Physical Human-Robot Interaction

Michel Valentini
VR472456

February 2023

Contents

1 Assignment 1

1.1 Implement the Single-Input Single-Output Four-channel bilateral teleoperation architecture

To satisfy the trasparency controllers are defined as follow:

- $C1 = Z_s + C_s$;
- $C2 = I$;
- $C3 = I$;
- $C4 = (Z_m + C_m)$

where Z_m and Z_s rapresents the master and slave impedance.

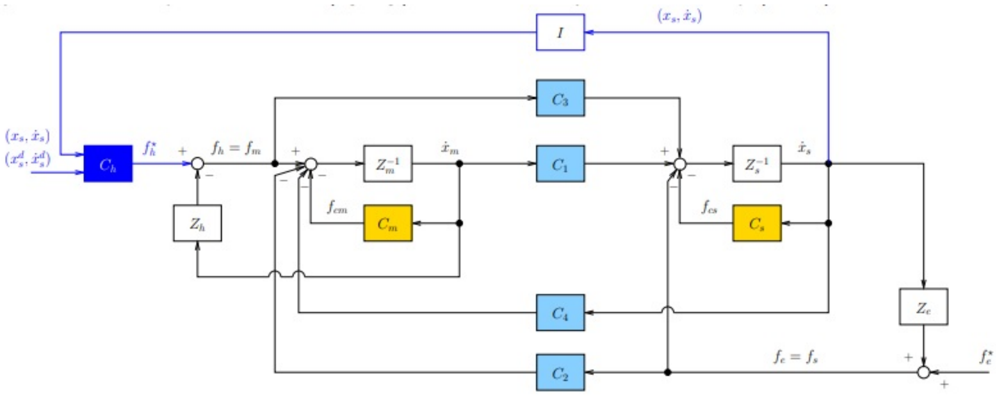

Figure 1: Four channel bilateral teleoperation architecture

The slave could be in free motion or in contact with the environment, that is modeled as a spring with stiffness $Ke = 200$. First define the robots dynamics as:

- without damping : $Z_m^{-1} = \frac{1}{M_m s}$ and $Z_s^{-1} = \frac{1}{M_s s}$;
- with damping : $Z_m^{-1} = \frac{1}{M_m s + Dm}$ and $Z_s^{-1} = \frac{1}{M_s s + Ds}$;

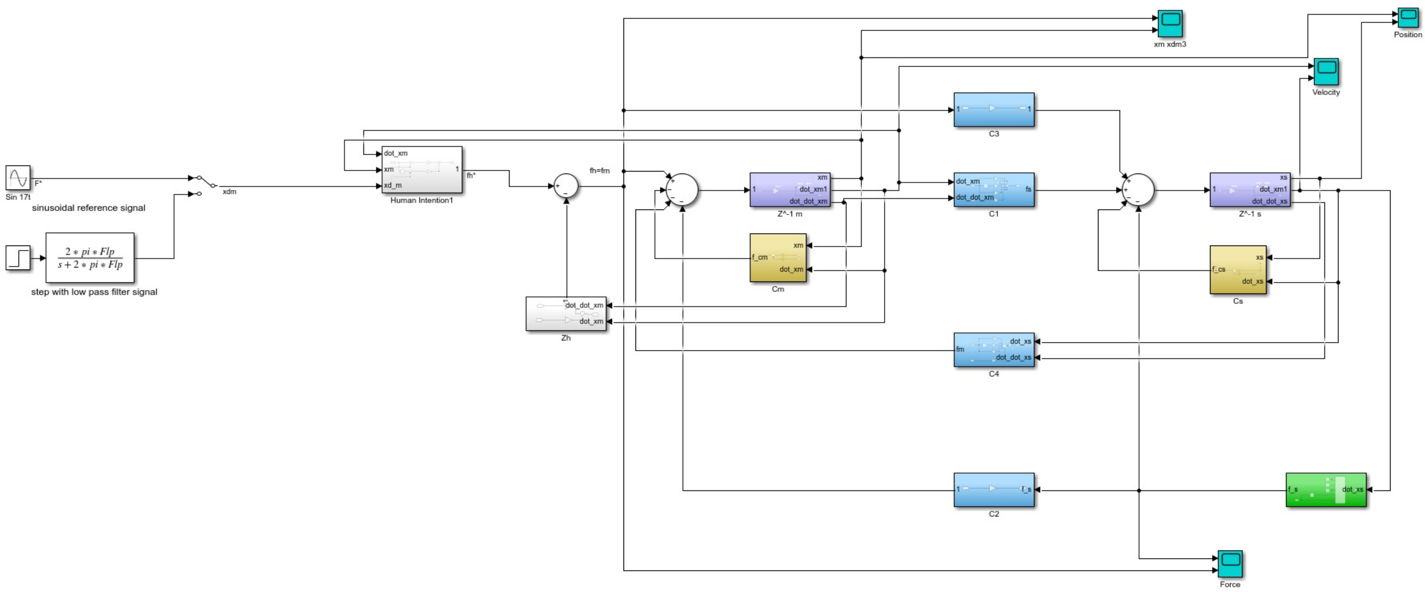

Figure 2: Four channel bilateral teleoperation architecture Simulink

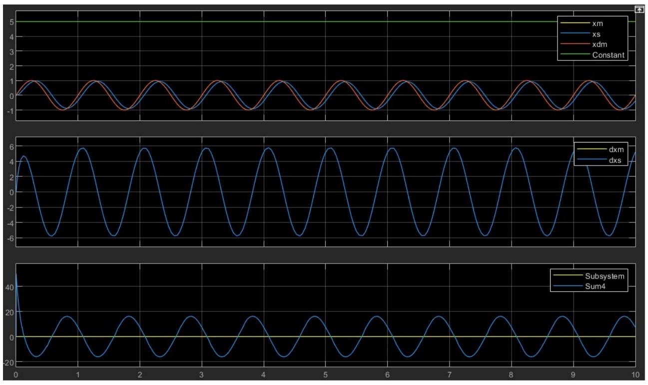

Figure 3: Four channel bilateral teleoperation architecture free motion no damping

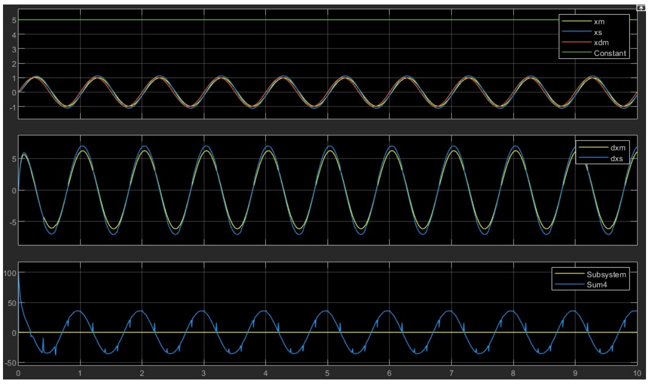

Figure 4: Four channel bilateral teleoperation architecture free motion with damping

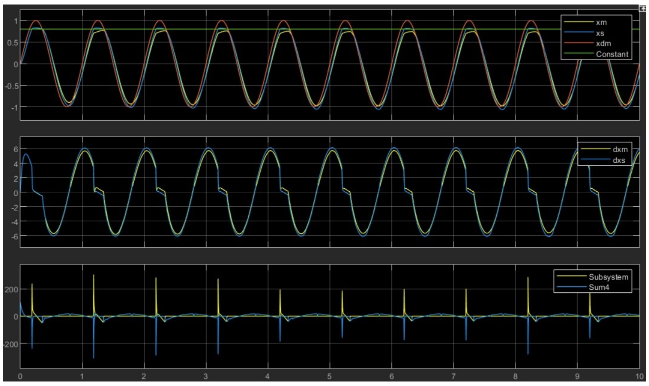

Figure 5: Four channel bilateral teleoperation architecture in contact no damping

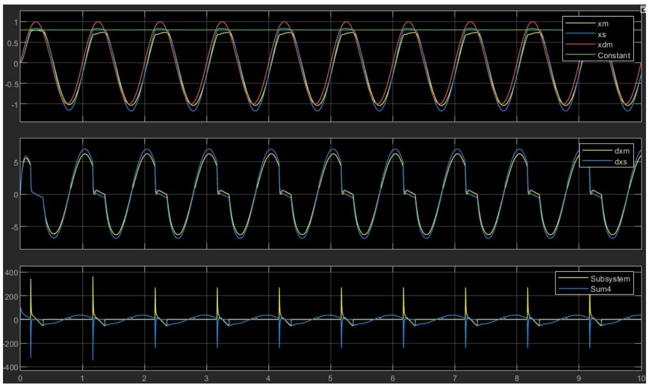

Figure 6: Four channel bilateral teleoperation architecture in contact with damping

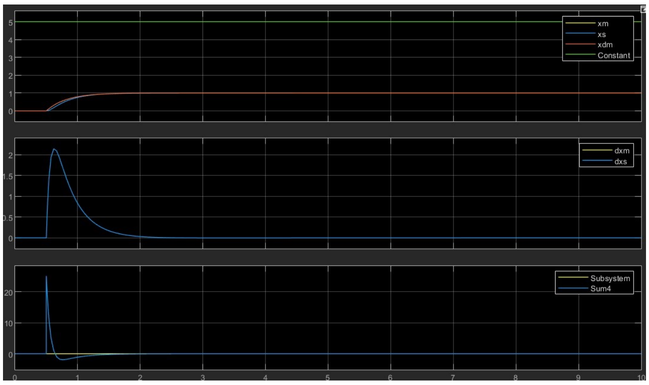

Figure 7: Four channel bilateral teleoperation architecture Step signal free motion no damping

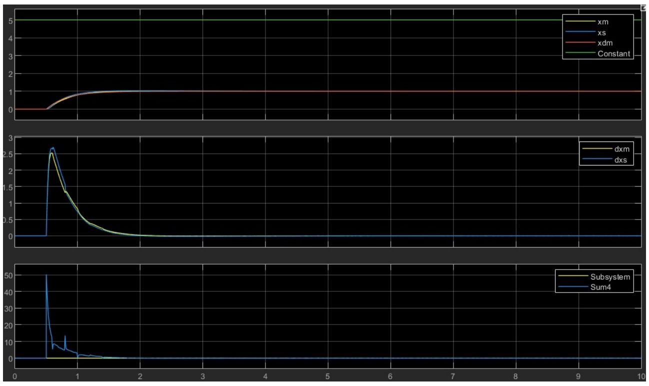

Figure 8: Four channel bilateral teleoperation architecture Step signal free motion with damping

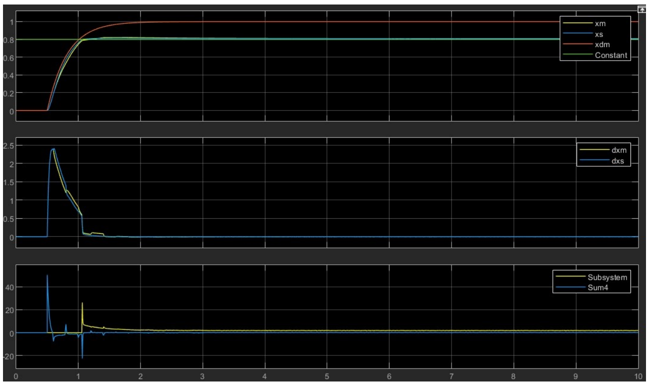

Figure 9: Four channel bilateral teleoperation architecture Step signal in contact no damping

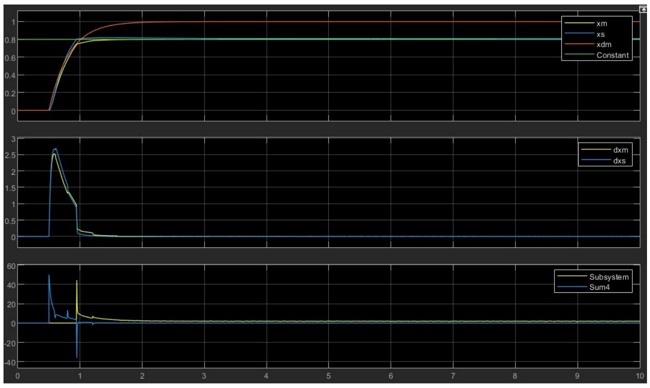

Figure 10: Four channel bilateral teleoperation architecture Step signal in contact with damping

2 Assignment2

2.1 Implement the three Two-Channel bilateral teleoperation architectures and the Three-Channel bilateral teleoperation architecture

2.2 Two-Channel: Position Position PP

Thanks to C4, the operator perceive a force feedback though the slave robot when it interact with the environment.

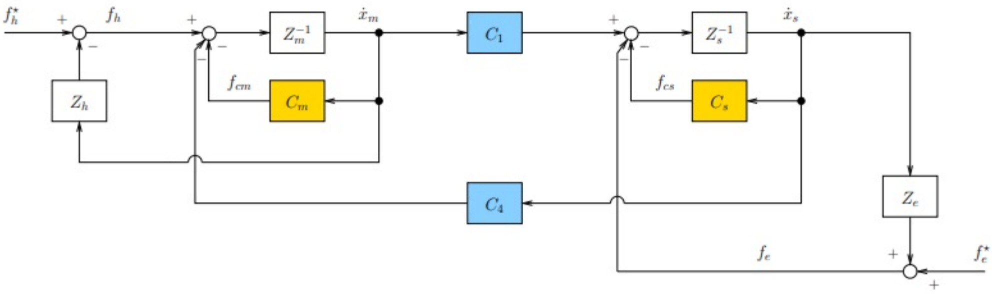

Figure 11: Two-Channel Position Position Block scheme

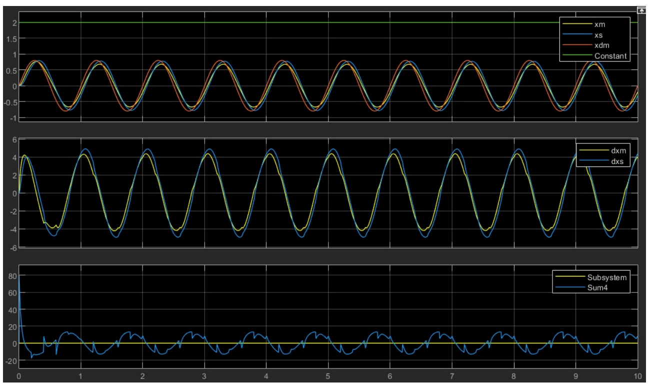

Figure 12: Two-Channel Position Position no delay in free motion

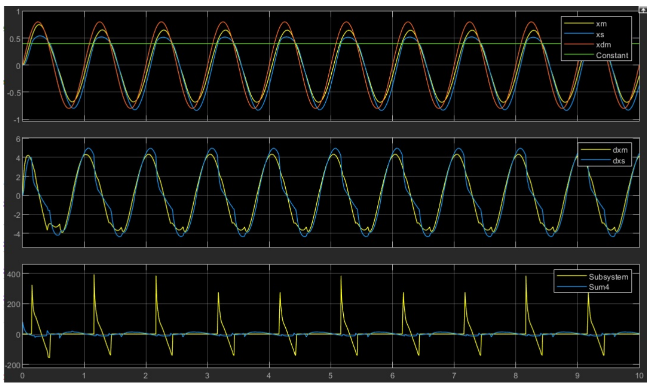

Figure 13: Two-Channel Position Position no delay in Contact

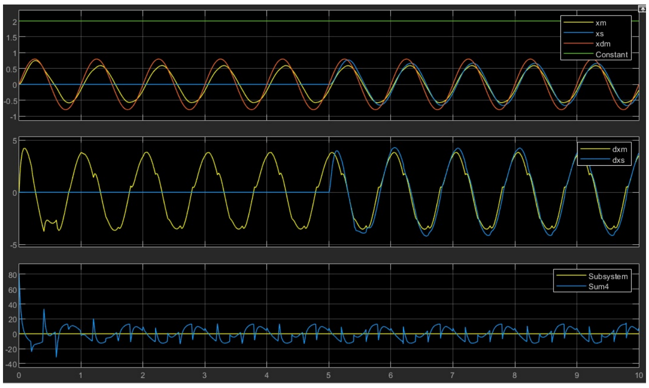

Figure 14: Two-Channel Position Position with delay in free motion

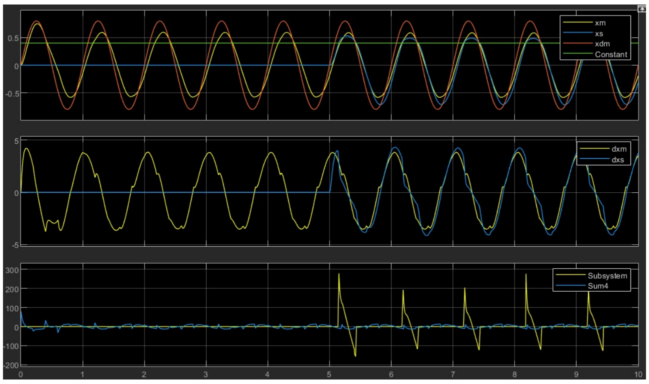

Figure 15: Two-Channel Position Position with delay in Contact

2.3 Two-Channel: Force Position FP

When the slave robot does not interact with the environment, the operator perceive zero force feedback.

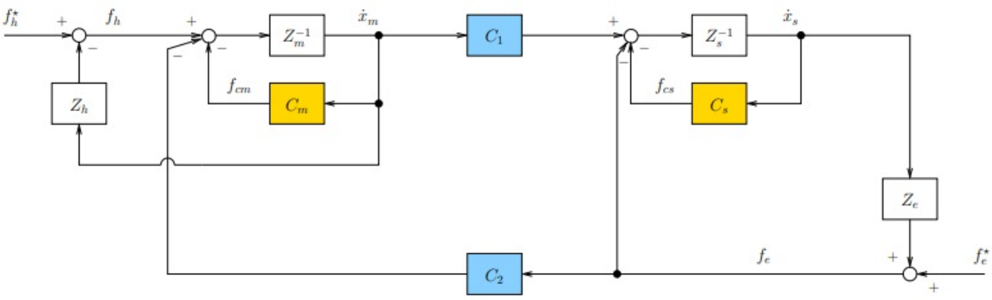

Figure 16: Two-Channel Force Position Block scheme

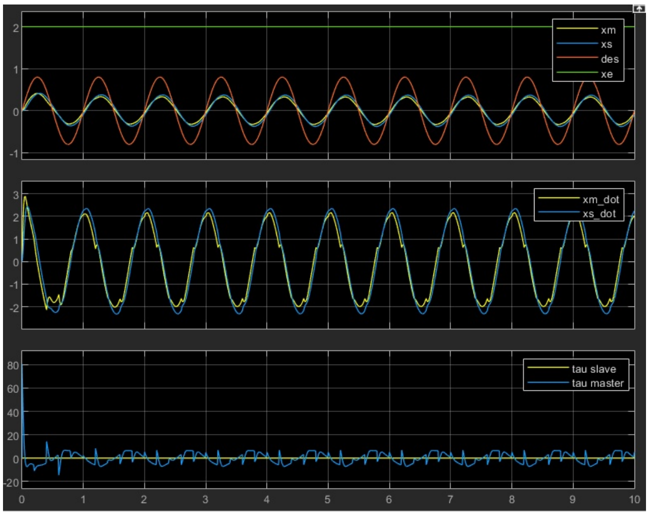

Figure 17: Two-Channel Force Position no delay in free motion

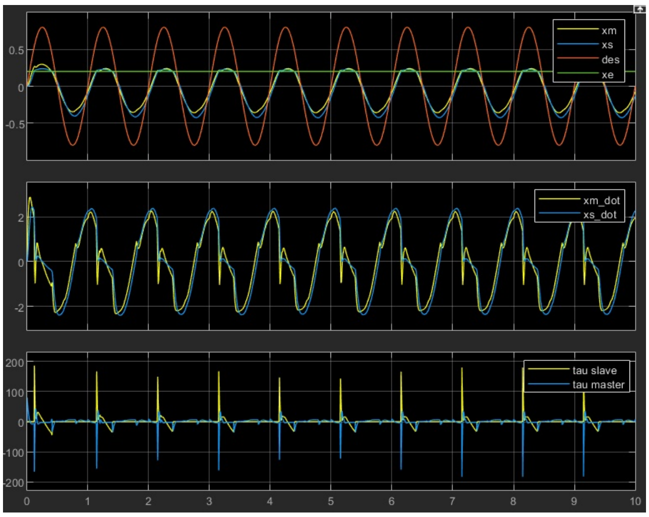

Figure 18: Two-Channel Force Position no delay in Contact

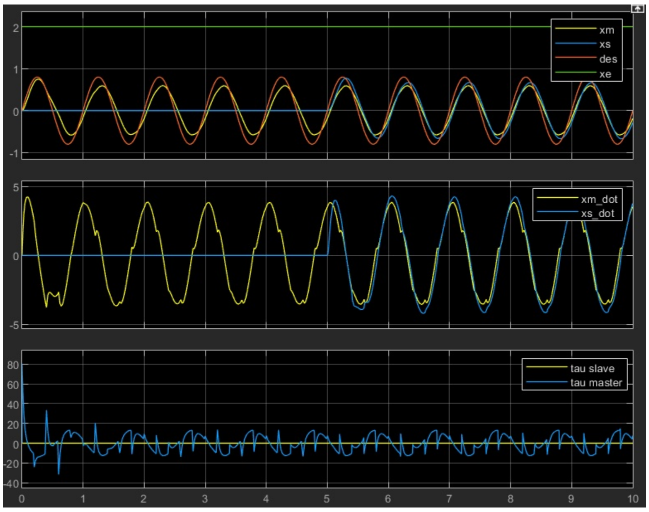

Figure 19: Two-Channel Force Position with delay in free motion

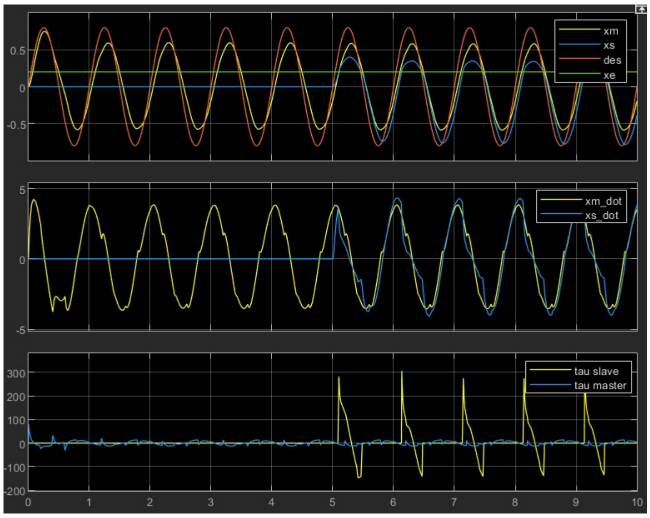

Figure 20: Two-Channel Force Position with delay in Contact

2.4 Two-Channel: Force Force FF

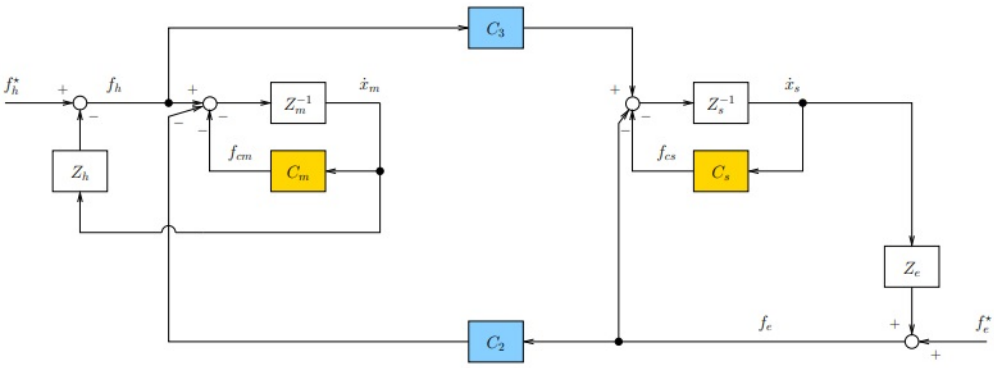

Figure 21: Two-Channel Force Force Block scheme

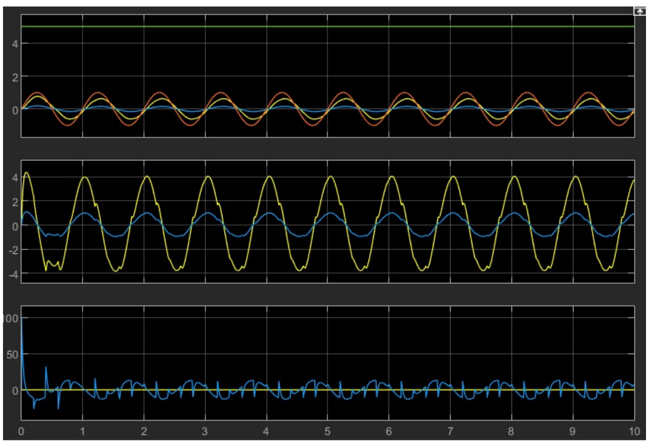

Figure 22: Two-Channel Force Force no delay in free motion

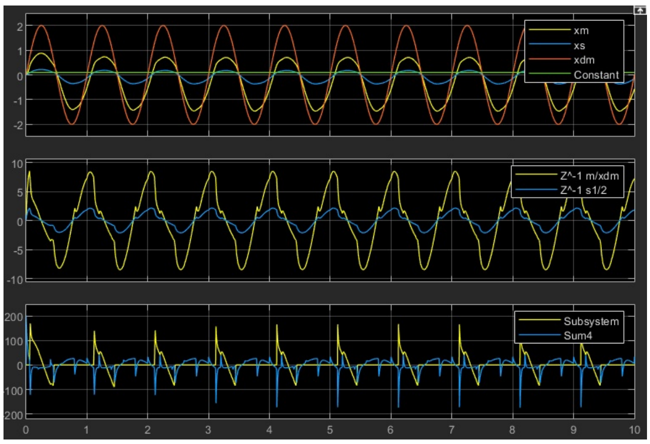

Figure 23: Two-Channel Force Force no delay in Contact

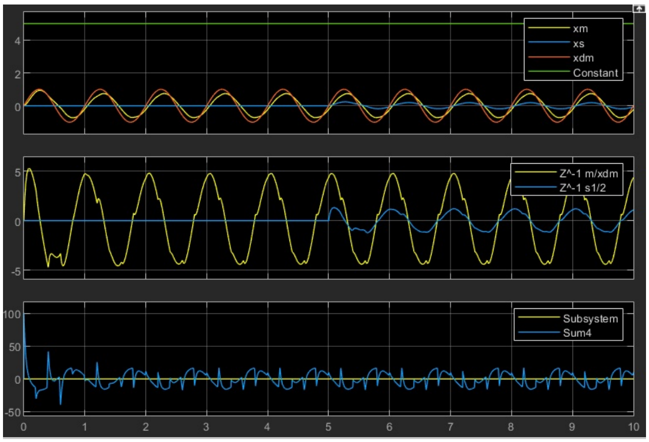

Figure 24: Two-Channel Force Force with delay in free motion

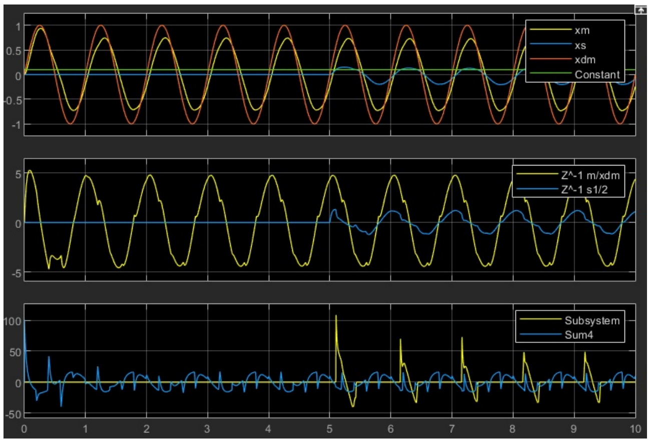

Figure 25: Two-Channel Force Force with delay in Contact

2.5 Three-Channel: Position and Force-Position

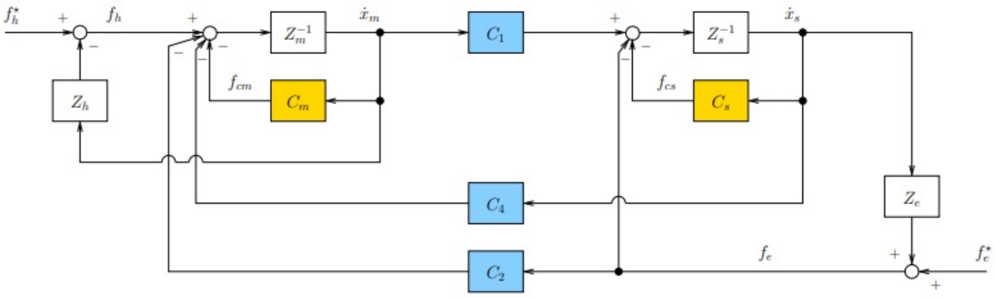

Figure 26: Two-Channel Force Force Block scheme

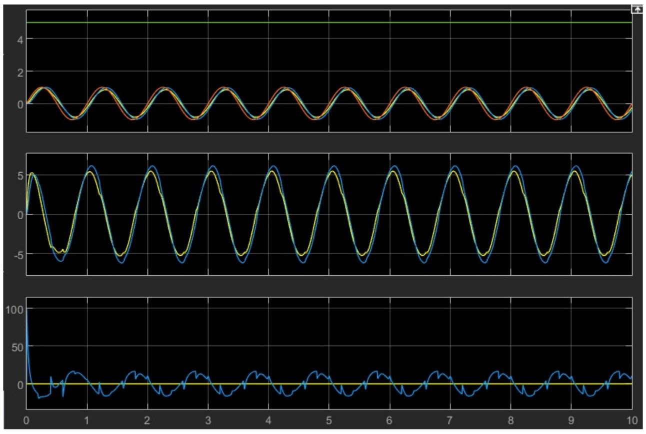

Figure 27: Three-Channel PF-P no delay in free motion

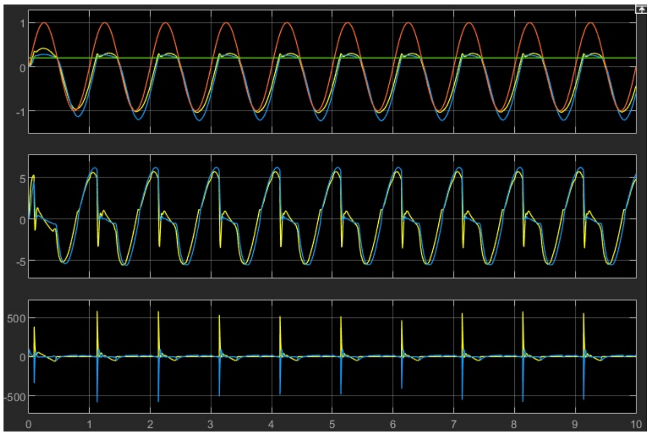

Figure 28: Three-Channel PF-P no delay in Contact

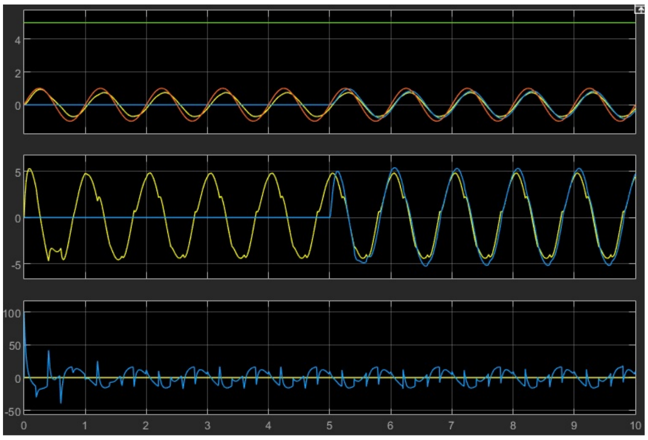

Figure 29: Three-Channel PF-P with delay in free motion

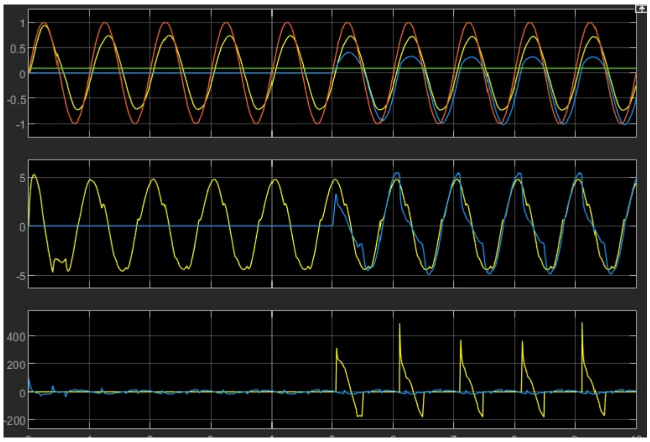

Figure 30: Three-Channel PF-P with delay in Contact

3 Assignment 3 and 4

3.1 Implement the Kalman filter/predictor and estimate the velocity and acceleration from noisy position measurements

To predict both velocity and acceleration we need to define a linear system with 3 states. The overall model is described as

$$\begin{cases} X_{k+1} = AX_k + Bw_k \\ y_k = CX_k + v_k \end{cases} \tag{1}$$

with $A = \begin{bmatrix} 1 & T_s & \frac{T_s^2}{2} \\ 0 & 1 & T_s \\ 0 & 0 & 1 \end{bmatrix}$, $B = \begin{bmatrix} \frac{T_s^3}{6} \\ \frac{T_s^2}{2} \\ T_s \end{bmatrix}$, $C = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$, w_k random variable describing model variance as $Q = qBB^2, q = 10^7, v_k$ random variable describing noise variance as $R = \frac{1}{1000}$ and the state of the system is then defined as $X = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$. To solve we use a recursive algorithm which at each iteration predict the next state value, and recompute variance and Kalman gain to better predict. To this we provide the initial condition for the state as $X_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$ and for the variance $P_0 = \begin{bmatrix} 0.1 & 0 & 0 \\ 0 & 0.1 & 0 \\ 0 & 0 & 0.1 \end{bmatrix}$

The difference between filter and predictor is that the former estimates the state $\hat{x}$ at time k+1 given measurements until time k + 1, while the latter estimates the state $\hat{x}$ at time k + 1 given measurements until time k (1-step ahead prediction). Therefore the resulting estimations will differ only in the first state. Both filter and predictor are applied to a position signal with Gaussian noise and a signal with quantization noise.

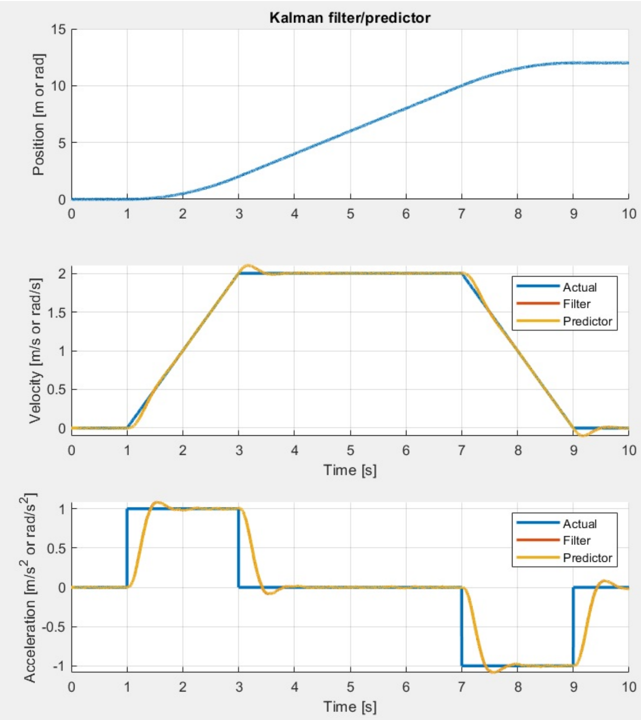

Figure 31: Kalman filter predictor Gaussian noise

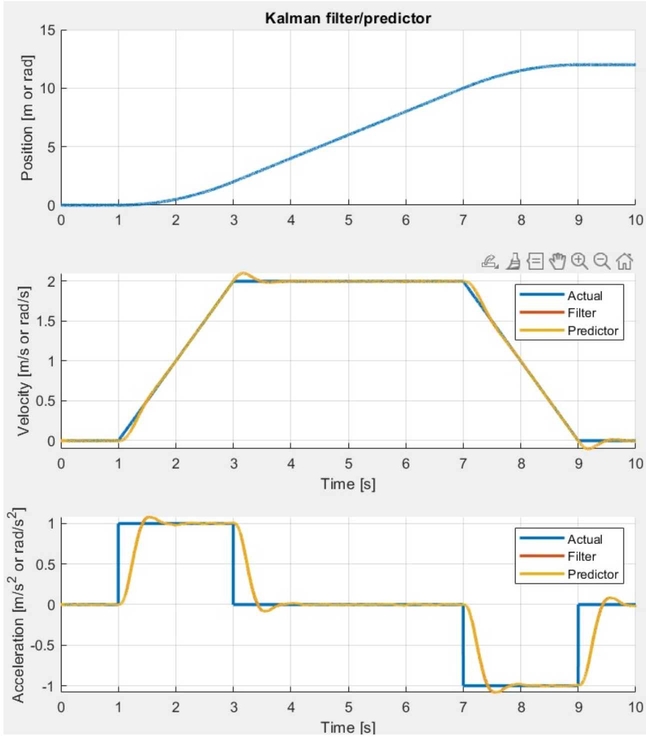

Figure 32: Kalman filter predictor Quantization Noise

3.2 Implement the steady-state Kalman filter/predictor and estimate the velocity and acceleration from noisy position measurements

In this case we have to provide a fixed solution for the Kalman Gain K inf and the variance P_{inf} is obtained by solving the Algebraic Riccati equation.

$$P_{\text{inf}} = AP_{\text{inf}}A^T - AP_{\text{inf}}C^T(CP_{\text{inf}}C^T + R)^{-1}CP_{\text{inf}}A^T + Q$$

Then, the Kalman Gains for the filter and predictor are defined as:

$$K_{\text{inf}} = P_{\text{inf}}C^T(CP_{\text{inf}}C^T + R)^{-1}$$

$$\bar{K}_{\text{inf}} = AP_{\text{inf}}C^T(CP_{\text{inf}}C^T + R)^{-1}$$

Finally, the steady state Kalman Filter is defined as:

$$\hat{x}_{k+1|k+1} = A\hat{x}_{k|k} + K_{\text{inf}}(y_{k+1} - CA\hat{x}_{k|k})$$

while the steady state Kalman predictor is defined as:

$$\hat{x}_{k+1|k} = A\hat{x}_{k|k-1} + \bar{K}_{\text{inf}}(y_k - CA\hat{x}_{k|k-1})$$

Those are algebrically computed using the matlab function idare, then the same gain is used for every iteration of the algorithm. Also both filter and predictor are applied to a position signal with Gaussian noise and a signal with quantization noise.

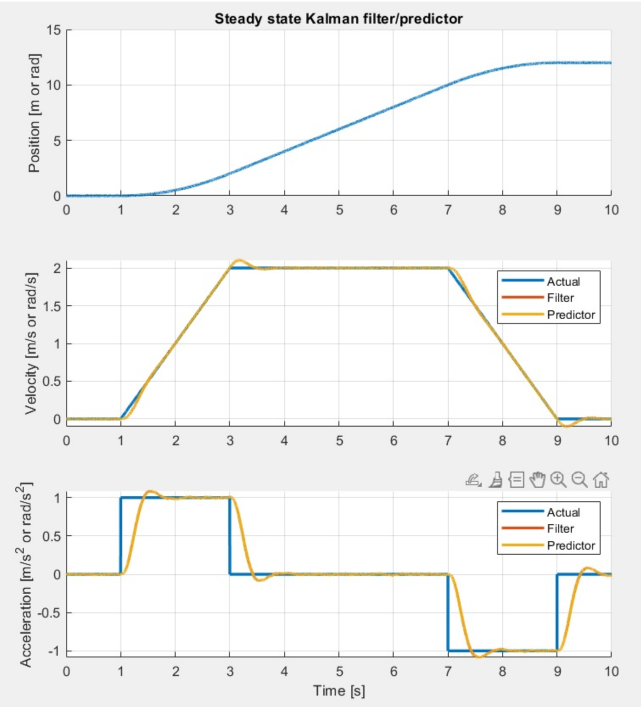

Figure 33: Steady state Kalman filter predictor Gaussian noise

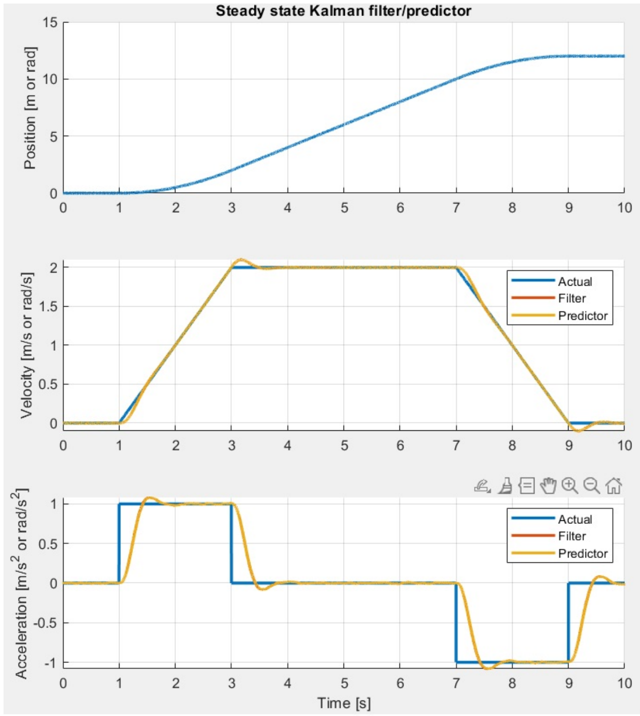

Figure 34: Steady state Kalman filter predictor Quantization Noise

3.3 Implement the Kalman smoother and estimate the velocity and acceleration from noisy position measurements

After compunting Kalman filter, we need to do a backward recursion starting from last estimation, at each iteration the values estimated from kalman filter will be smoothed.

In the forward step, a standard Kalman filtering procedure is carried out:

$$\hat{x}^f_{k+1|k+1} = A\hat{x}^f_{k|k} + K_{k+1}(y_{k+1} - CA\hat{x}^f_{k|k})$$

$$\hat{x}^f_{0|0} = \bar{x}_0$$

$$P^f_{0|0} = P_0$$

$$P^f_{k|k} = P^f_{k-1|k-1} - KCP^f_{k-1|k-1}$$

$$P^f_{k+1|k+1} = AP^f_{k|k}A^T + Q$$

Then in the backward step, smooothing is carried out as follows:

$$\hat{x}^s_{k|N} = \hat{x}^f_{k|k} + \bar{K}_k(\hat{x}^s_{k+1|N} - \hat{x}^f_{k+1|k})$$

$$\hat{x}^s_{N|N} = \hat{x}^f_{N|N}$$

Where:

$$\hat{K}_k = P^f_{k|k}A^T(P^f_{k+1|k})$$

and the conditional covariance matrix $P_{k|N}$ satisfies:

$$P_{k|N} = p^f_{k|k} + \hat{K} - k(P_{k+1|N} - P^f_{k+1|k})$$

$$P_{N|N} = P^f_{N|N}{}^{s}$$

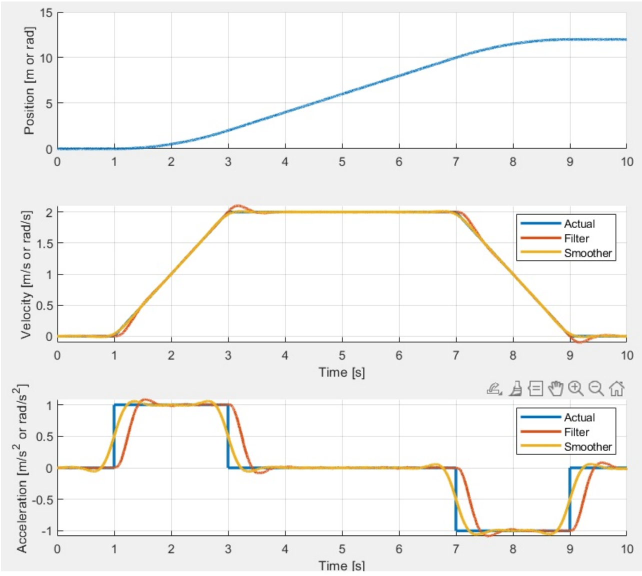

Figure 35: Kalman smoother Gaussian noise

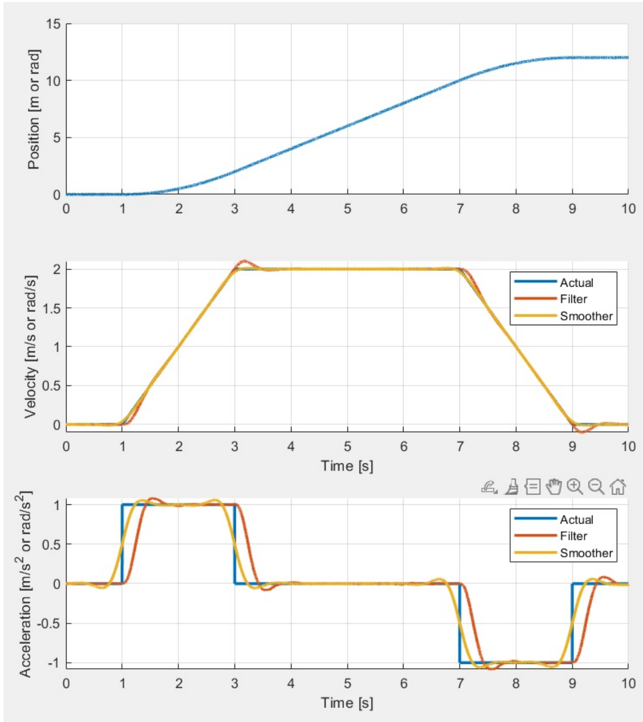

Figure 36: Kalman smoother Quantization Noise

4 Assignment 5

4.1 Identify the parameters k and τ (i.e. J and D) using the LS and the RLS on the DC motors data.

The differential equation describing the motors is $\tau\dot{\omega} + \omega(t) = kV(t)$. From data we can extract the voltage $V(t)$ and the position. In order to retrive velocities ω and acceleration $\dot{\omega}$, we use Kalman Filter and Smoother on sensor position.

Then we rewrite in a linear function the equation as $Y = X\theta$ with $Y = V, X = [\dot{\omega}\omega], \theta = \begin{bmatrix} \frac{\tau}{k} \\ \frac{1}{k} \end{bmatrix}$

- Using Least Square method we can estimate θ as $\hat{\theta} = (X^TX)^{-1}X^TY$. The result value are $k = 0.7012, \tau = 0.2587$.

- Using Recursive Least Square instead, for each new input we will estimate a different beta parameters, which leads to continuous innovation / correction of previus estimation.
  Also a discrete version of adaptive algotithm have been tried. This is an iterative procedure which updates params each iteration using the gradient of the estimation error $(e(k) = y(k) - x(k)\theta(k))$. The differential equation is $\hat{\theta} = \hat{\theta}(k-1) + T_s g x^t(k) e(k)$ where $g > 0$ is a constant. At the end we get $k = 0.0174, \tau = 0.0642$. We used also different values of λ so that is possible to see the difference between take more or less last results. $(\lambda 1 = 0.8, \lambda 2 = 0.975, \lambda 3 = 0.999)$.

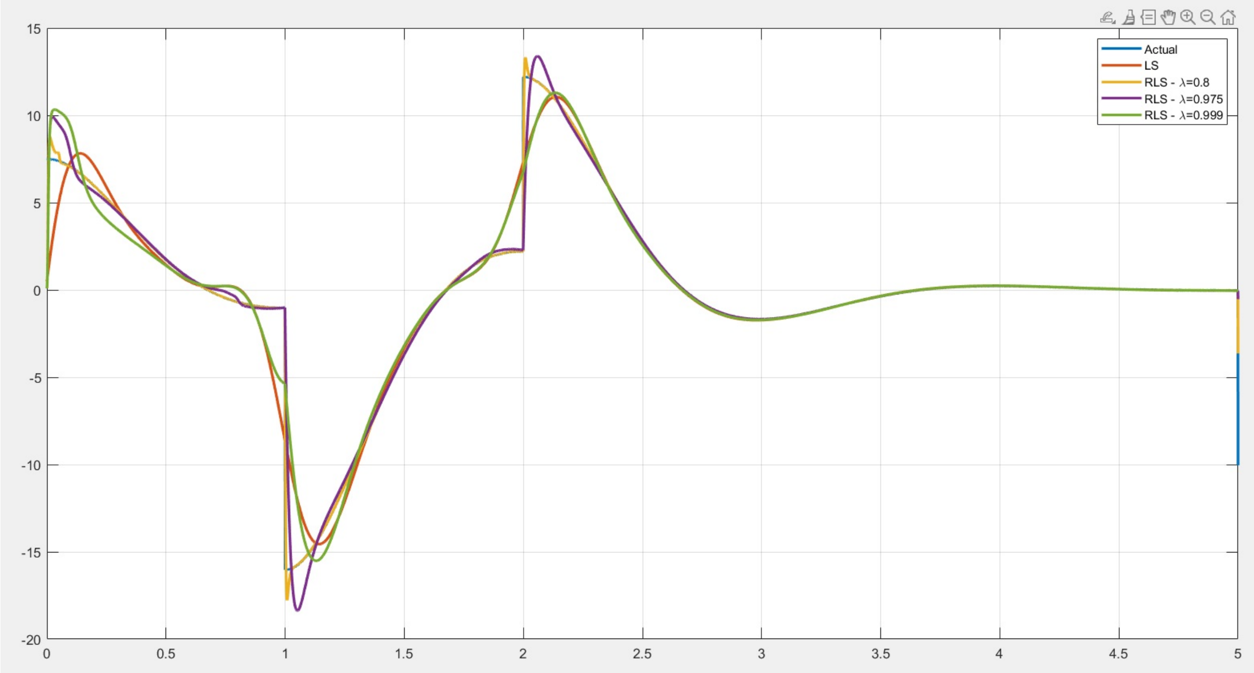

Figure 37: Recursive Least Square

5 Assignment 6

The environment was set at x = 5, the amplitude of the signals in free motion is 1 The architecture was evaluated with a delay of 10 time steps in the communication channel, the environment was modelled as spring-damper model with stiffness Ke = 200 and damping Be = 100.

5.1 Implement the Scattering-based bilateral teleoperation architecture for the Force-Position case, and Position-Position Case

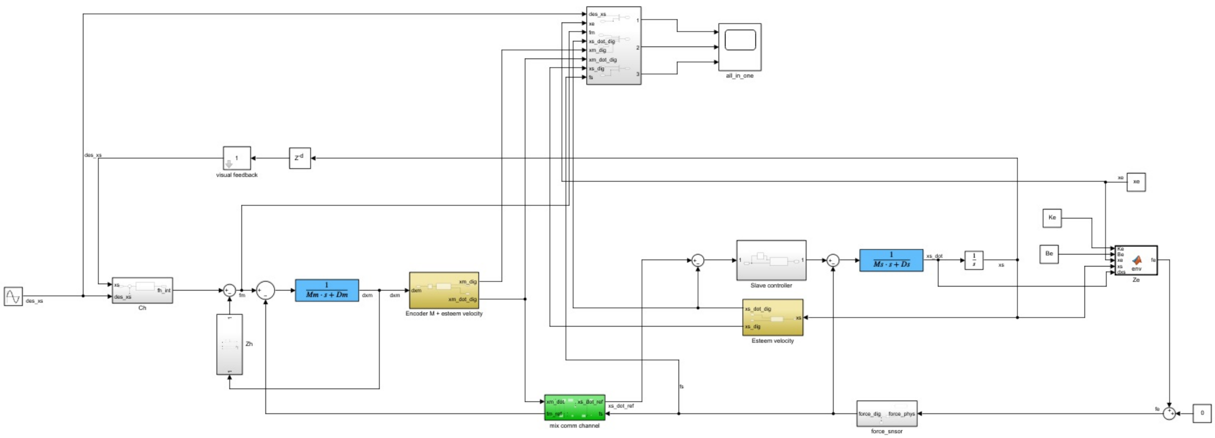

Figure 38: Simulink model Scattering Force Position

For this case the communication channel master will send its velocity which will become the velocity command for slave side where there will be a velocity controller, while slave will send force which will be subtracted to the user input. The mapping to and from scattering variables is as follow:

$$\begin{cases} u_l &= \sqrt{2b\dot{x}_l + v_l} \\ F_l &= b\dot{x}_l + \sqrt{2bv_l} \end{cases}$$

$$\begin{cases} u_r &= \sqrt{\frac{2}{b}F_r} - v_r \\ \dot{x}_r &= -\frac{1}{b}(F_r - \sqrt{2bv_l}) \end{cases}$$

Where the subscripts l and r refers to left(master) and right(slave) ports of communication channel. For simplicity the characteristic impedance b is set to 1.

At first image we can see that without contact with environment, the desired trajectory is followed and the velocities of two side also are aligned. To have a best align the damper of master (d_m) is set to 5 and damper of slave (d_s) set to 10.

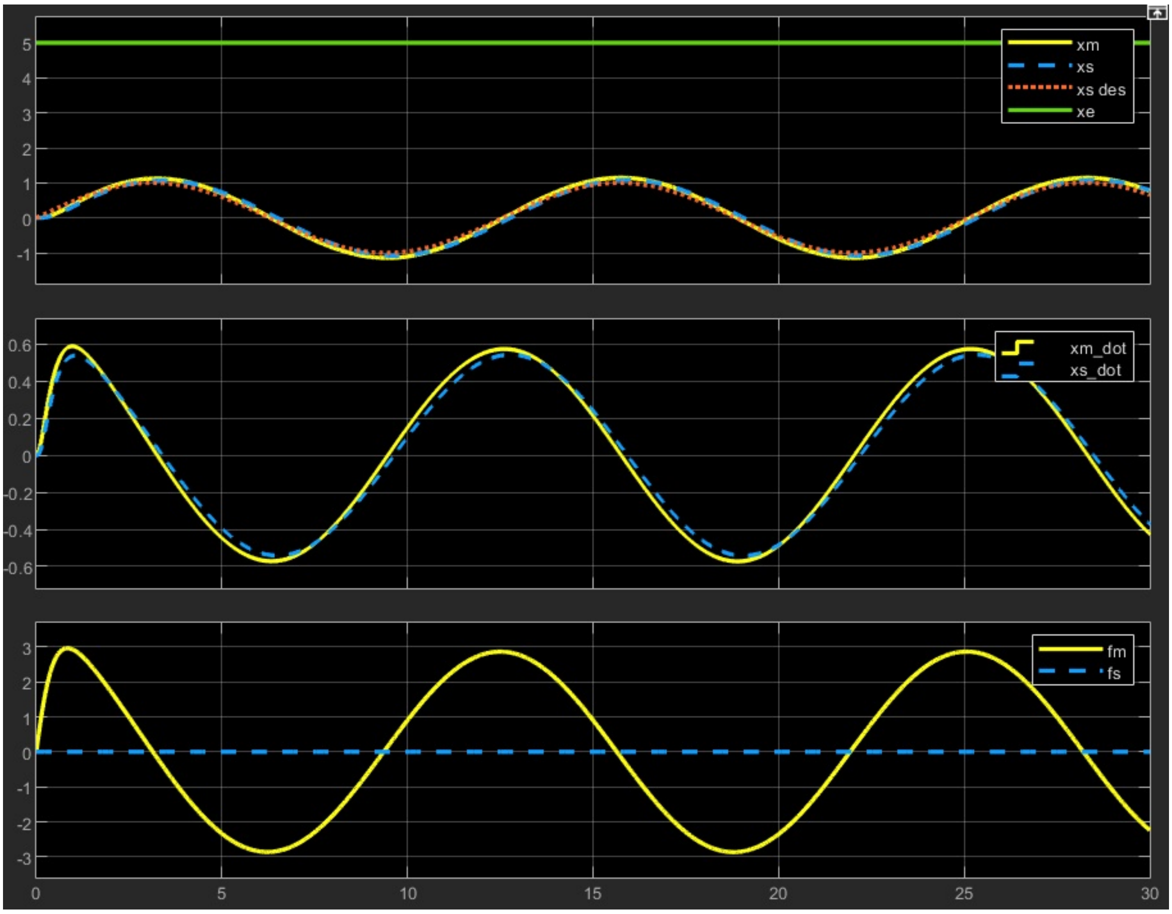

Figure 39: Scope: Scattering Force Position not in contact

In Figure 40 we can see how in contact with the environment x_e also the forces are aligned.

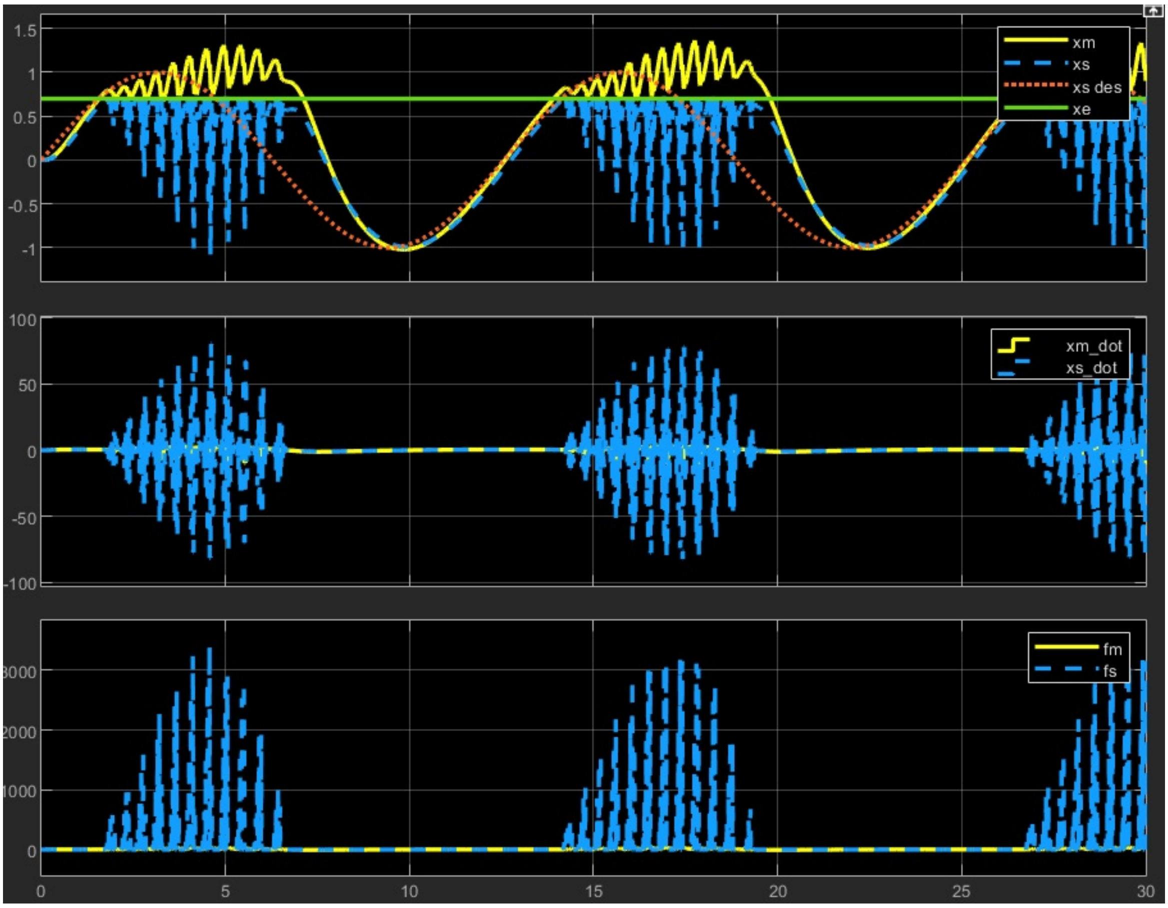

Figure 40: Scope: Scattering Force Position in contact

5.2 Implement the Scattering-based bilateral teleoperation architecture for the Position-Position case

For this case both master and slave will send their forces throught the communication channel, and thanks to the conversion to scattering variable, the output of the channel will be on both side the reference velocity. The mapping is as follow:

$$\begin{cases} u_l &= \sqrt{\frac{2}{b}}F_l - v_l \\ \dot{x}_l &= \frac{1}{b}(F_l - \sqrt{2b}v_l) \end{cases}$$

$$\begin{cases} u_r &= \sqrt{\frac{2}{b}}F_r - v_r \\ \dot{x}_r &= -\frac{1}{b}(F_r - \sqrt{2b}v_l) \end{cases}$$

First as before, first we see the model not in contact :

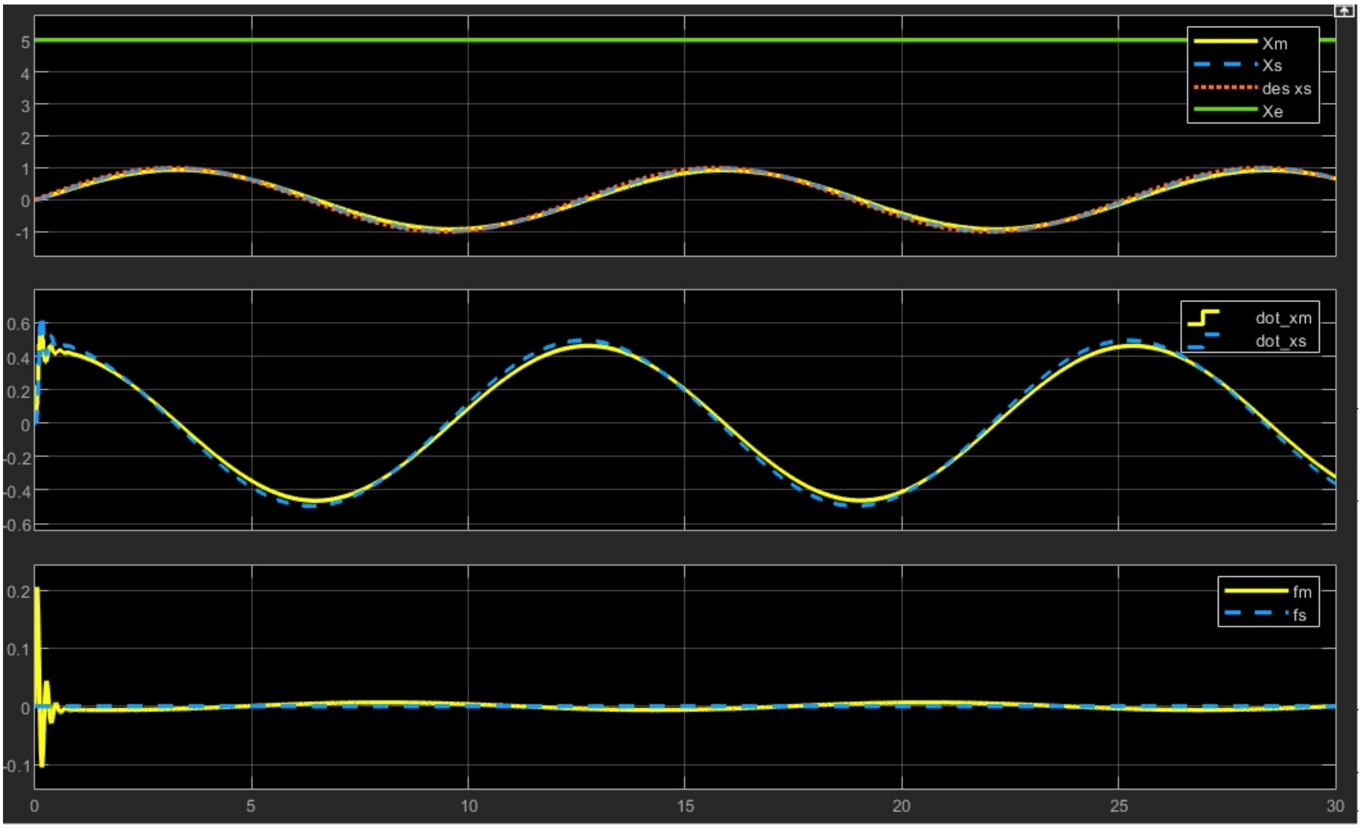

Figure 41: Scope: Scattering Position Position not in contact

Then, the slave in contact with the environment. In Figure 43 is shown how is the signal when there is noisy position readings and the velocity is estimated using Kalman filter. The noise variance is $1 \cdot 10^{-6}$ and we can see how it is affecting the velocity estimation.

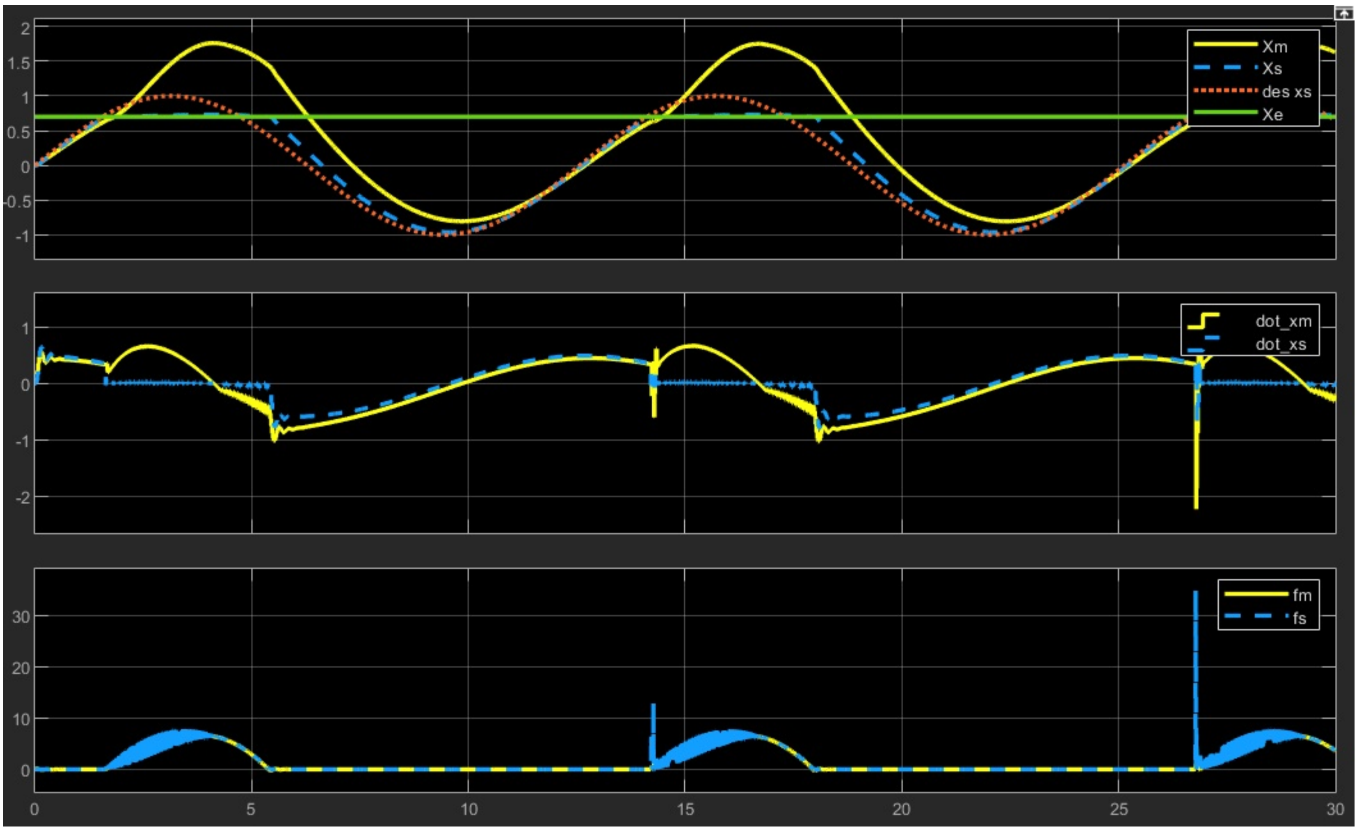

Figure 42: Scope: Scattering Position Position in contact

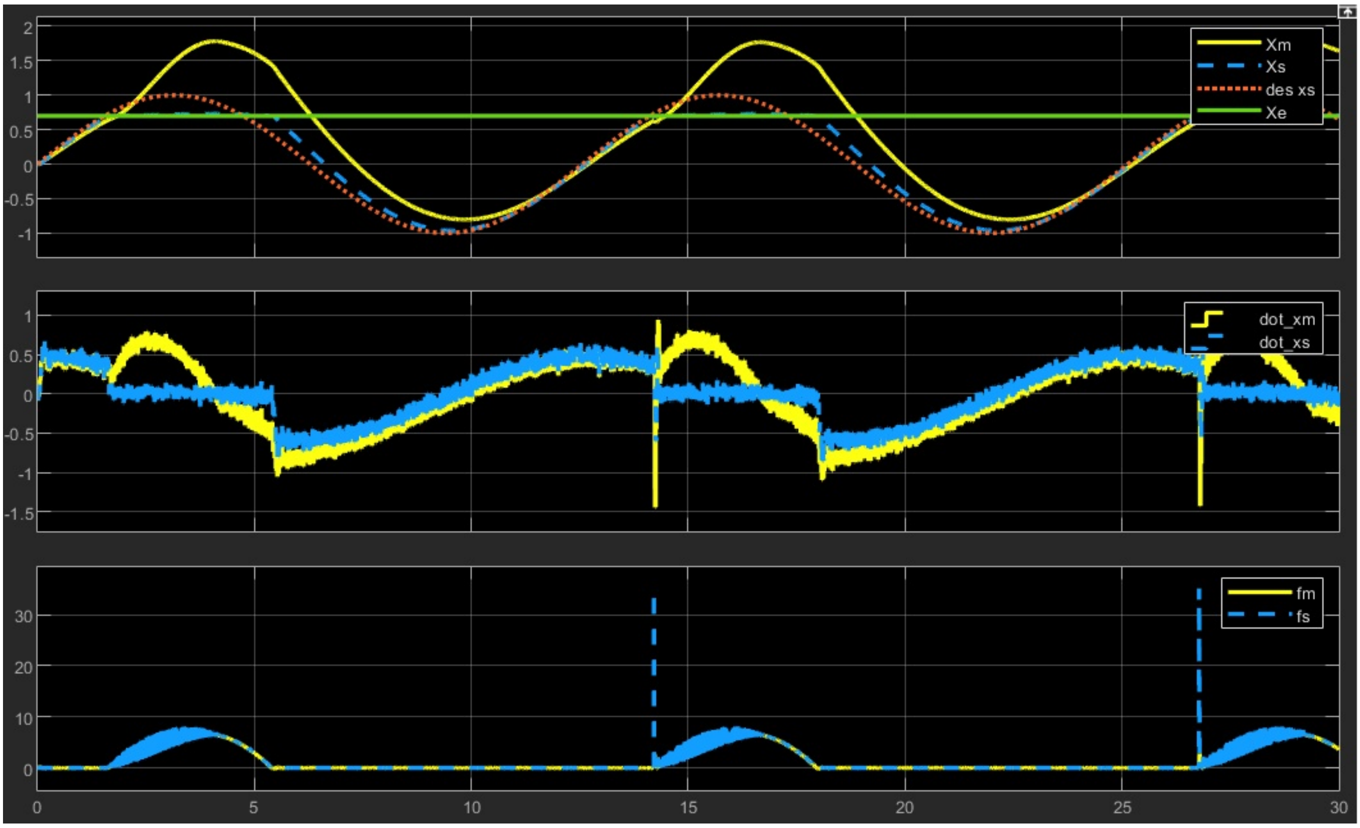

Figure 43: Scope: Scattering Position Position in contact with noise

6 Assignment 7

The tank based bilateral teleoperation architecture implements a two-layer approach: The hierarchical top layer is used to implement a strategy that addresses the desired transparency, and the lower layer ensures that no “virtual”energy is generated. The architecture does not need any about the time delay in the communication channel which can be also time-varying. The main concept of this architecture is the presence of two communication energy storage tanks from which the motions of both the slave and the master are powered.

- Transparency layer : control structure which is implemented to provide the best possible transparency of the tele-manipulation chain, taking into account all available information about the system, the environment, and the task the user is executing
- Passivity layer: given the command computed by the transparency layer, this layer contains an algorithm to maintain passivity of the total system

The level of these tanks can be interpreted as a tight energy budget from which controlled movements can be powered and which are being replenished by the user at the master side when necessary or if possible/desired also at the slave side.

At each discrete instant, the energy at each side is computed by the following equations

$$H(k) = H(\hat{k}) + H_{+}(k) - \delta H_{I}(k)$$

Where $H(\hat{k})$ is the energy from the previus time instant, $H_{+}(k)$ is the sum of energy received from the other side, and $H_{I}(k)$ is the energy used at time k computed in the following manner :

$$\delta H_{I}(k) = \tau(k)\delta_{q}(k)$$

The energy level for the next time instead is computed by

$$H(k \hat{+} 1) = H(K) - H_{(k)}$$

6.1 Implement the Tank-based bilateral teleoperation architecture for the Force-Position case

In a first scenario we start with a little amount of energy in the master tank.

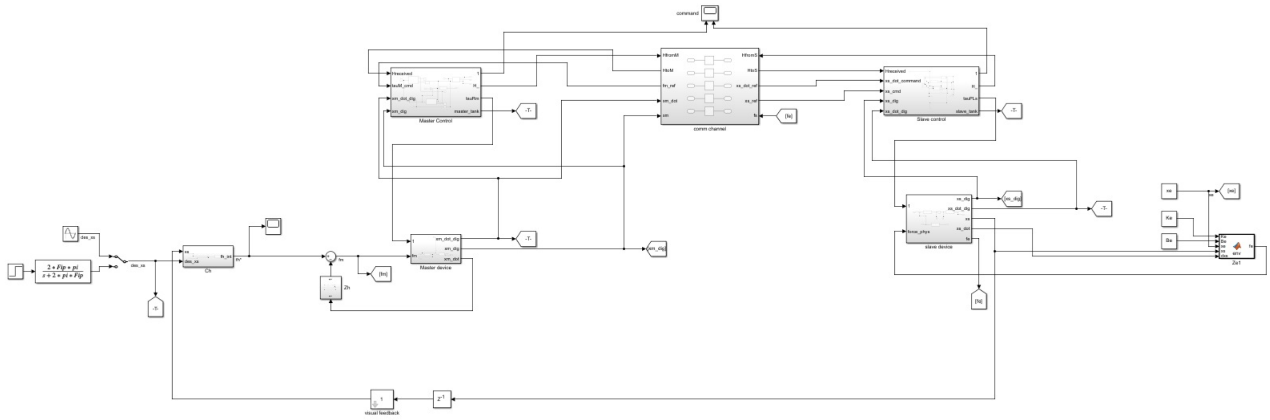

Figure 44: Simulink : Tank-based bilateral teleoperation architecture

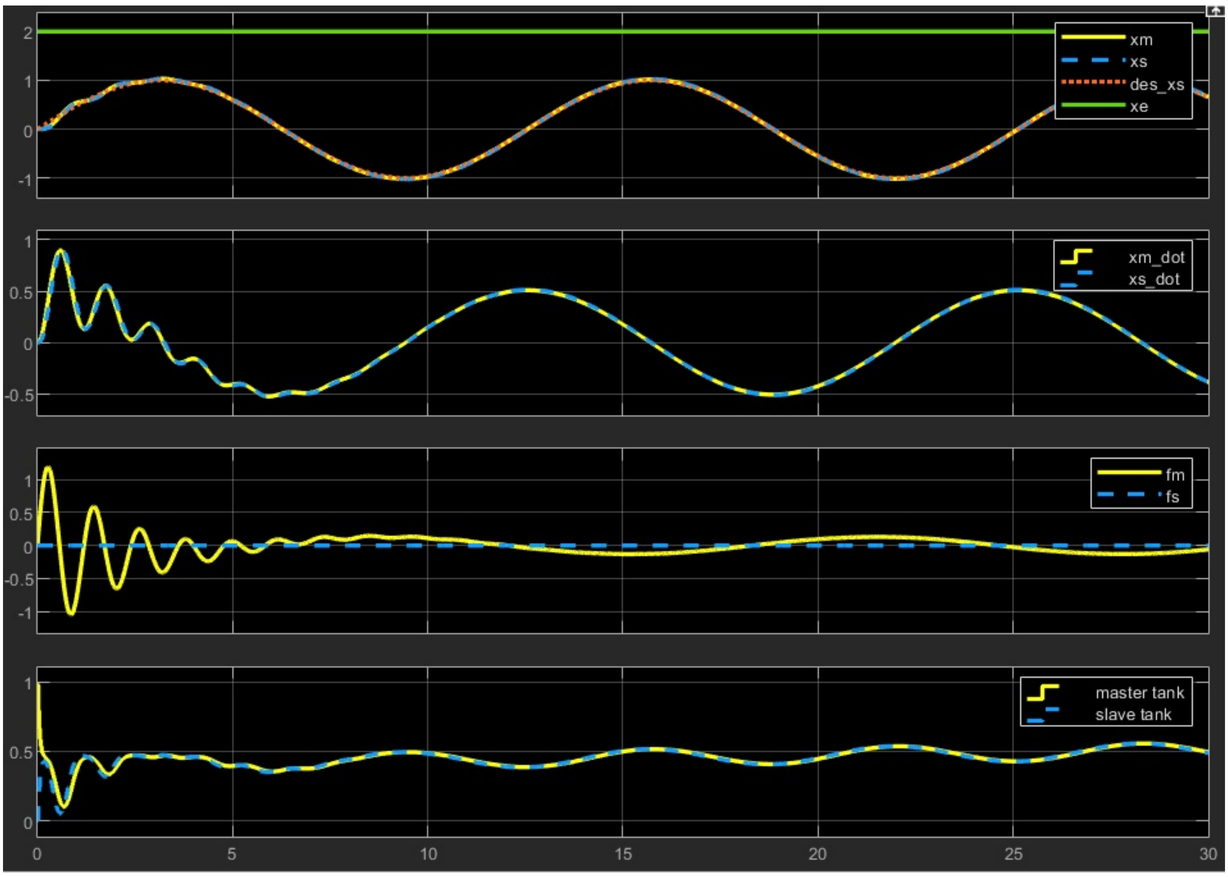

Figure 45: Tank-based bilateral force-position not in contact

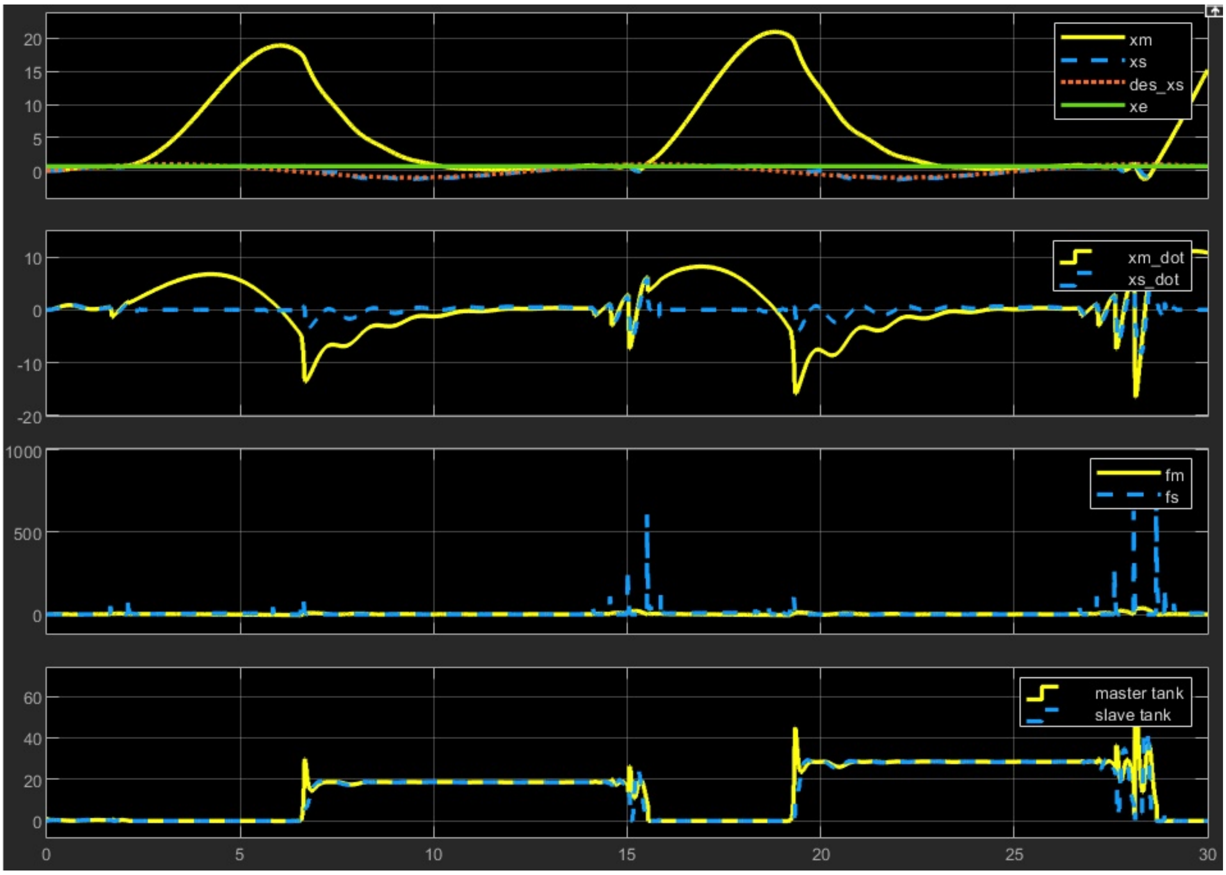

Figure 46: Tank-based bilateral force-position in contact

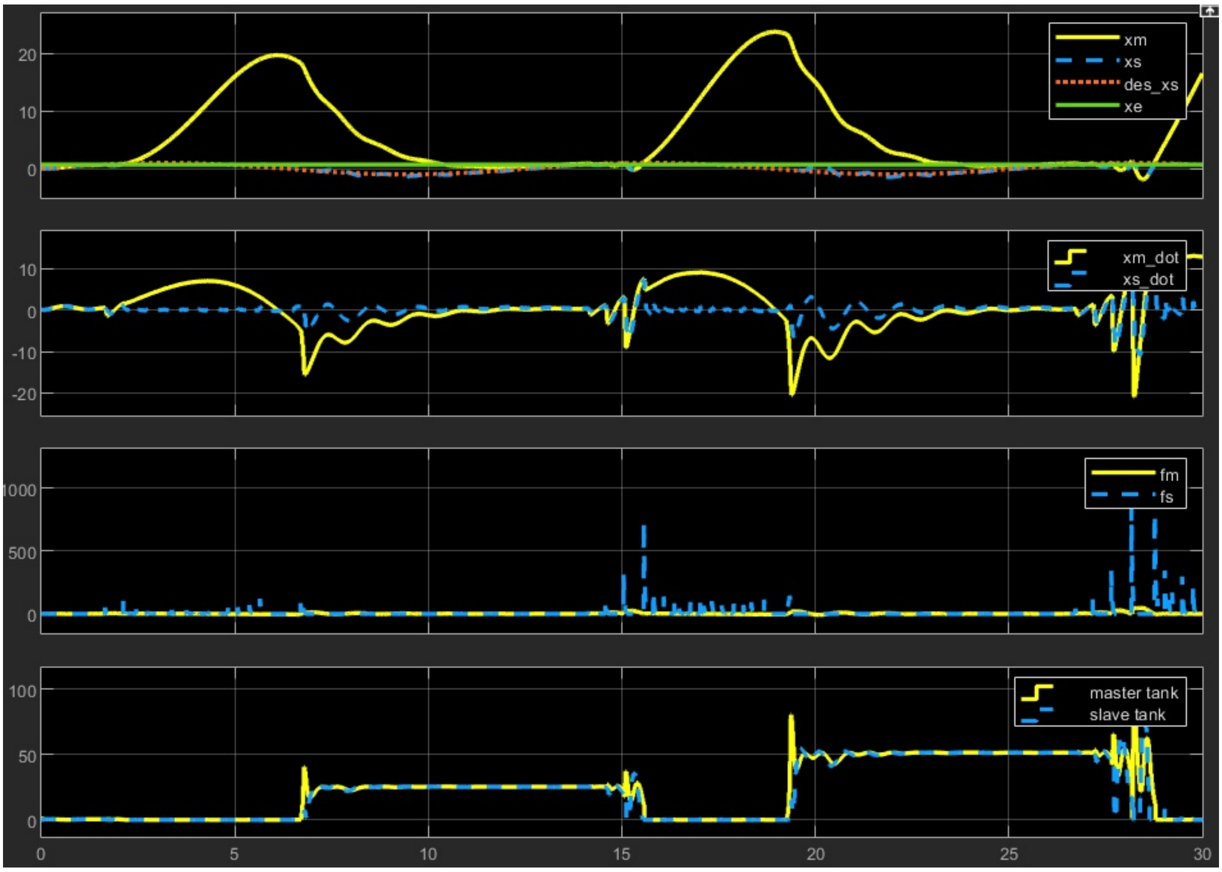

Figure 47: Tank-based bilateral force-position in contact with noise

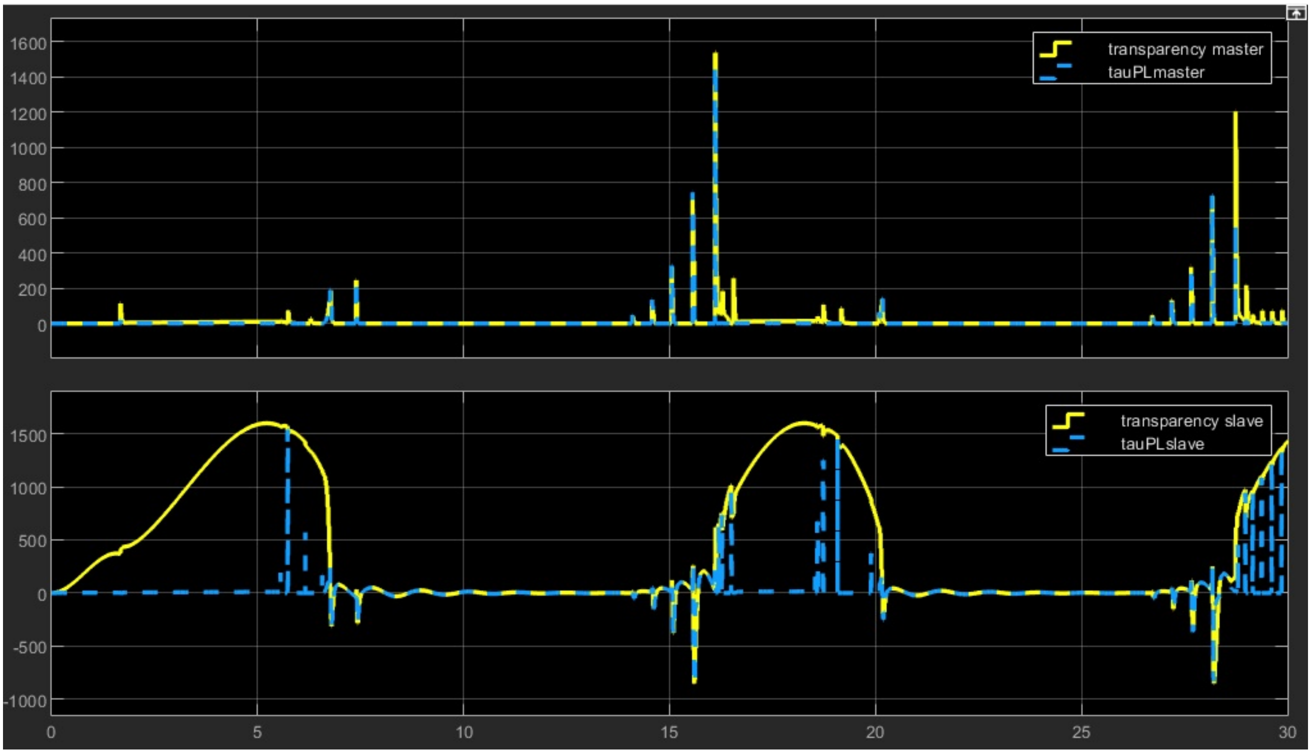

Figure 48: Tank-based bilateral force-position not initial tank energy

6.2 Implement the Tank-based bilateral teleoperation architecture for the Position-Position case

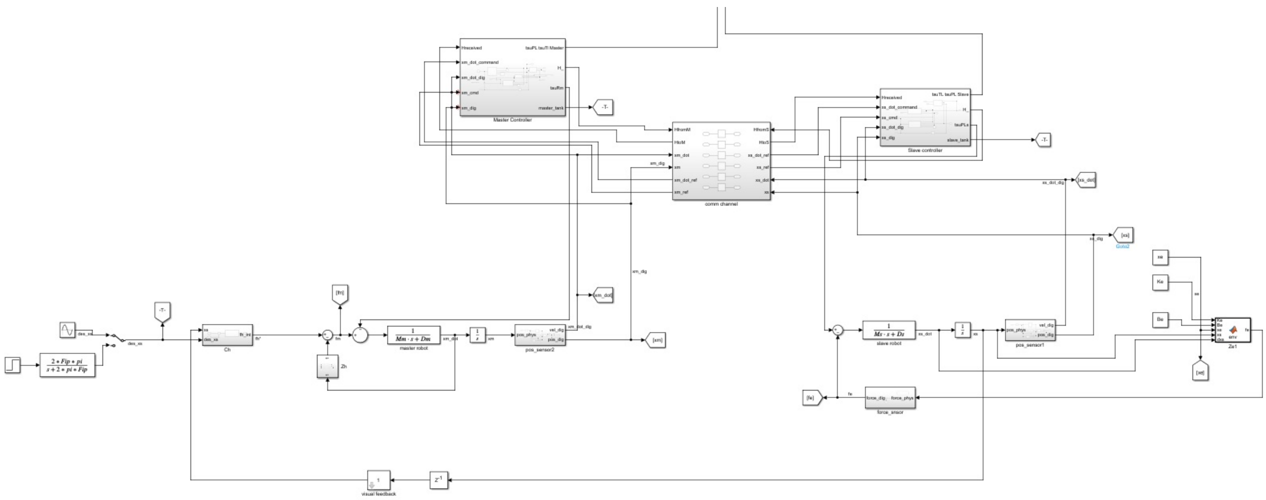

Figure 49: Tank-based bilateral position-position simulink model

Also here the first scenario will start with a small energy in master tank $Hm_{init} = 1$. In this case in the communication channel both sides send position and velocities, which means that both transparency layers are implementing a position controller.

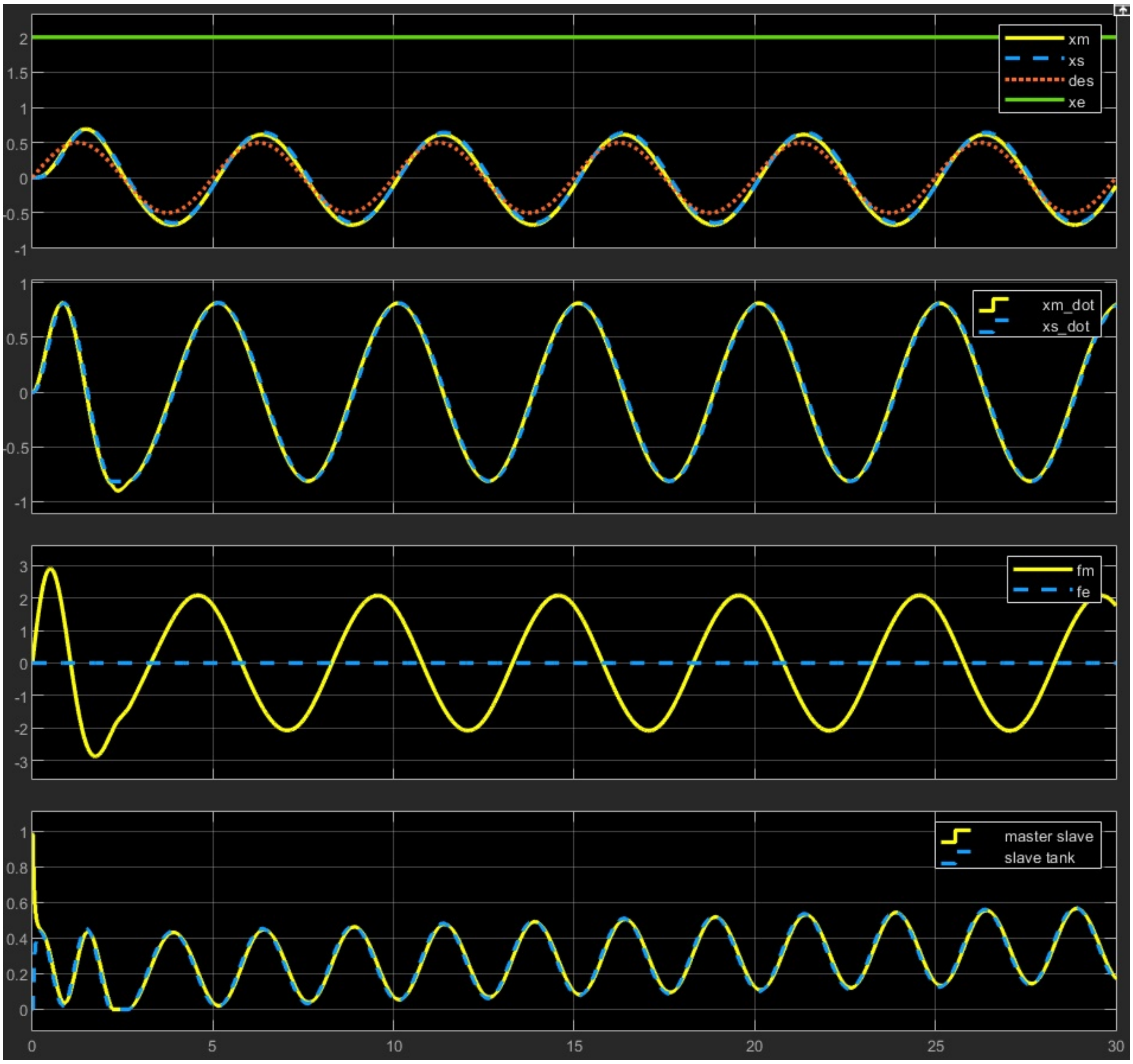

Figure 50: Tank-based bilateral position-position scope not in contact with initial energy in master tank

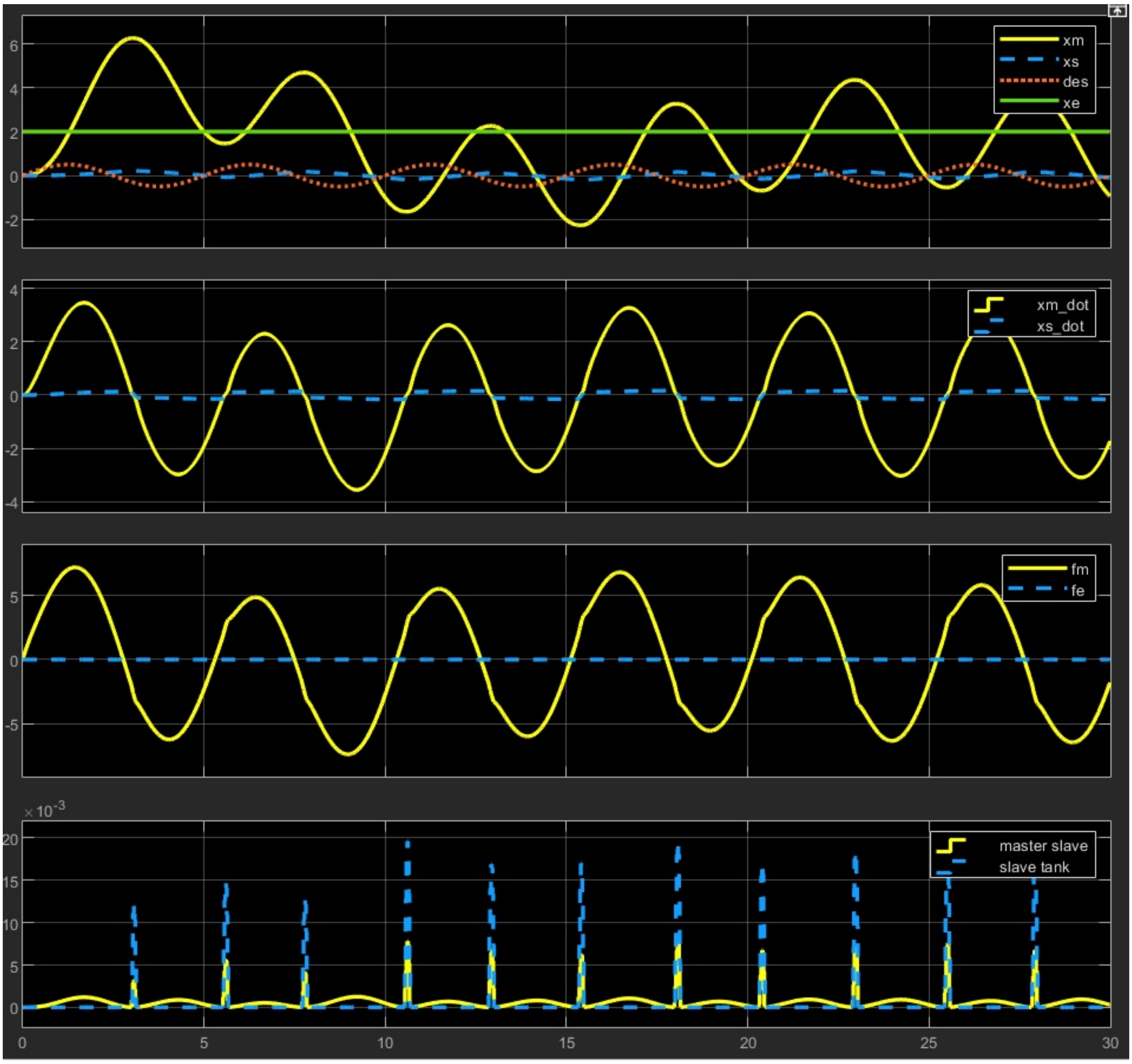

Figure 51: Tank-based bilateral position-position scope not in contact without initial energy in master tank

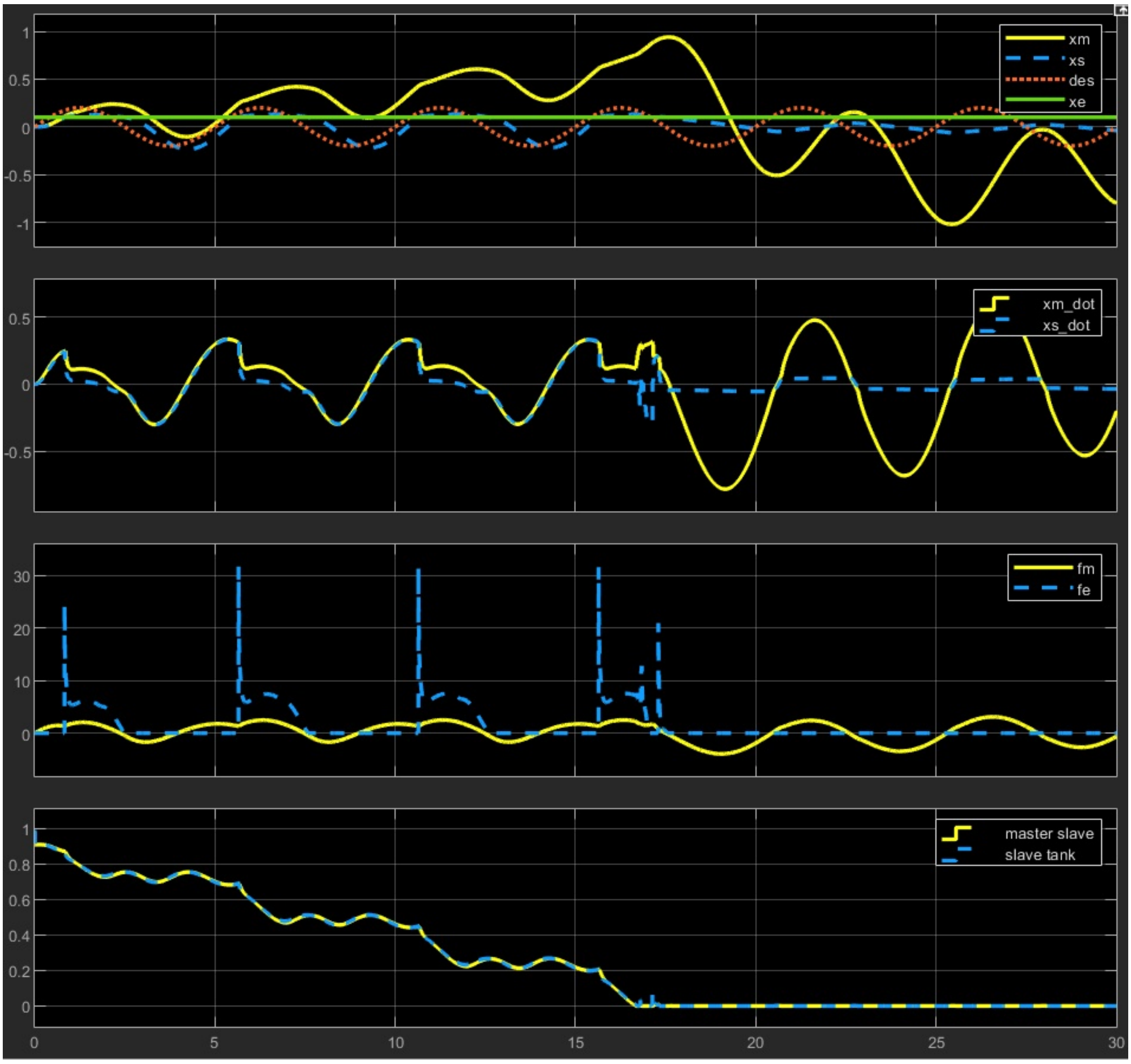

Figure 52: Tank-based bilateral position-position scope in contact with initial energy in master tank

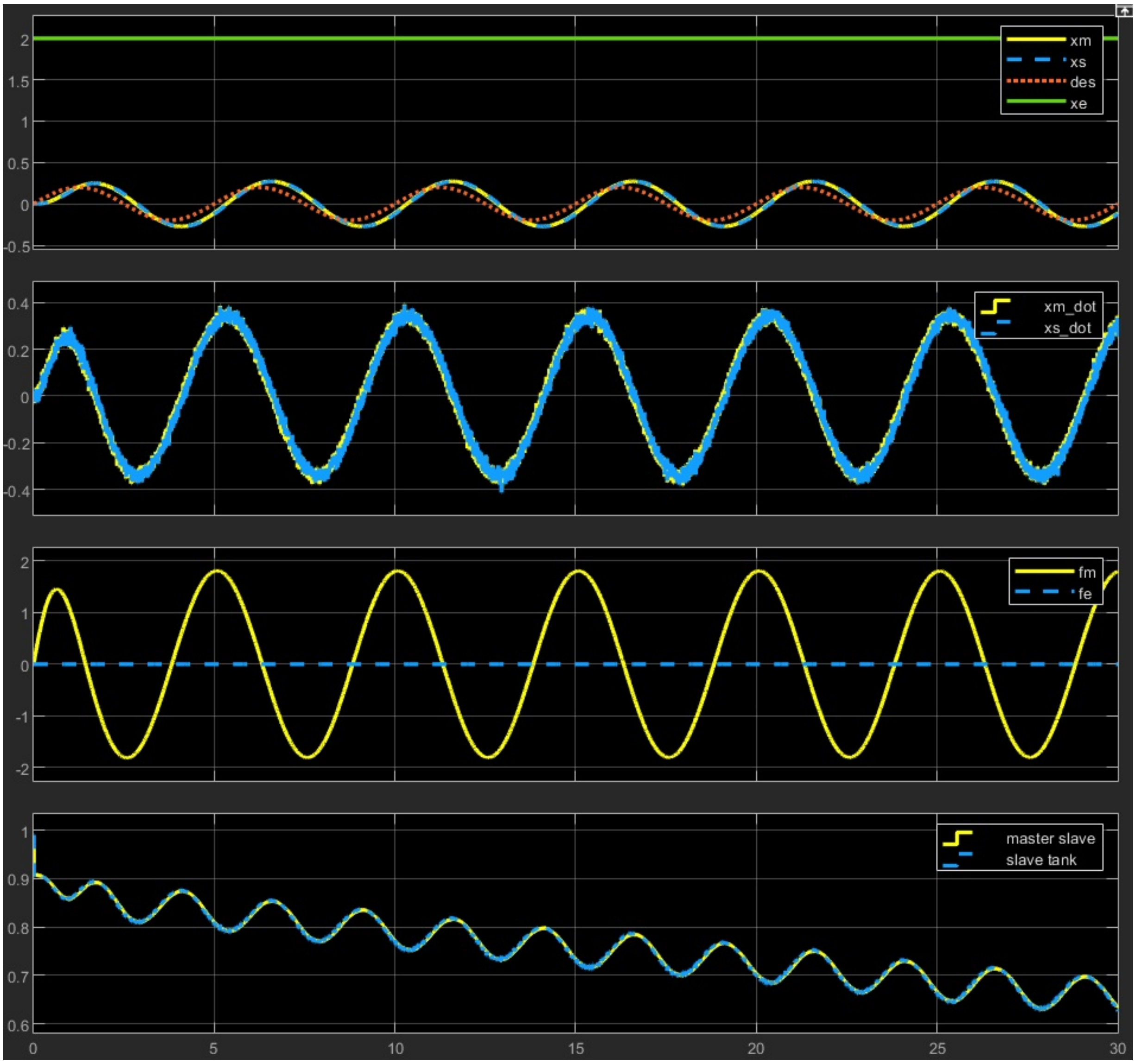

Figure 53: Tank-based bilateral position-position scope not in contact with initial energy in master tank and noise

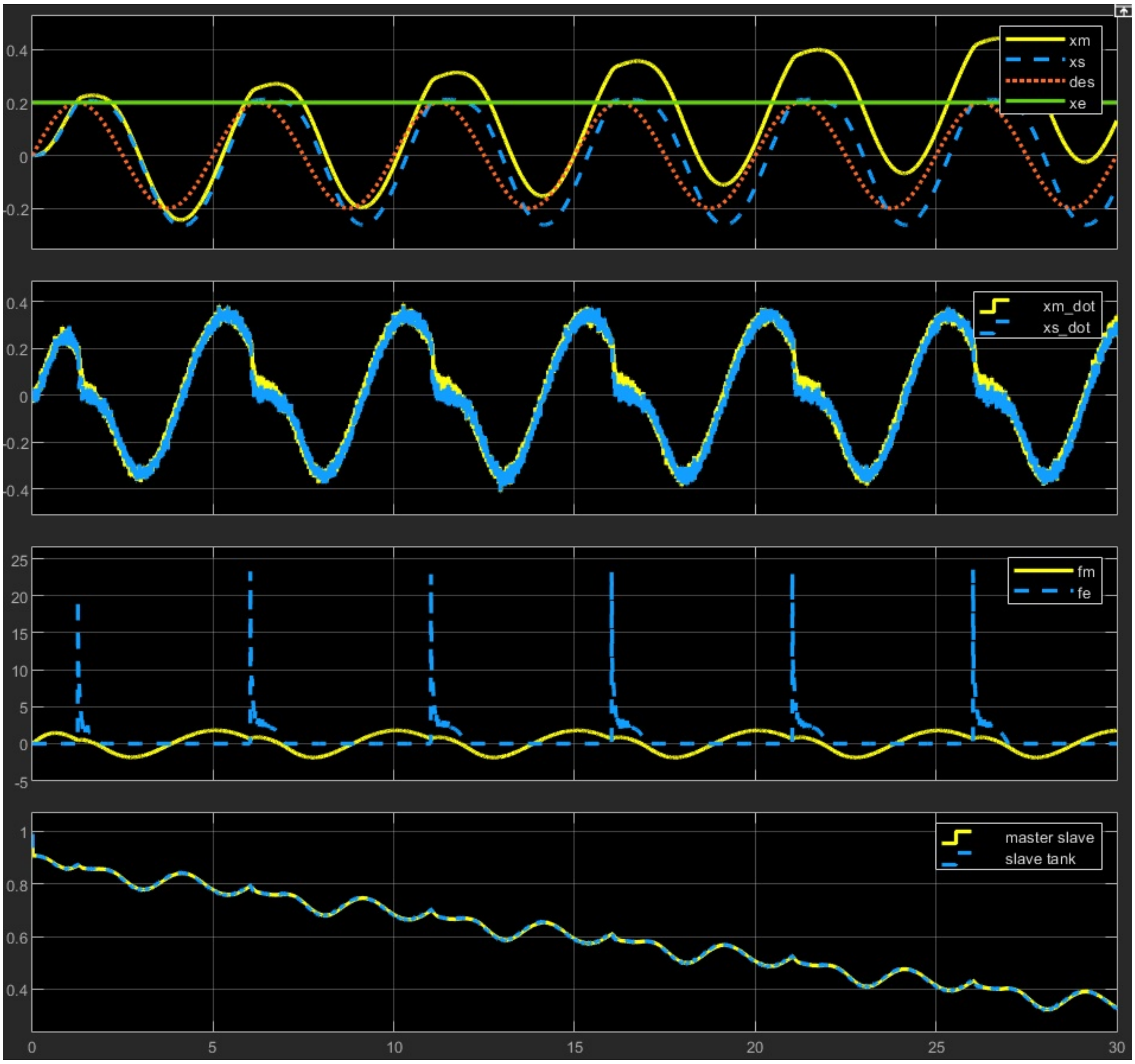

Figure 54: Tank-based bilateral position-position scope in contact with initial energy in master tank and noise and noise