i}^{i-1} \tag{31}$$

3.3 Backward equations

The backward equations are needed actually to compute the torques starting from the last joint where is applied an external wrench H_e, and starting from it we compute all the joint forces and moments (f_i^i, μ_i^i), that are mandatory to compute τ_i.

$$f_i^i = R_{i+1}^i f_{i+1i+1} + m_i \ddot{p}_{C_i}^i \tag{32}$$

$$\dot{\mu}_i^i = -f_i^i \times (r_{i-1,i}^i + r_{i,C_i}^i) + R_{i+1}^i \mu_{i+1i+1} + R_{i+1}^i f_{i+1i+1} \times r_{i,C_i}^i + \bar{I}_i^i \dot{\omega}_i^i + \omega_i^i \times (\bar{I}_i^i \omega_i^i) + k_{r,i} \ddot{q}_{i+1} I_{m+1} z_{m+1}^i + k_{r,i} \dot{q}_{i+1} I_{m+1} \omega_i^i \times z_{m+1}^i \tag{33}$$

$$\ddot{p}_i^i = \begin{cases} (\mathrm{f}_i^i)^T (R_i^{i-1})^T z_0 + k_{r,i} I_m (\dot{\omega}_{m_i}^{i-1})^T z_{m_i}^{i-1} + F_{vi} \dot{d}_i + F_{si} sign(\dot{d}_i) & \text{prismatic} \\ (\mu_i^i)^T (R_i^{i-1})^T z_0 + k_{r,i} I_m (\dot{\omega}_{m_i}^{i-1})^T z_{m_i}^{i-1} + F_{vi} \dot{d}_i + F_{si} sign(\ddot{\theta}_i) & \text{rotational} \end{cases} \tag{34}$$

4 Control section

4.1 PD controller with gravity compensation

Params:

- Kp : eye(3) * 50
- Kd : eye(3) * 4

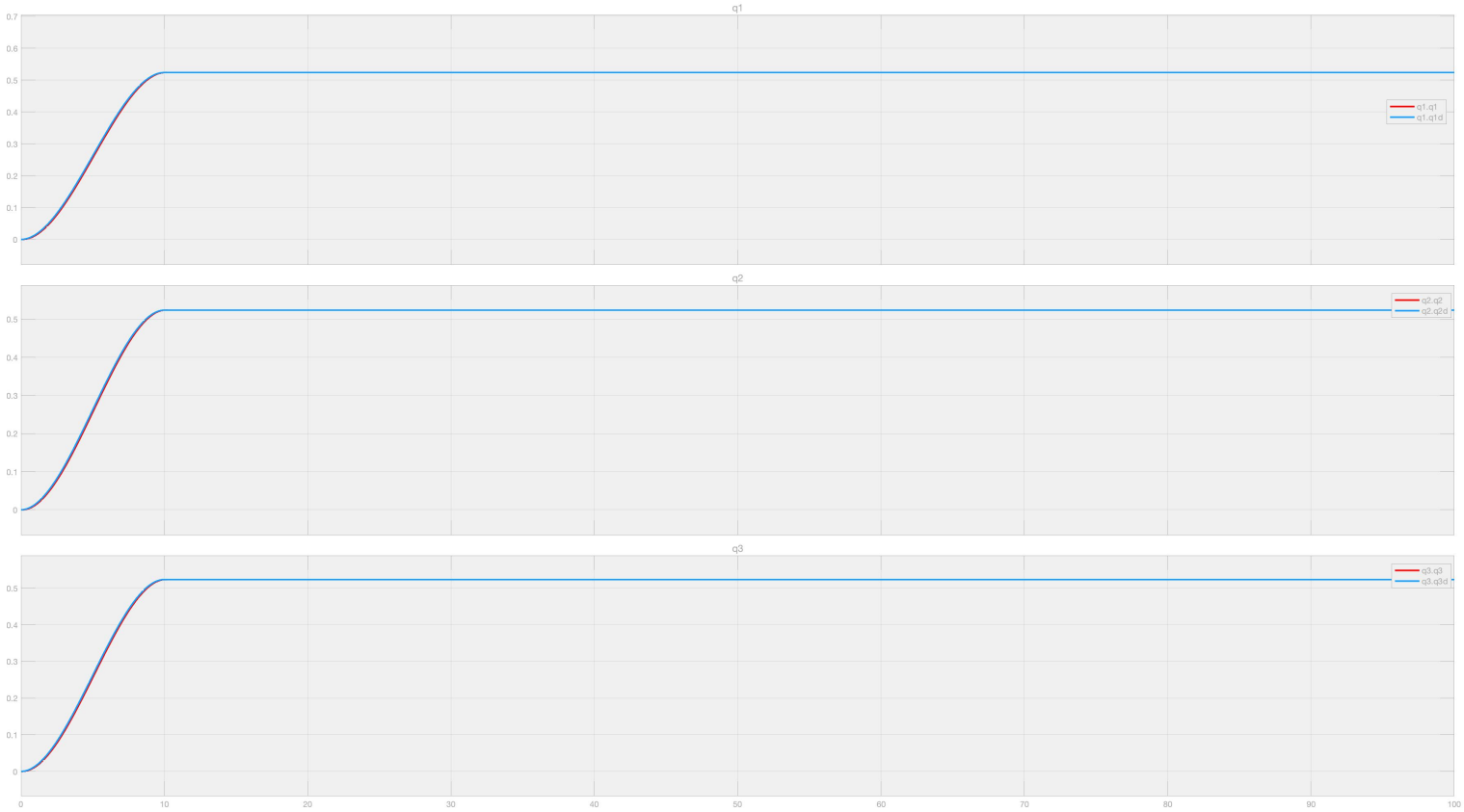

Figure 4: PD with gravity compensation

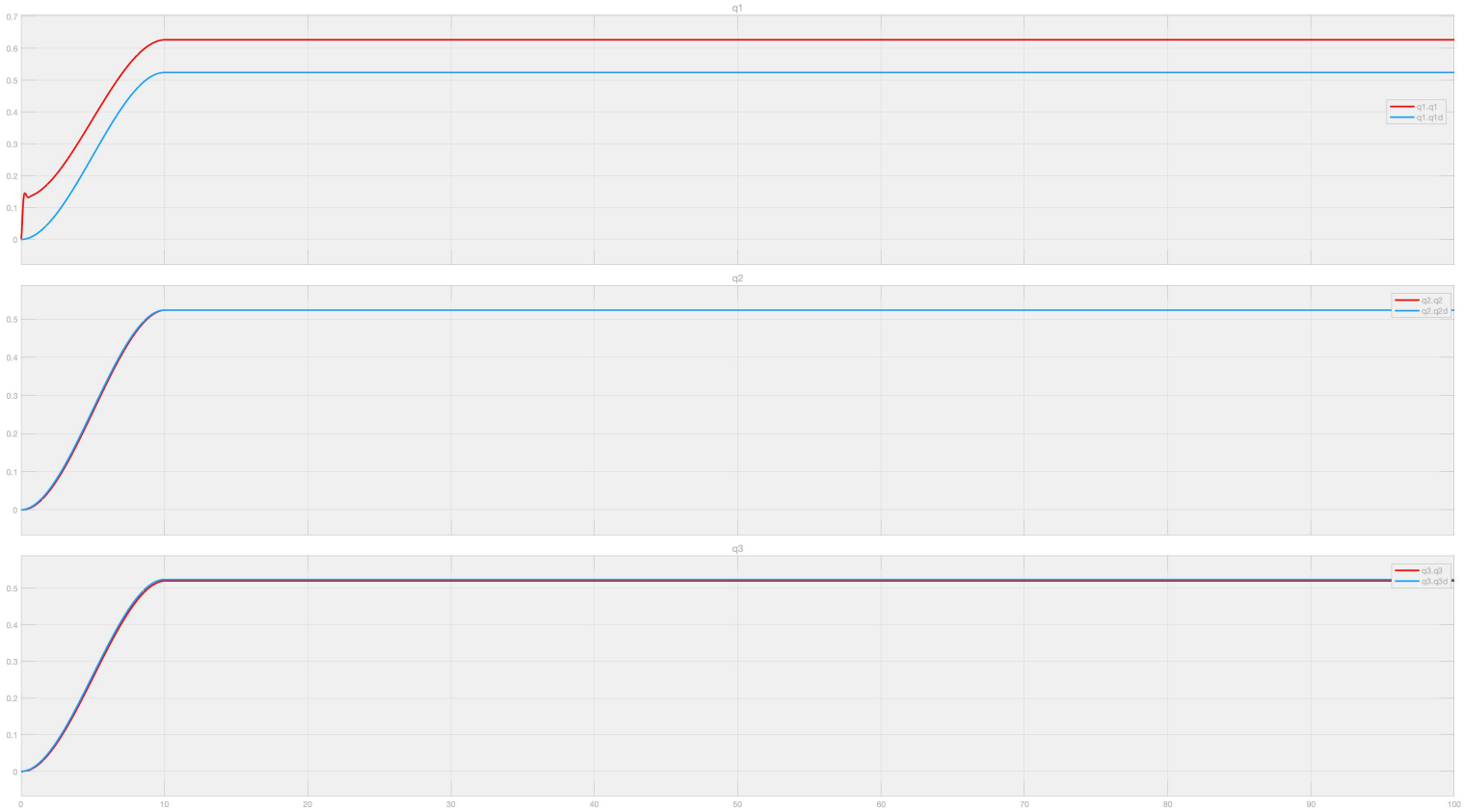

Figure 5: PD without gravity compensation

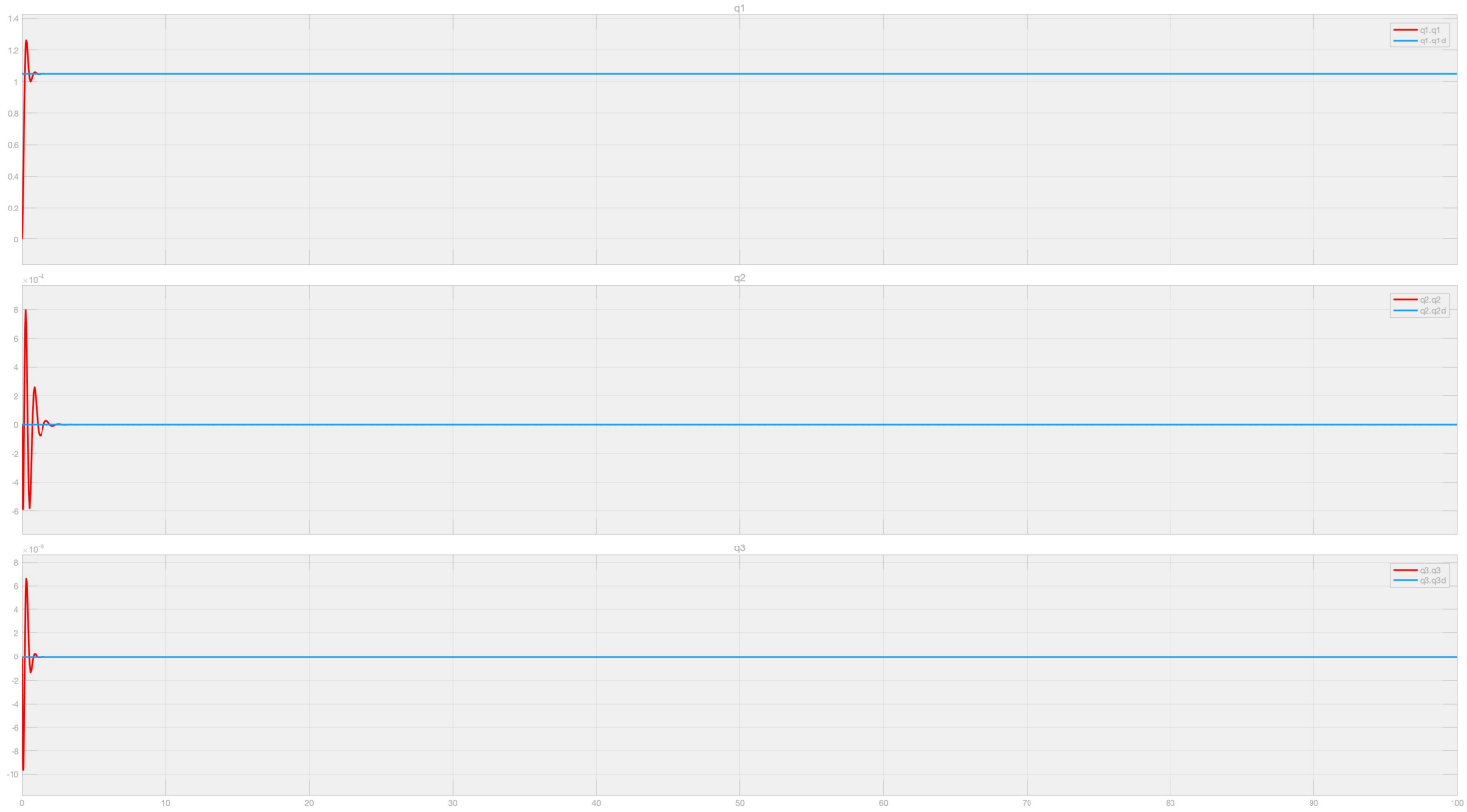

Figure 6: PD with gravity compensation - constant compensation at xd

4.2 Joint space inverse dynamics

Params:

- Kp : eye(3) * 1000
- Kd : eye(3) * 40

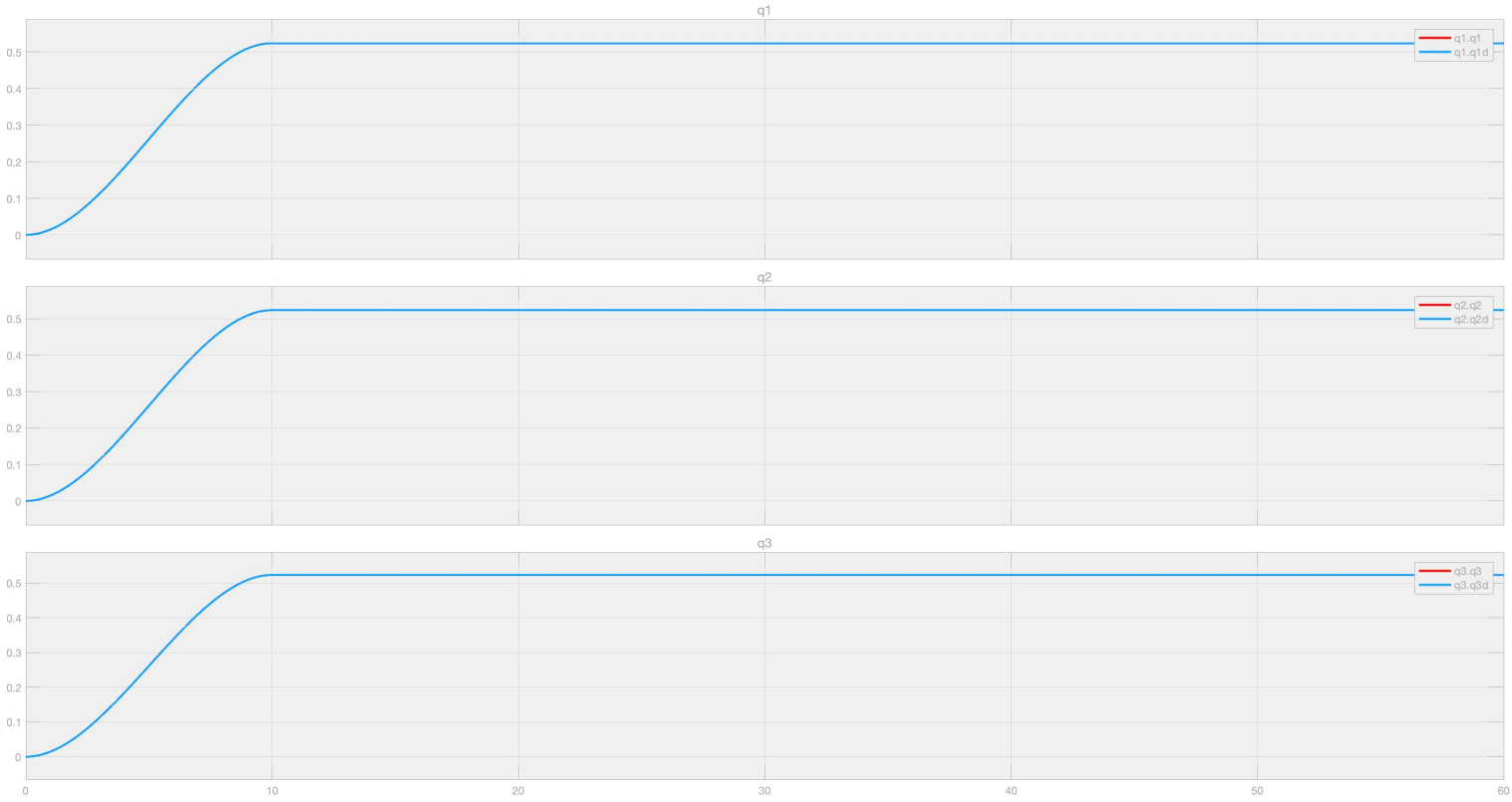

Figure 7: Joint space inverse dynamics

4.3 Adaptive control

Params:

- Kp : eye(3) * 1000
- Kd : eye(3) * 40

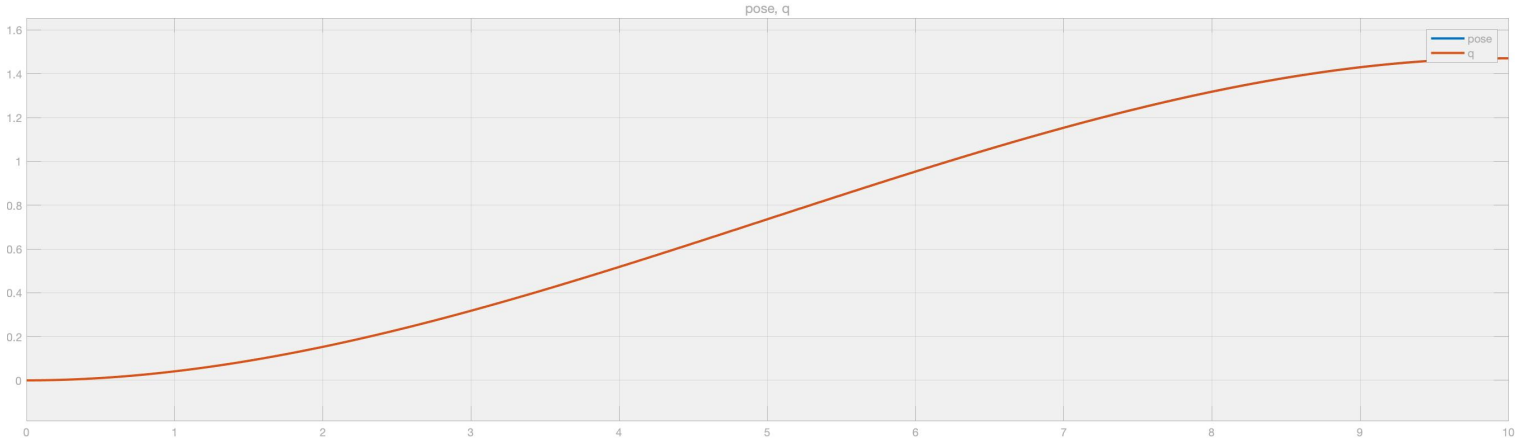

Figure 8: Adaptive control

4.4 Operational space PD controller with gravity compensation

- kp : diag([250 200 150 30 30 50]);
- kd : diag([50 30 10 10 10 10]);

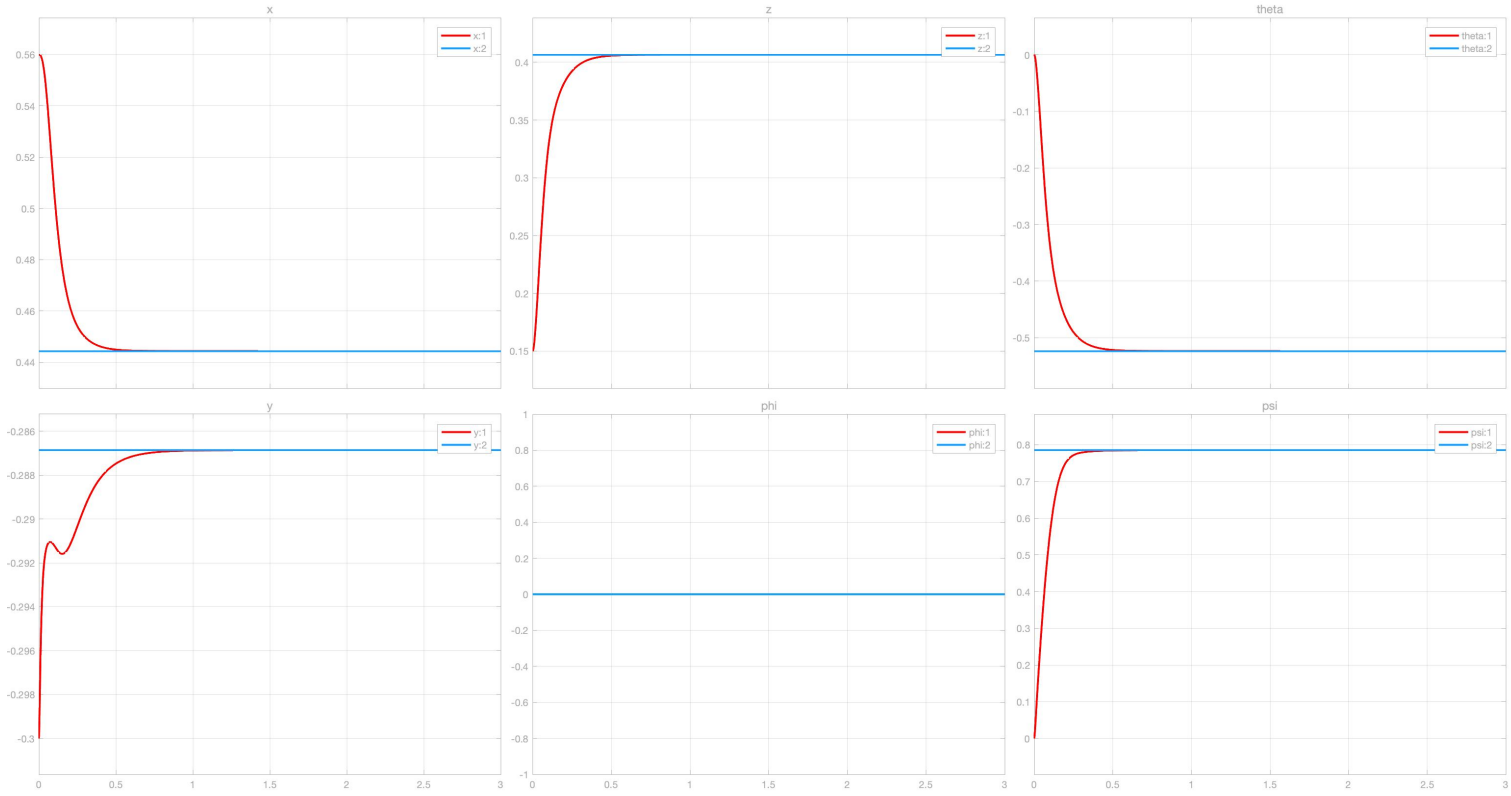

Figure 9: PD in operational space

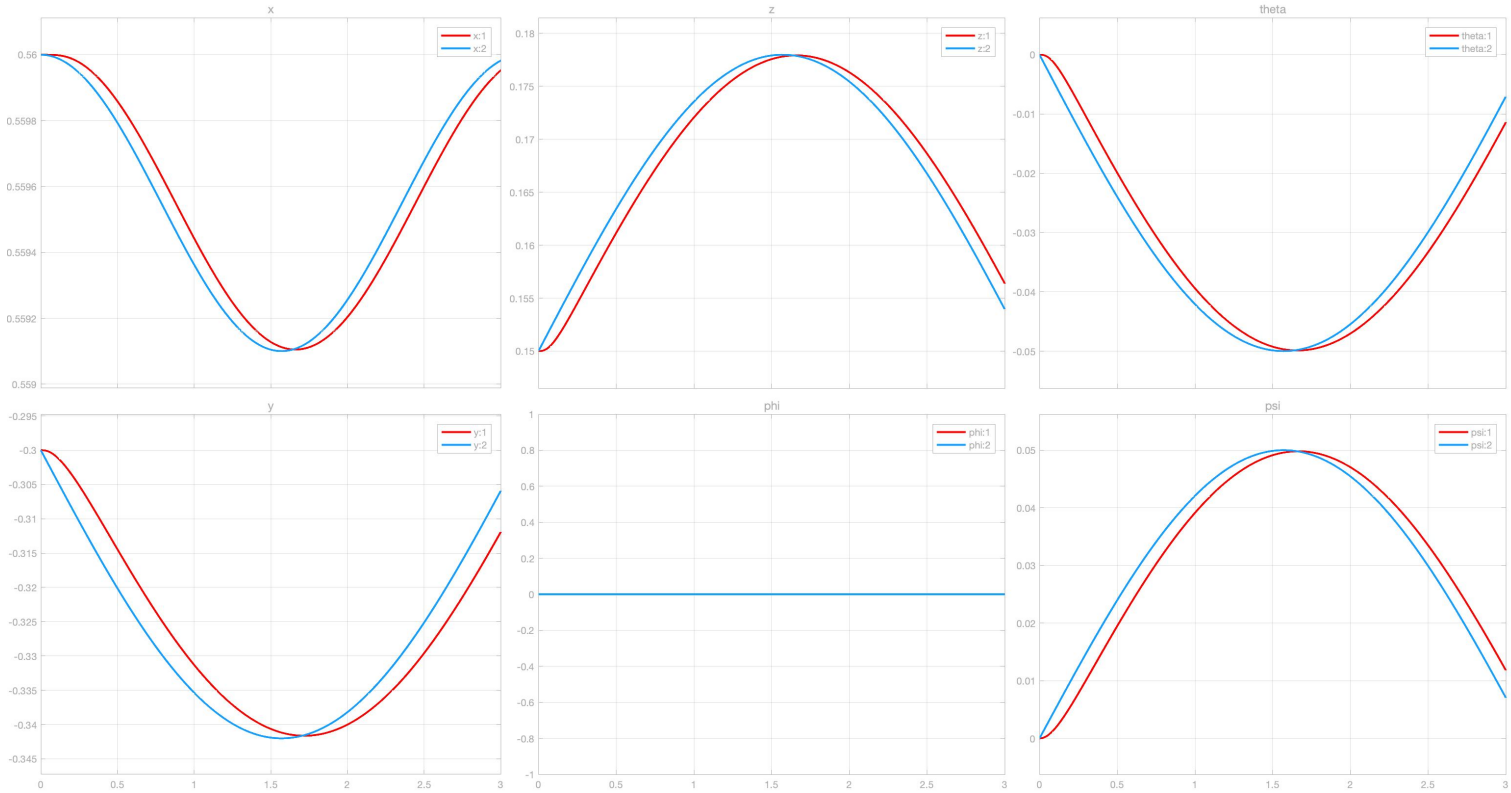

Figure 10: PD in operational space - Sine wave

4.5 Operational space inverse dynamics

- w : diag([200 200 200 10 10 10]);
- z : diag([0.95 0.95 0.95 0.95 0.95 0.95]);
- Kp : w^2;
- Kd : $2 * w * z$

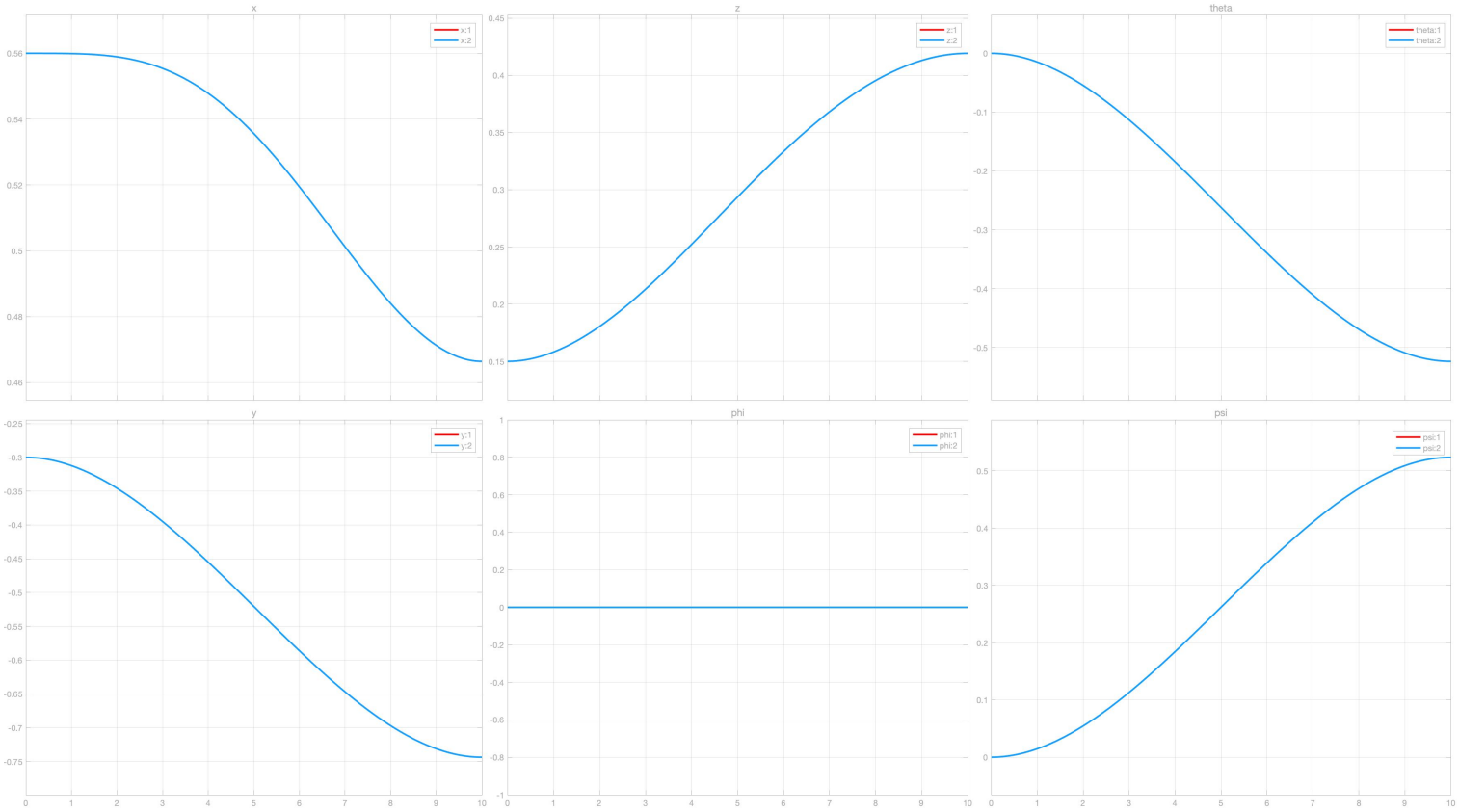

Figure 11: Inverse dynamics in operational space

4.6 Compliance

4.6.1 case 1 : KP >> K environment

- w : eye(6)*5
- z : eye(6)*0.1
- Kp : w^2;
- Kd : $2 * w * z$
- K : diag([2 2 2 0.1 0.1 0.1])

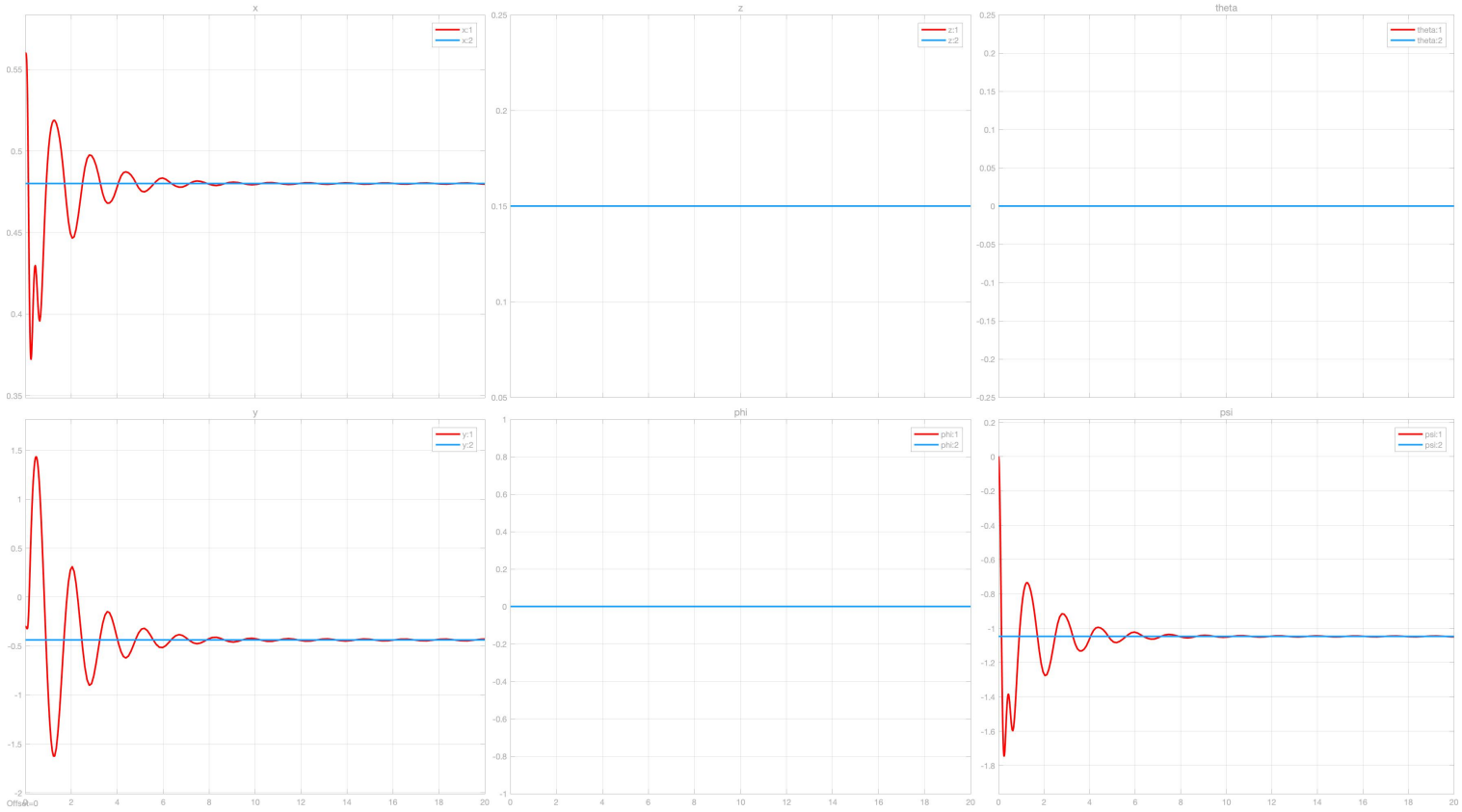

Figure 12: Compliance control in operational space

Figure 13: Compliance control in operational space - he plot

4.6.2 case 1 : K environment >> KP

- w : eye(6)*1
- z : eye(6)*0.1
- Kp : w^2;
- Kd : $2 * w * z$
- K : diag([2 2 2 0.1 0.1 0.1])*10

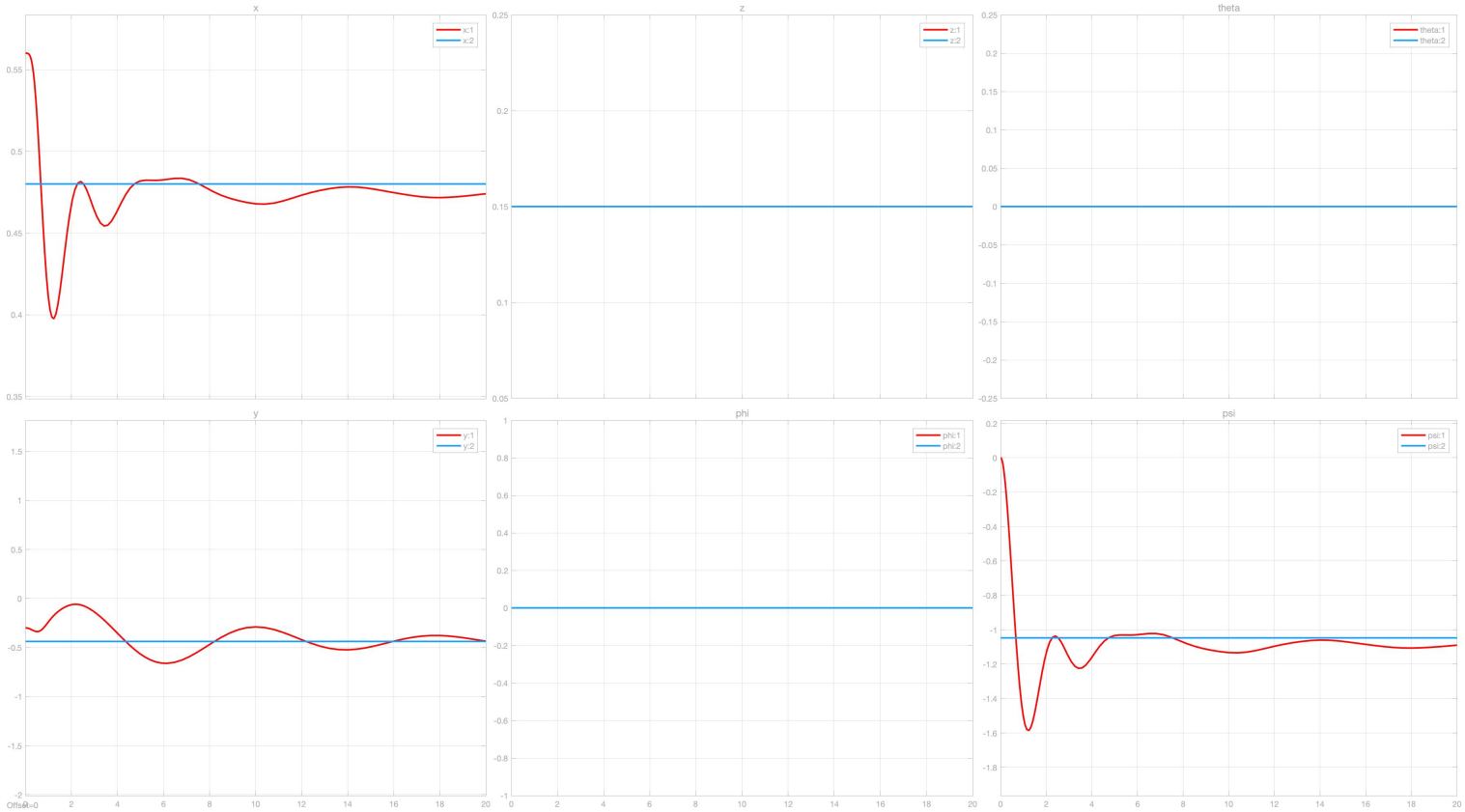

Figure 14: Compliance control in operational space

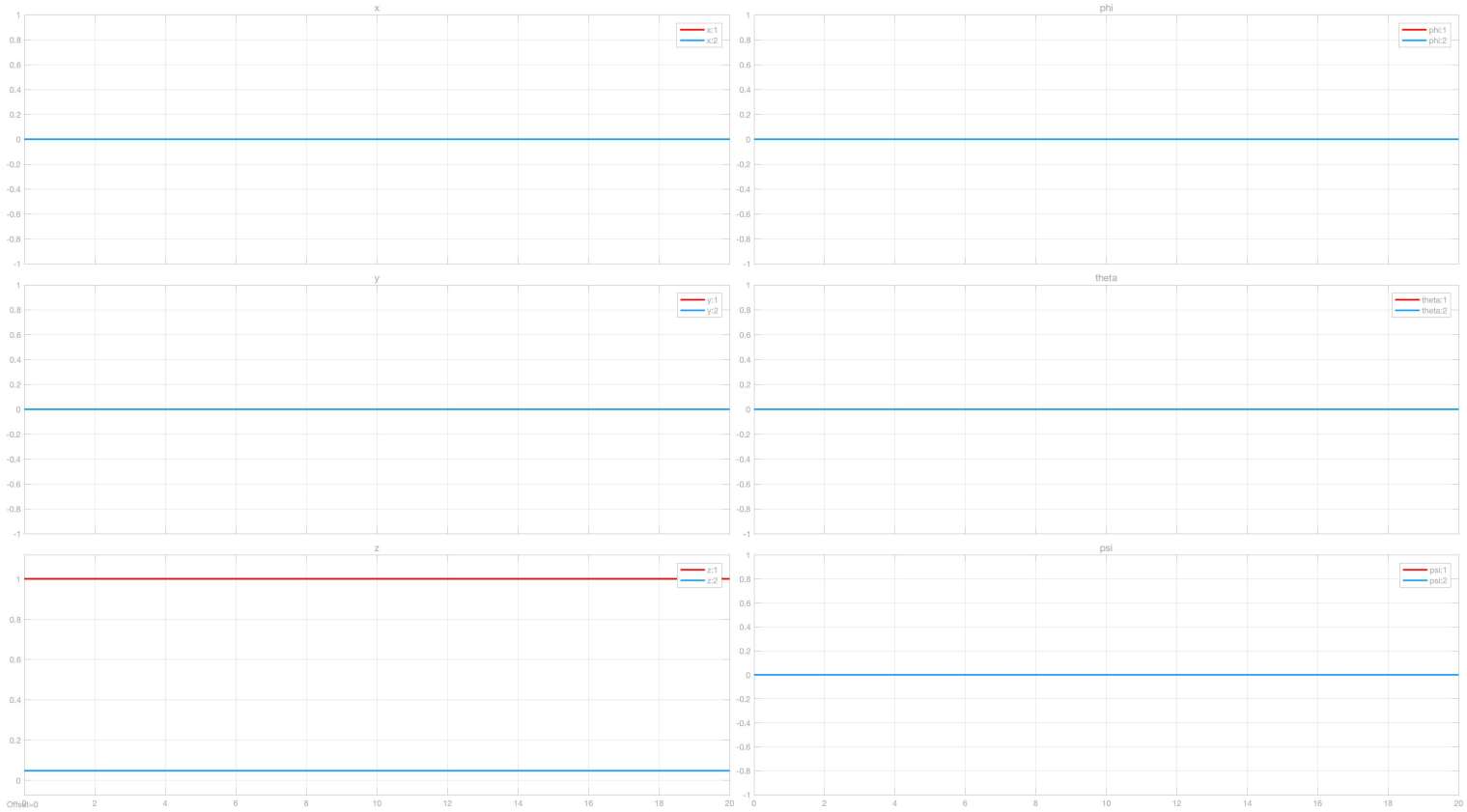

Figure 15: Compliance control in operational space - he plot

4.7 Impedance

- w : eye(6)*10
- z : eye(6)*1
- Md : eye(6);
- Kp : w^2;
- Kd : $2 * w * z$

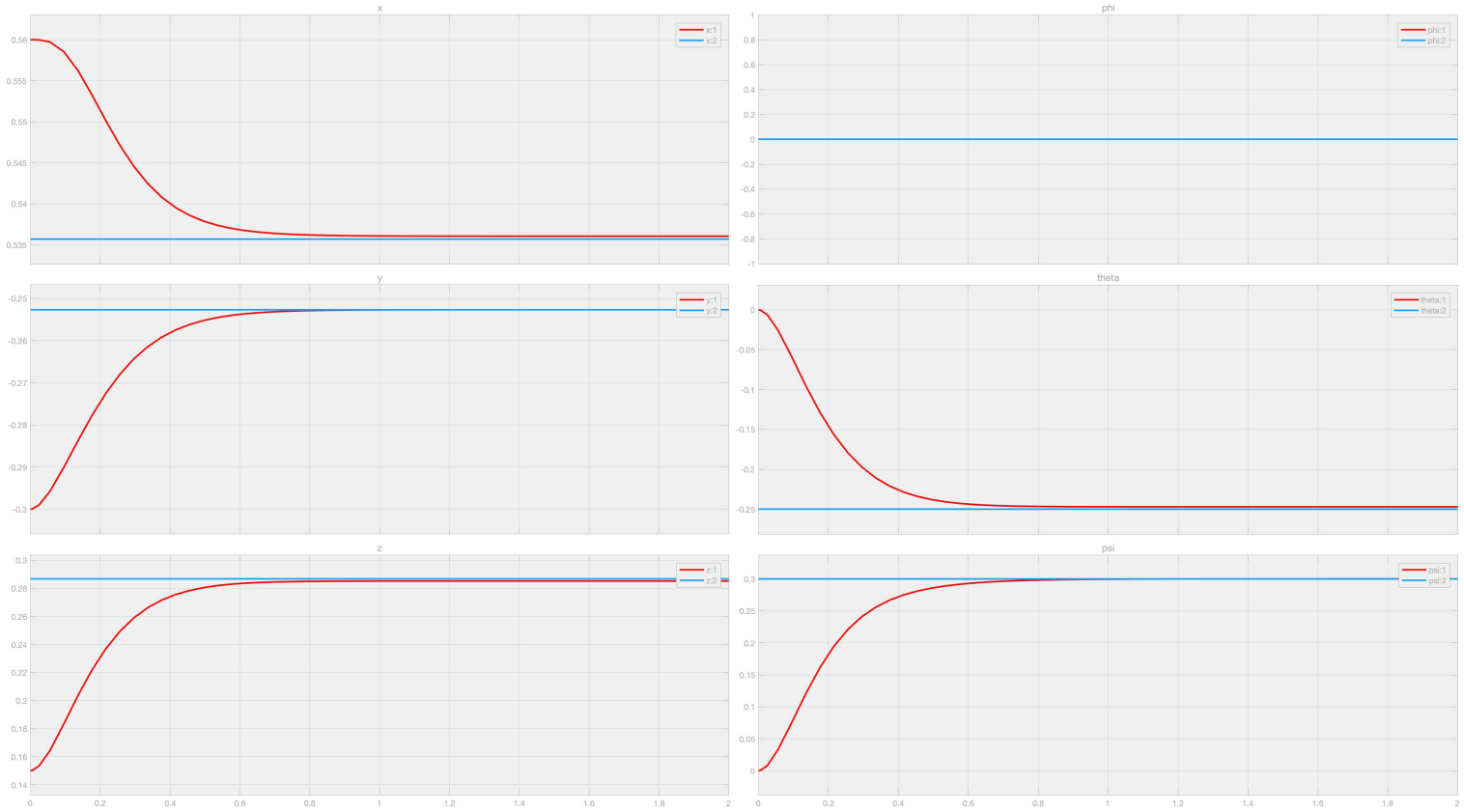

Figure 16: Impedance control in operational space

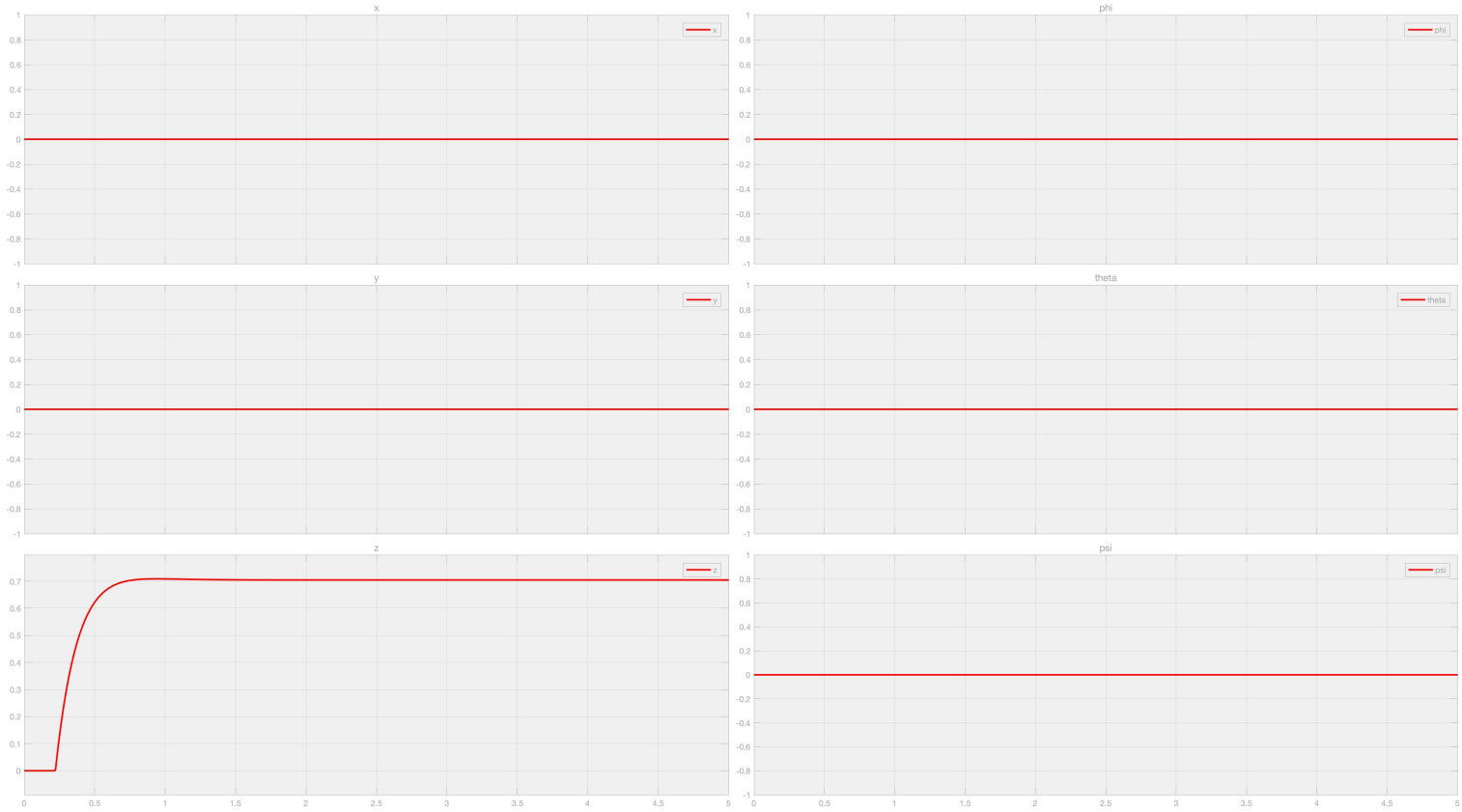

Figure 17: Impedance control in operational space - he

4.8 Admittance

- Kp : $eye(6) * 200$;
- Kd : $eye(6) * 35$;
- wt $= 10 * eye(6)$;
- zt $= 0.95 * eye(6)$;
- Mt : $0.075 * eye(6)$;
- KPt $= wt^2$;
- KDt $= 2 * zt * wt$;

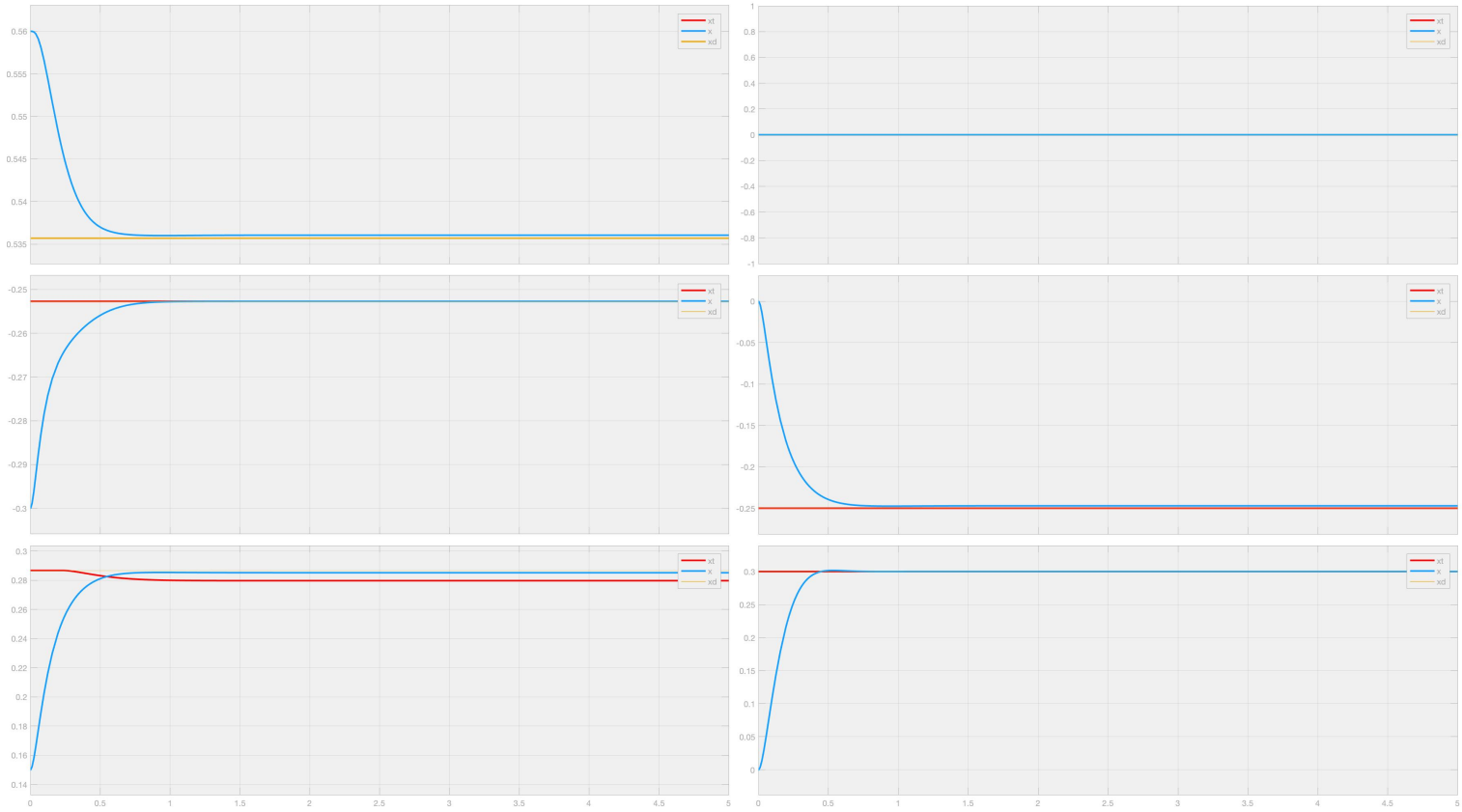

Figure 18: Admittance control in operational space

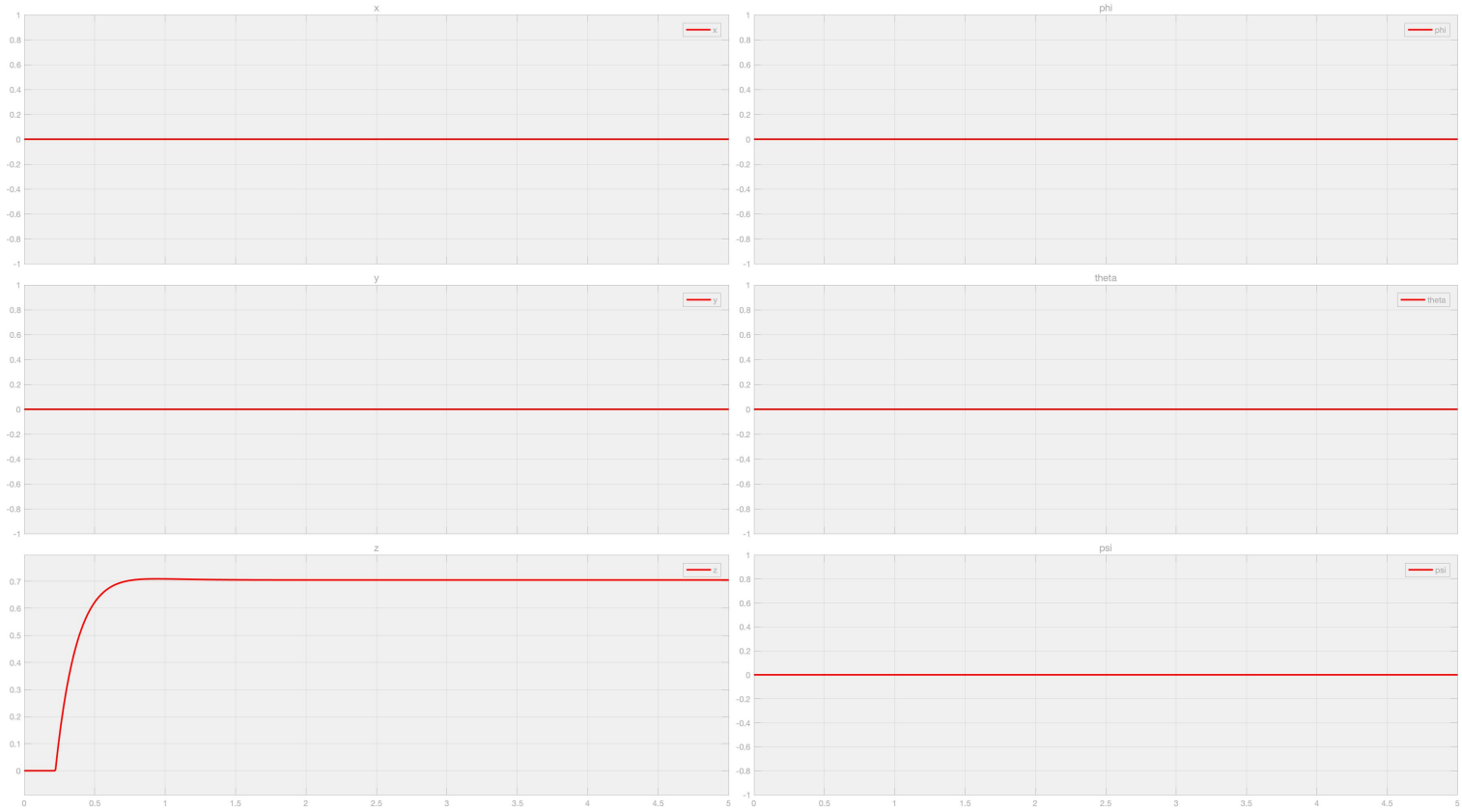

Figure 19: Admittance control in operational space - he

4.9 Force with inner position loop

- w : $eye(6) * 5$;
- z : $eye(6) * 0.95$;
- Md : $0.035 * eye(3)$;
- Kp : w^2;
- Kd : $2 * w * z$;
- kfp : $5 * eye(3)$;
- kfi : $5 * eye(3)$;

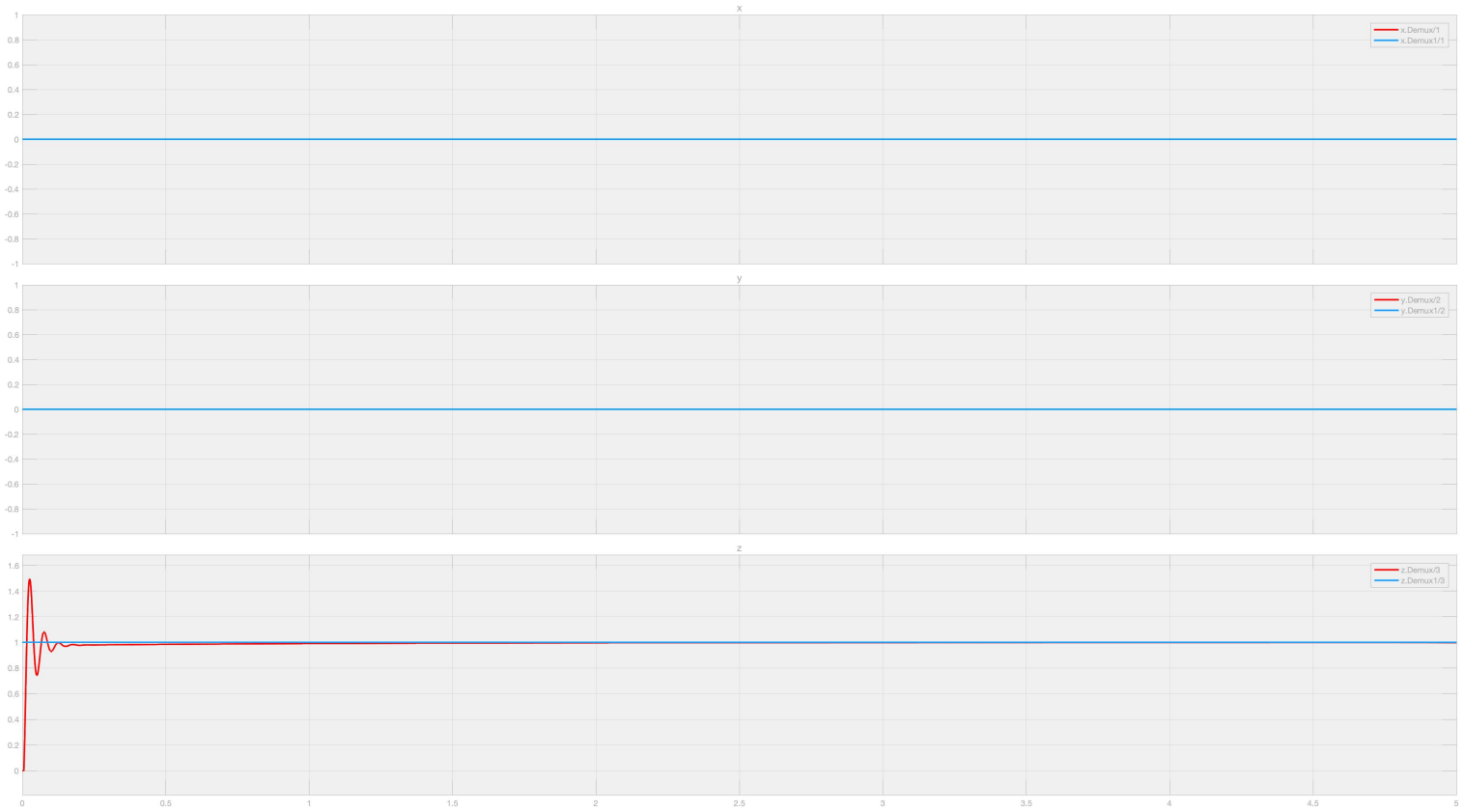

Figure 20: Force-position control in operational space, only [x,y,z] considered

4.10 Parallel force-position

- w : $eye(6) * 2.5$;
- z : $eye(6) * 0.8$;
- Md : $0.035 * eye(3)$;
- Kp : w^2;
- Kd : $2 * w * z$;
- kfp : $1.95 * eye(3)$;
- kfi : $1.5 * eye(3)$;

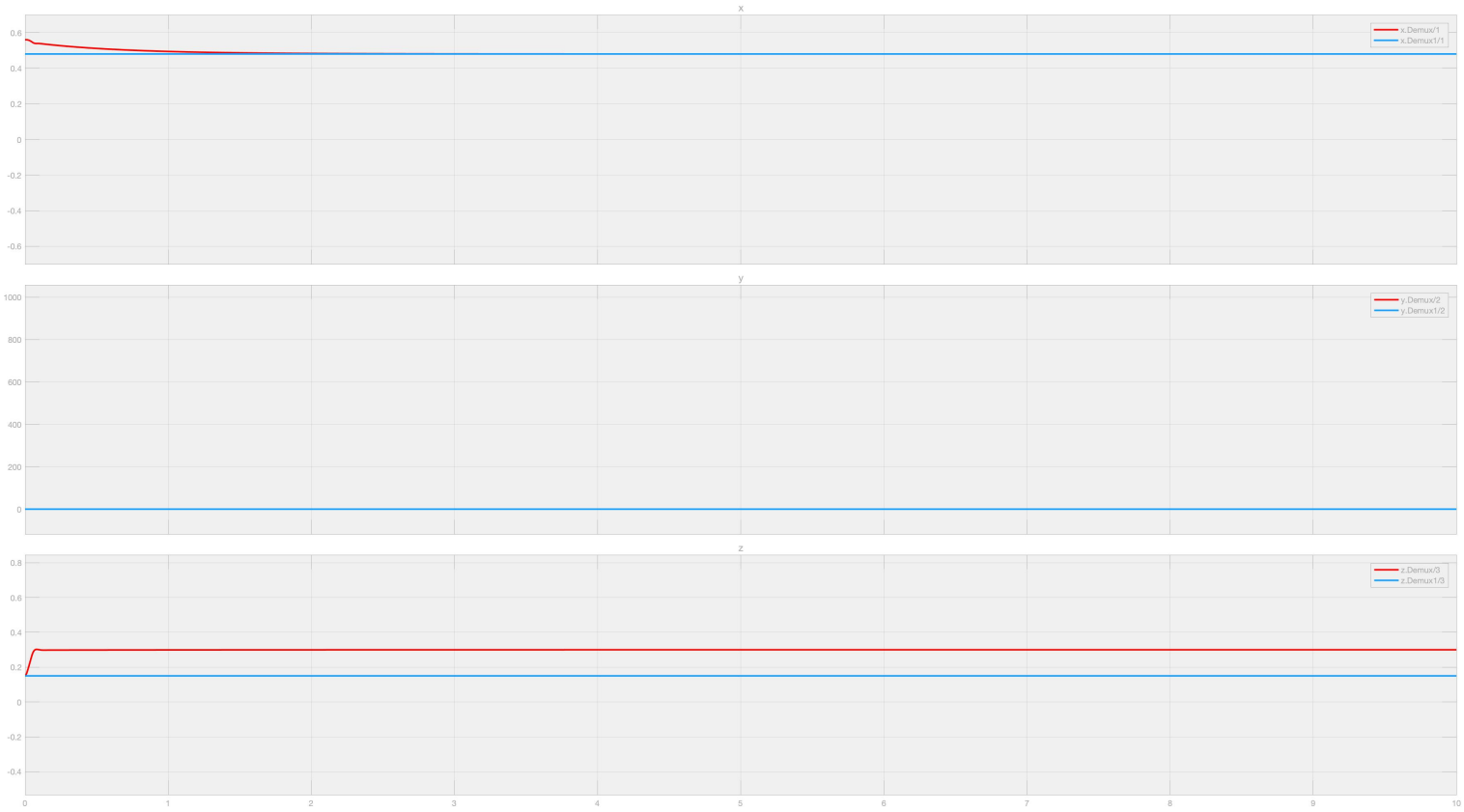

Figure 21: Imposing position - Parallel force-position control in operational space, only [x,y,z] considered

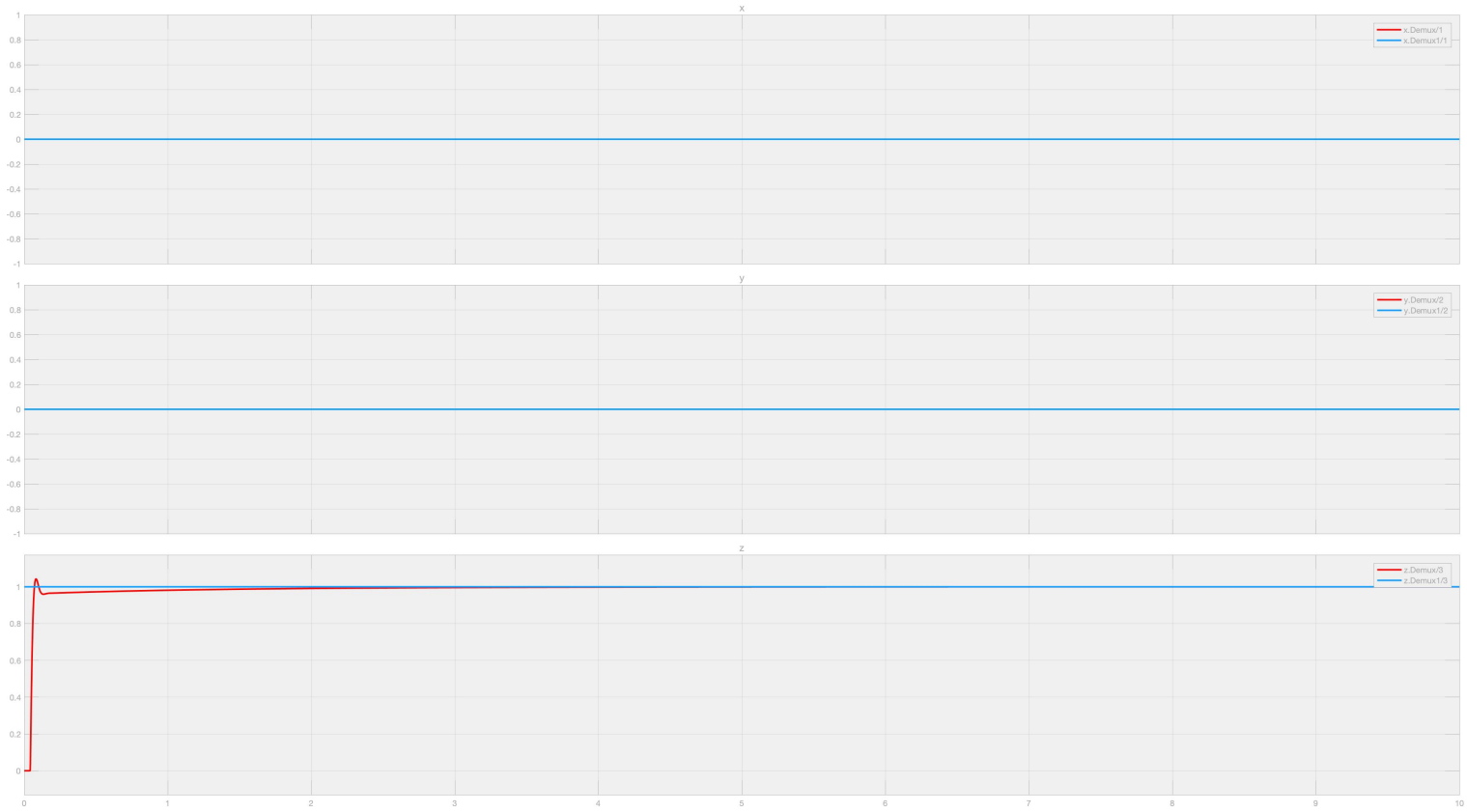

Figure 22: Imposing force - Parallel force-position control in operational space, only [x,y,z] considered