**Two-Link Planar Control**

# Arm Description

* Two-link planar, also called double compound pendulum.
* The arm is assumed to move in X-Y plane and the joints revolve about Z-axis.
* Gravity is assumed to work in Y-axis
* Links are assumed to be thin bricks with dimensions () where  is the longest side.
* Each link has a revolute joint in the bottom side.
* Actuators are not modeled

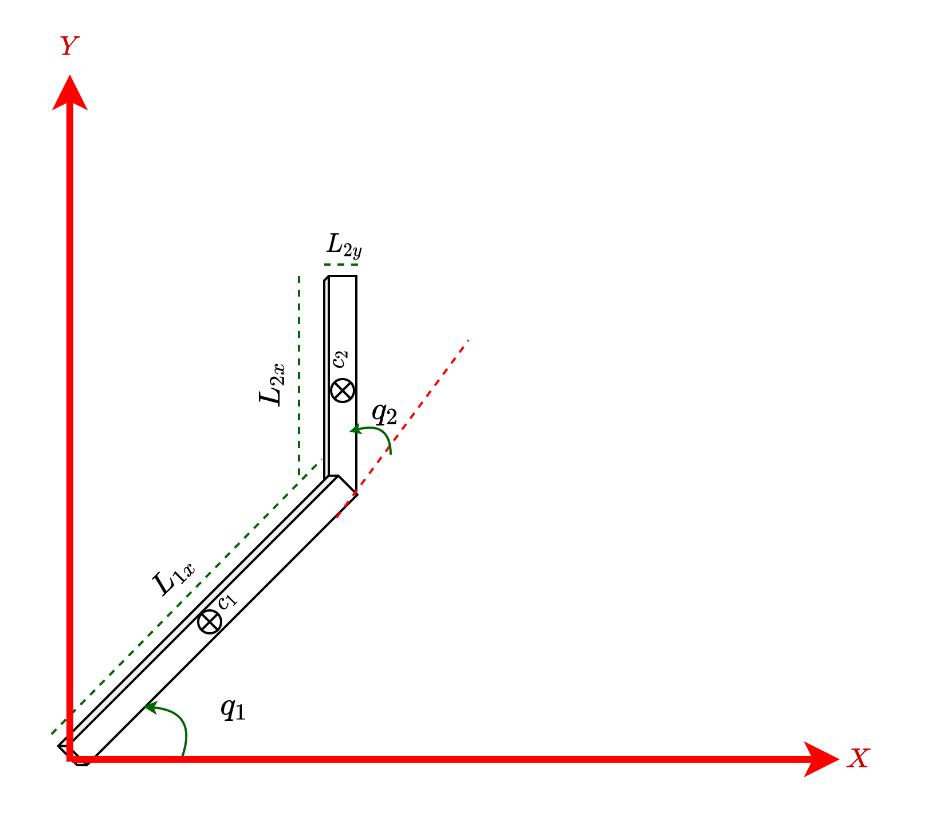


Figure 1: Two-link Planar

# Arm kinematics

## Center of mass position

# Arm dynamics

* Each link has a mass .
* The center of mass of each link is in the middle of the link
* Each link has a viscous friction .

## Inertia about center of mass:

## Lagrange formulation:

* Generalized coordinates:
* Refer to 1\_twolink\_planar\_dynamics.mlx Matlab live script for detailed derivation
* General equation of motion:



Where **M** is the inertia matrix, **C** has the velocity terms (Coriolis and centipedal forces), **K** are any terms related to joint positions, **g** is the gravity terms, **Q** has the generalized forces including friction, and is the torque of joint i.

# Arm Control

## PID Joint control (Decentralized)

* A PID controller for each joint
* coupling terms are assumed to be disturbance
* Zero steady state error with I term and sufficient time to converge
* Asymptotically stable assuming positive controller gains and limited integral gain. It goes unstable if integral gain is very large.
* Performance degrades when tracking reference trajectories at high speed and acceleration

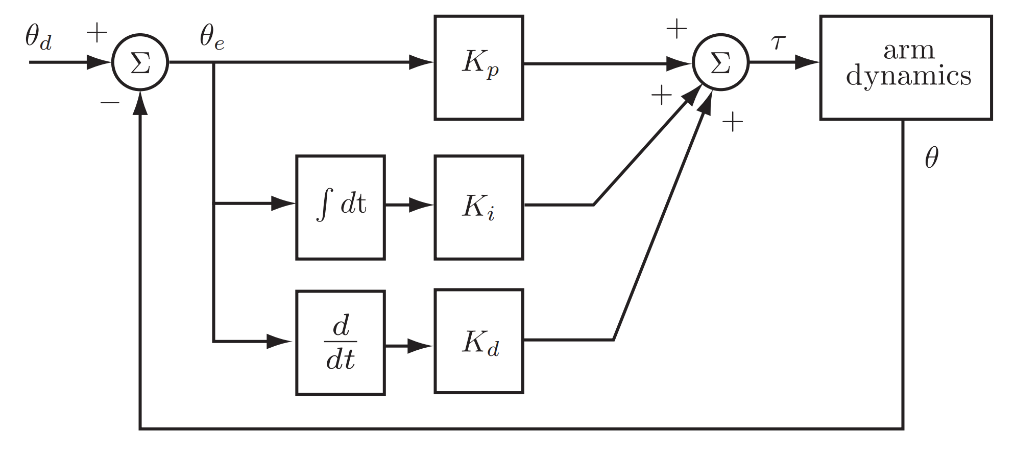


Figure 2: PID joint Control

## Full state feedback control (Decentralized)

The equation of motion can be linearized and converted to state space as follows:

* Decompose the inertia matrix into a constant diagonal terms and residual terms
* Consider the disturbance equation as:
* Select state vector as
* Taking derivatives of state vector
* State equations:
* Friction terms:
* Decentralized simplified model
* Convert to state space and use pole placement to synthesize a control law
* Disturbance rejection can be improved by adding the disturbance terms to the control law
* Multiple implementation strategies are possible:
  + Scaled reference: the gain matrix K is placed in the feedback route and another scaling gain scales the input.
  + State error: the gain is multiplied by state error
  + Integral action: same as scaled reference with additional integral term to the input

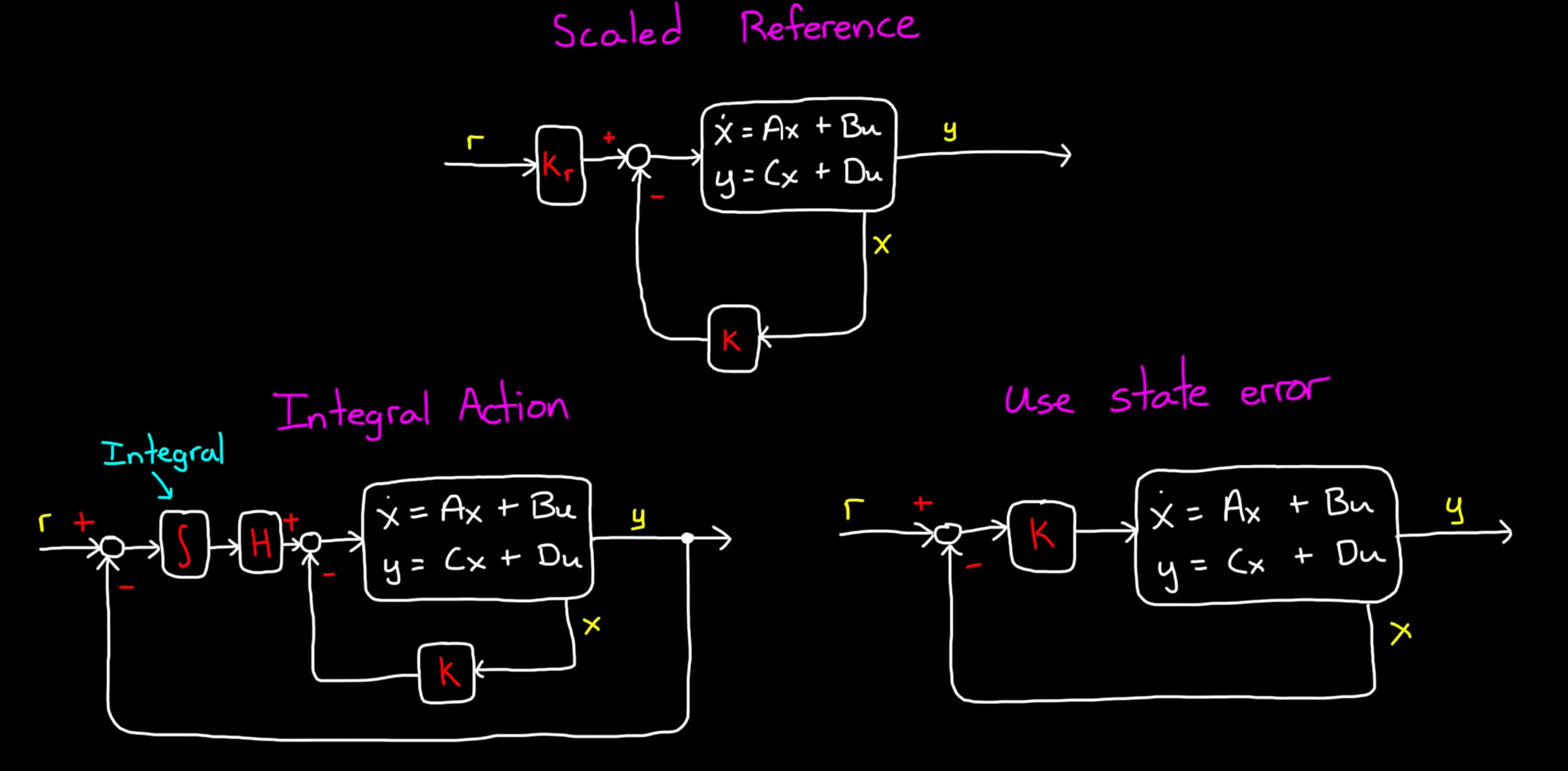


Figure 3: Full state feedback control strategies (source: [Matlab youtube channel video](https://www.youtube.com/watch?v=E_RDCFOlJx4&list=RDCMUCgdHSFcXvkN6O3NXvif0-pA&index=29))

* The gain matrix can be found in multiple ways:
  + Pole Placement method
  + Linear Quadratic regulator (LQR)

### Pole Placement method

* Desired poles of the system are used to calculate the gain matrix K
* The command *K=place(A,B,p);* in Matlab returns the gain matrix K for the provided poles *p.*

### Linear Quadratic regulator (LQR)

LQR formulates an optimization cost function that is minimized to find the optimal gain matrix K with minimum cost. The cost function is composed of weighted sum of performance and actuator effort over time.

is the matrix to panelize bad performance. It is positive definite matrix and is typically diagonal.

is the matrix to panelize actuator effort. It is positive definite matrix and is typically diagonal.

* The command *K=lqr(A,B,Q,R);* in Matlab solves the optimization and returns the optimal gain matrix K.



Figure 4: Full state feedback control with disturbance compensation (State error format)

## PD+Gravity compensation Joint control (Centralized)

* Takes into account the gravity terms
* It is computationally more expensive than PID control
* It is good when we don't have a good model of the full dynamics, or the equations may be too computationally expensive to calculate at servo rate.
* It performs well when the desired velocities and accelerations are small
* PD controller can also be replaced by PID controller
* Performance can be improved if friction effects are also added
* The linear controller can either use position derivative or velocity for the D term, the derivative is used in the simulations.

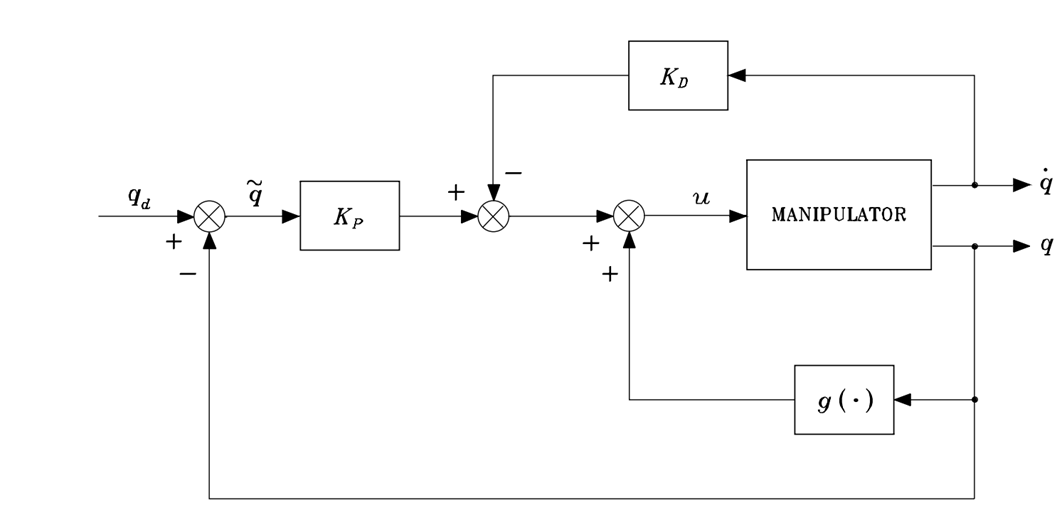


Figure 5: PD+Gravity compensation Control

## Inverse dynamics control (Centralized)

* Also called Computed Torque Feedforward Control!
* Very sensitive to dynamics modelling.
* Feedback linearization + linear controller (PD or PID)
* The linear controller can either use position derivative or velocity for the D term, the derivative is used in the simulations.

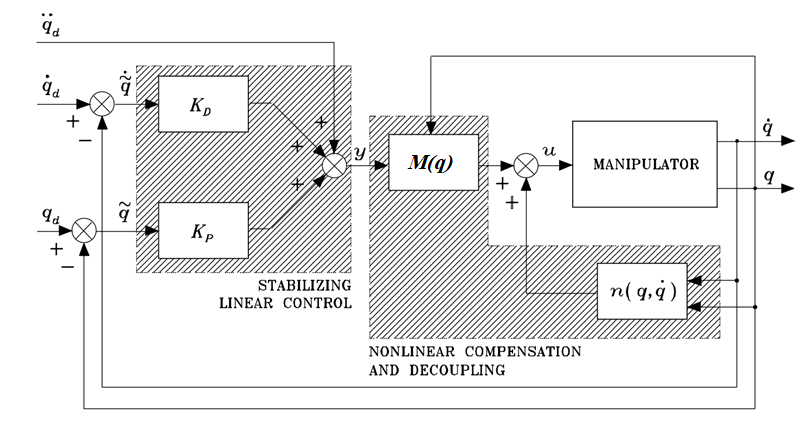


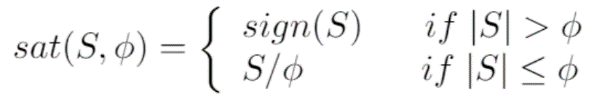
Figure 6: Inverse Dynamics control

## Sliding Mode control

* There are many ways to add robustness to the controller, one way is to add sliding mode terms.
* There are many variants of sliding mode control, however we will focus on a basic form here that is model-free.
* The sliding surface is chosen as  where  are the error and derivative of error
* A simple control form is to add the sliding control term  to an existing control law. For example, modifying LQR+disturbance control law to

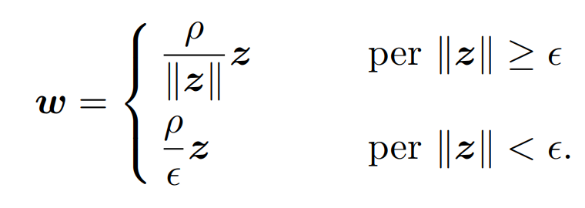


* To reduce chattering, we can smooth the discontinuity with a saturation layer as follows:

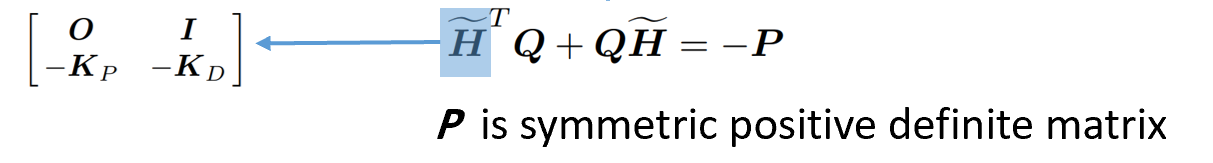


## Robust control (Centralized)

* A form of sliding mode control formulated in chapter 8 of Robotics Modelling, Planning and Control by Bruno Siciliano, Lorenzo Sciavicco, Luigi Villani, and Giuseppe Oriolo
* Extends inverse dynamics control to deal with imperfect model compensation and intentional simplification in inverse dynamics computation



* Requires solving the Lyapunov equation below to get **Q**



*  and **P** are free parameters. e.g. 

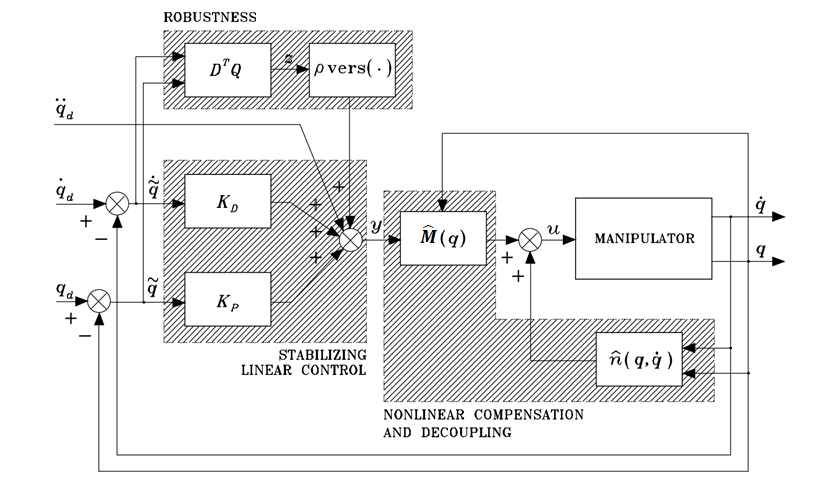


Figure 7: Robust control

# Simulation parameters

## Robot parameters

Table 1: Robot parameter

|  |  |
| --- | --- |
| Parameter | Value |
| g | 9.8066 (in Y-direction) |
| Link density | 3700 kg/m3 |
| Link 1 dimensions (x\*y\*z) | 0.3m\*0.05m\*0.05m |
| Link 2 dimensions (x\*y\*z) | 0.15m\*0.05m\*0.05m |
| m1 | 2.775 kg |
| m2 | 1.3875 kg |
| b1 | 0.01 |
| b2 | 0.01 |
| Initial values ( | All zeros |

## Realistic simulation parameters

* Gaussian measurement noise is added to joint position with standard deviation equal to position sensor precision
* Only joint noise is added since it will propagate through the integrals to velocity and acceleration.
* A first-order low-pass filter is added to smooth joint position measurements (introduces some lag)
* All controllers are designed in discrete form.

Table 2: Simulation parameters (realistic)

|  |  |
| --- | --- |
| Parameter | Value |
| Sampling time | 1ms |
| Maximum velocity |  |
| Maximum joint 1 torque | 20 Nm |
| Maximum joint 2 torque | 10 Nm |
| Joint position precision |  |
| Robot parameters deviation (mass, length, viscosity)  (Used in the dynamics terms of some controllers) |  |

## Measurement filter design

Low pass filter design:

* Joint position:
* Joint velocity:
* Joint acceleration:

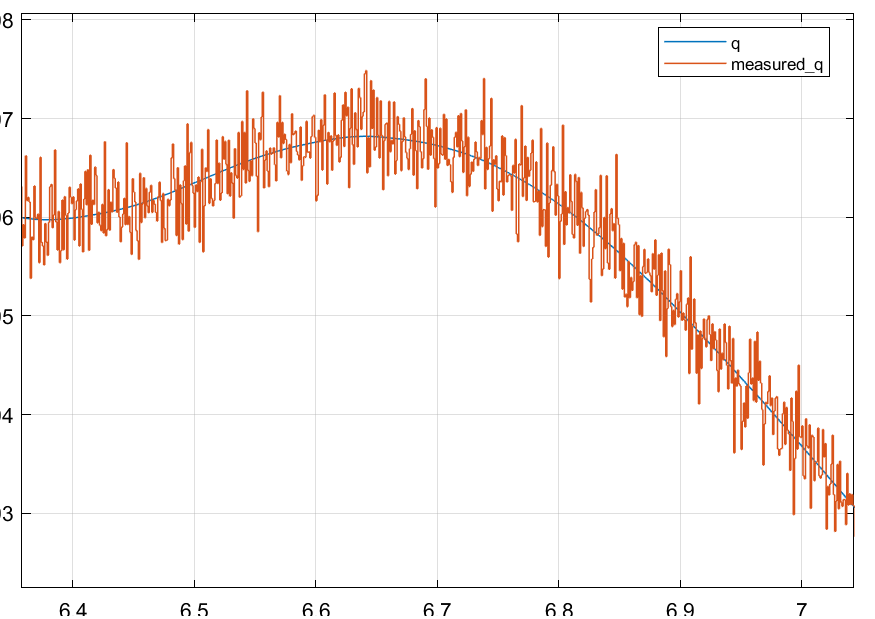


Figure 8: Actual joint angle (degrees) vs measured after gaussian noise

# Simulation results – Ideal vs realistic

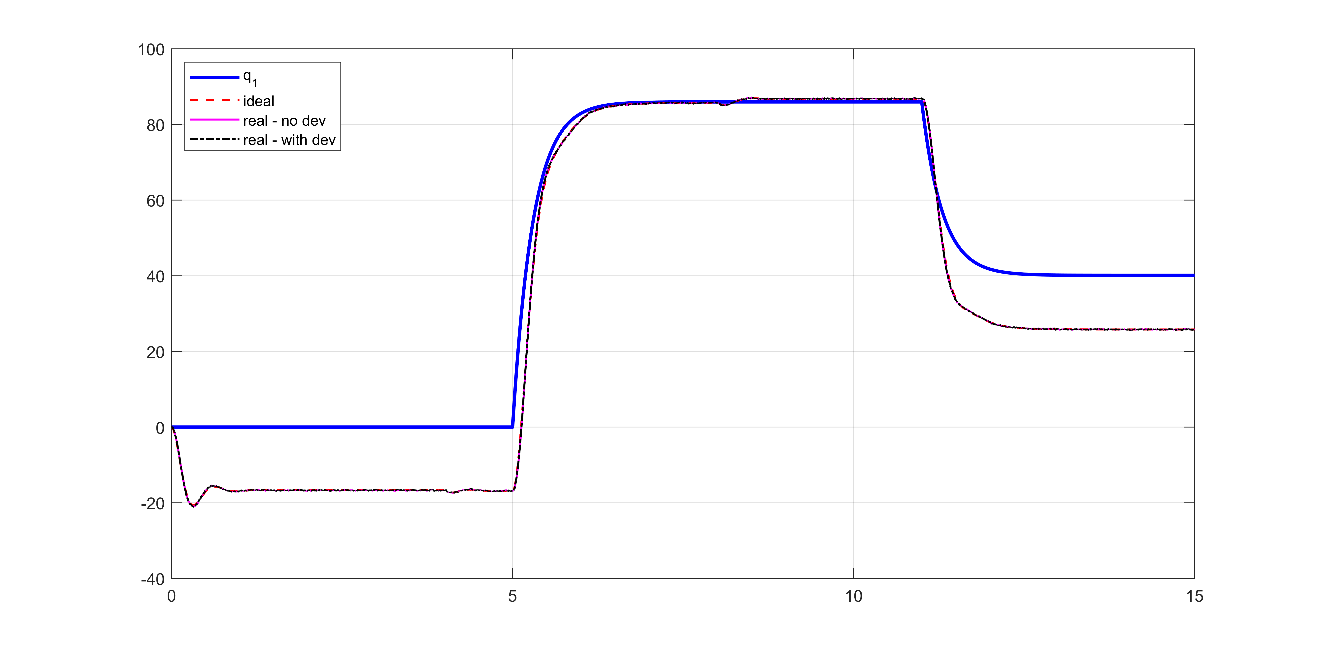
This section shows the response of each controller to a step and sine wave trajectories when the system is ideal and when it sensor noise is included (realistic). There is also additional simulation to look at the effects of dynamic parameter variations (mass, length and viscous coefficient).

## PD Joint control

Table 3: PD gains

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Ideal case | | | | Realistic case | | | |
| Parameter | Value | Parameter | Value | Parameter | Value | Parameter | Value |
|  | 30 |  | 7 |  | 30 |  | 7 |
|  | 2.5 |  | 0.5 |  | 2.5 |  | 0.5 |

### Step response



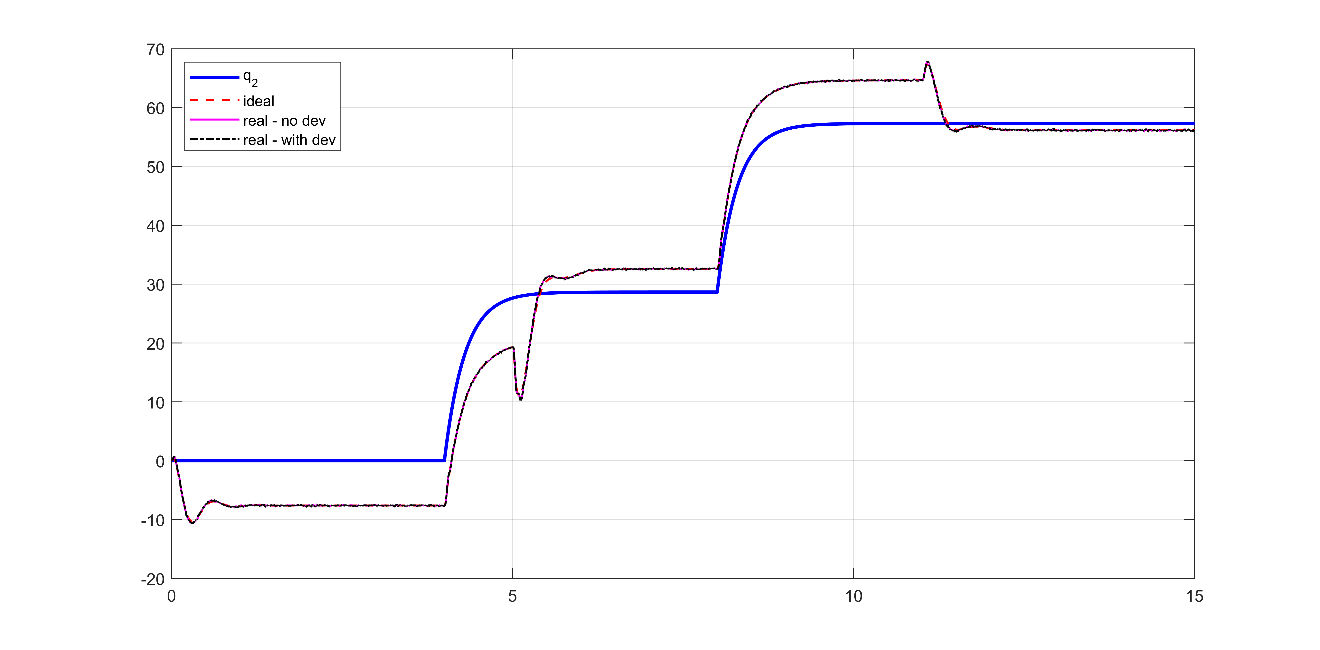
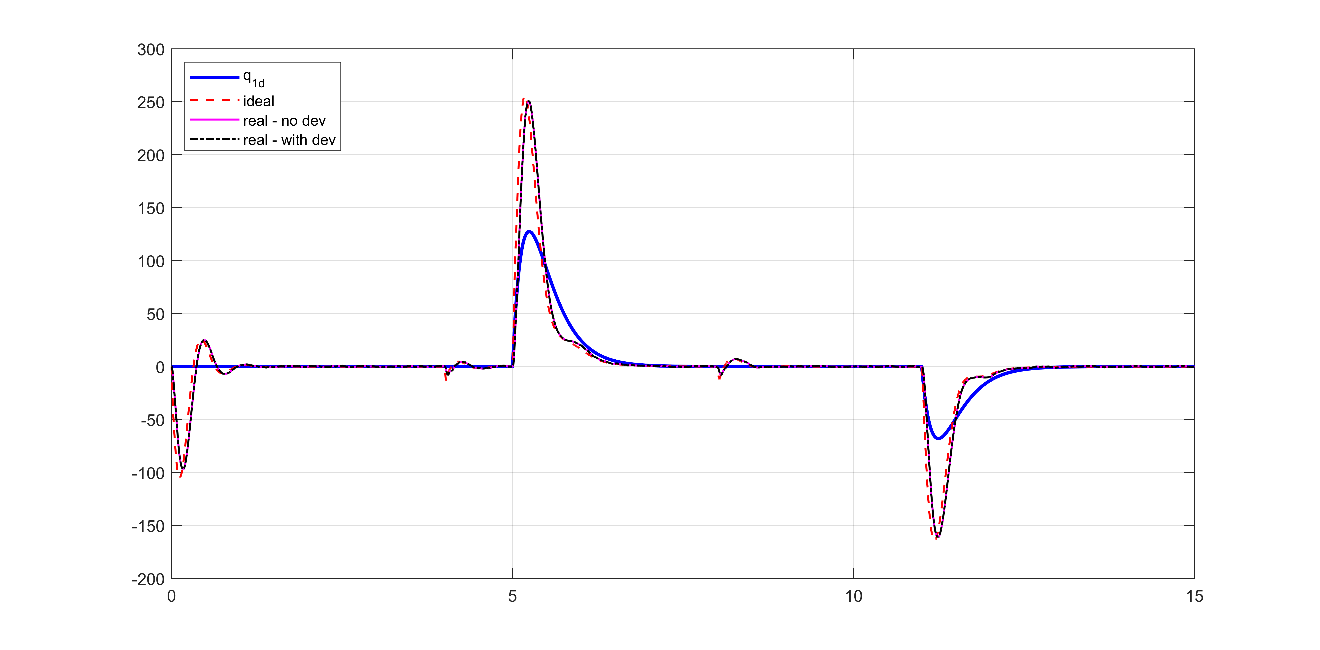


Figure 9: PD control step response - joint position



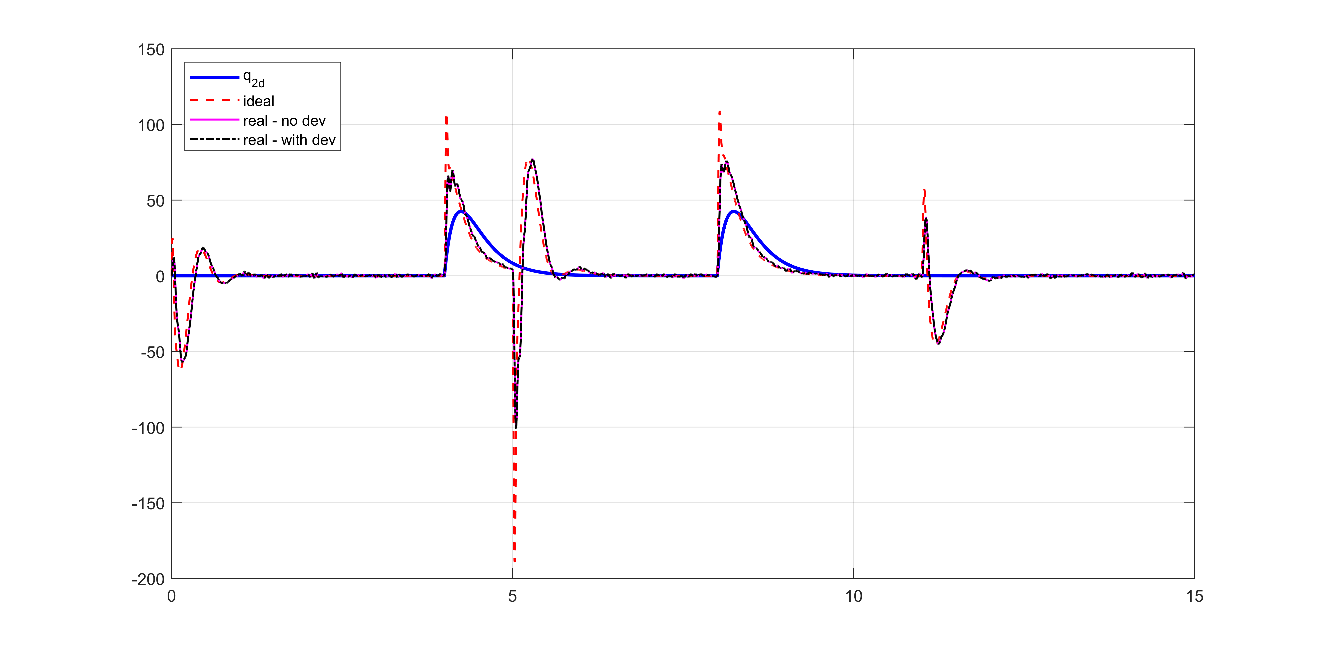


Figure 10: PD control step response - joint velocity

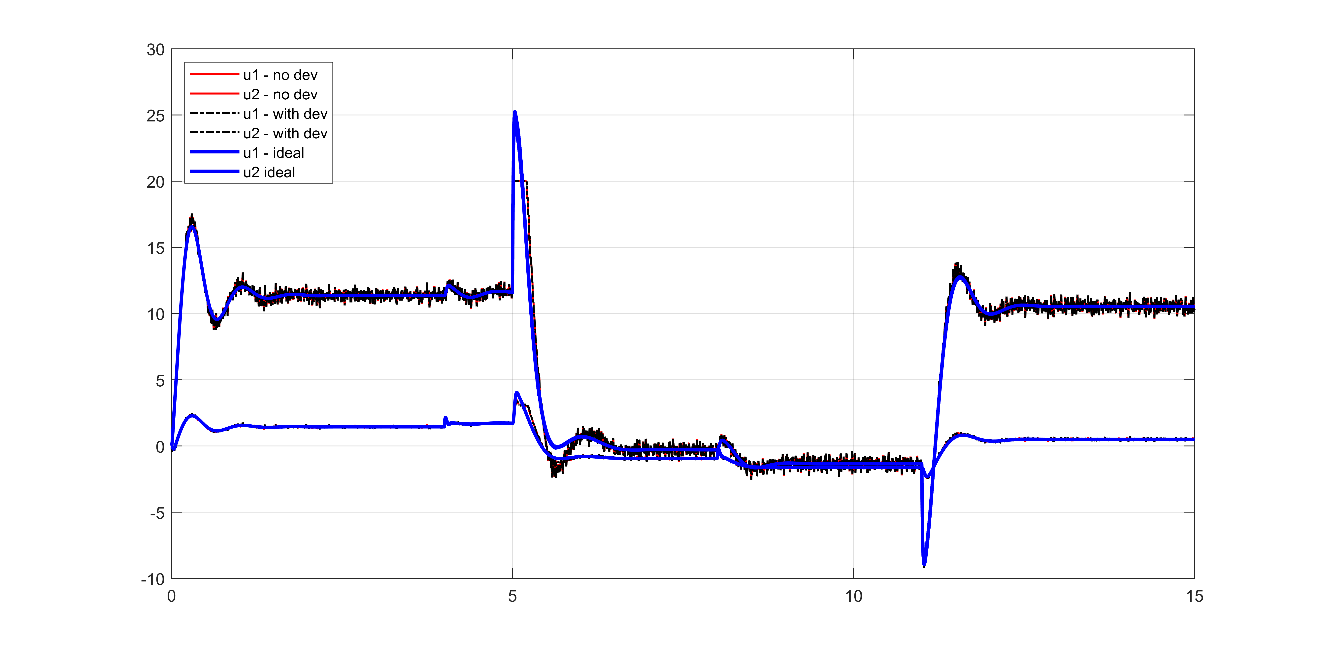
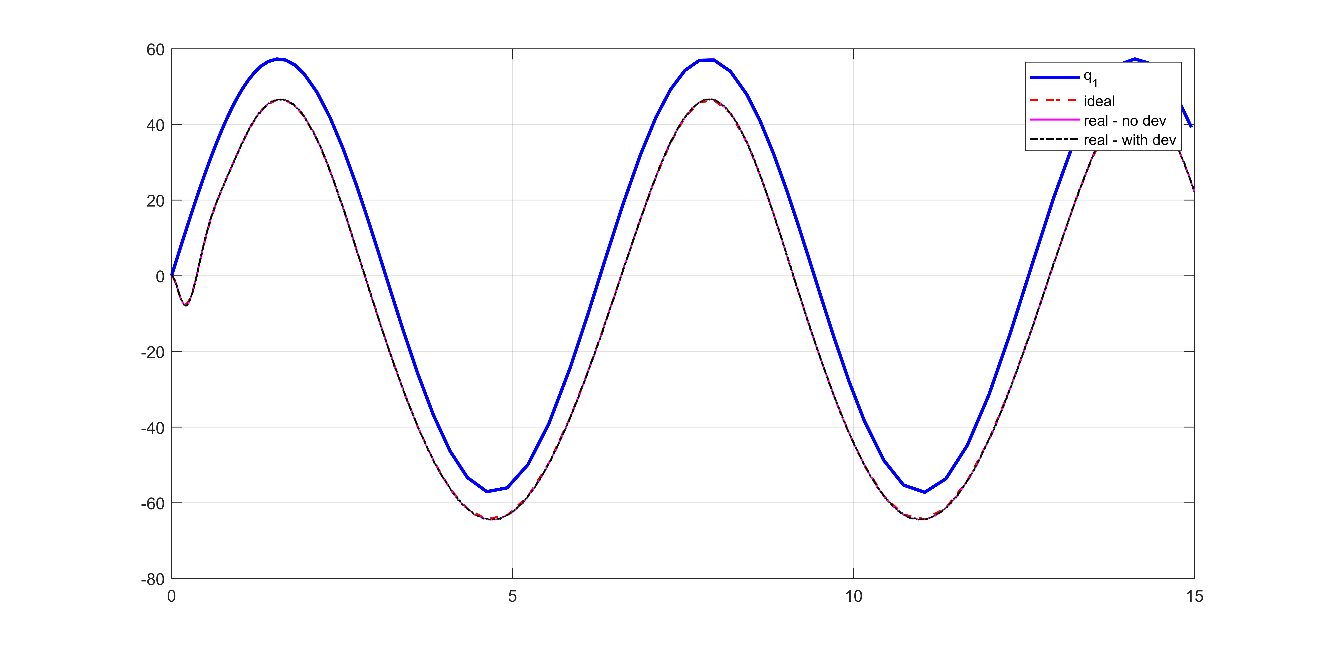


Figure 11: PD control step response - joint input torque

### Sine wave response



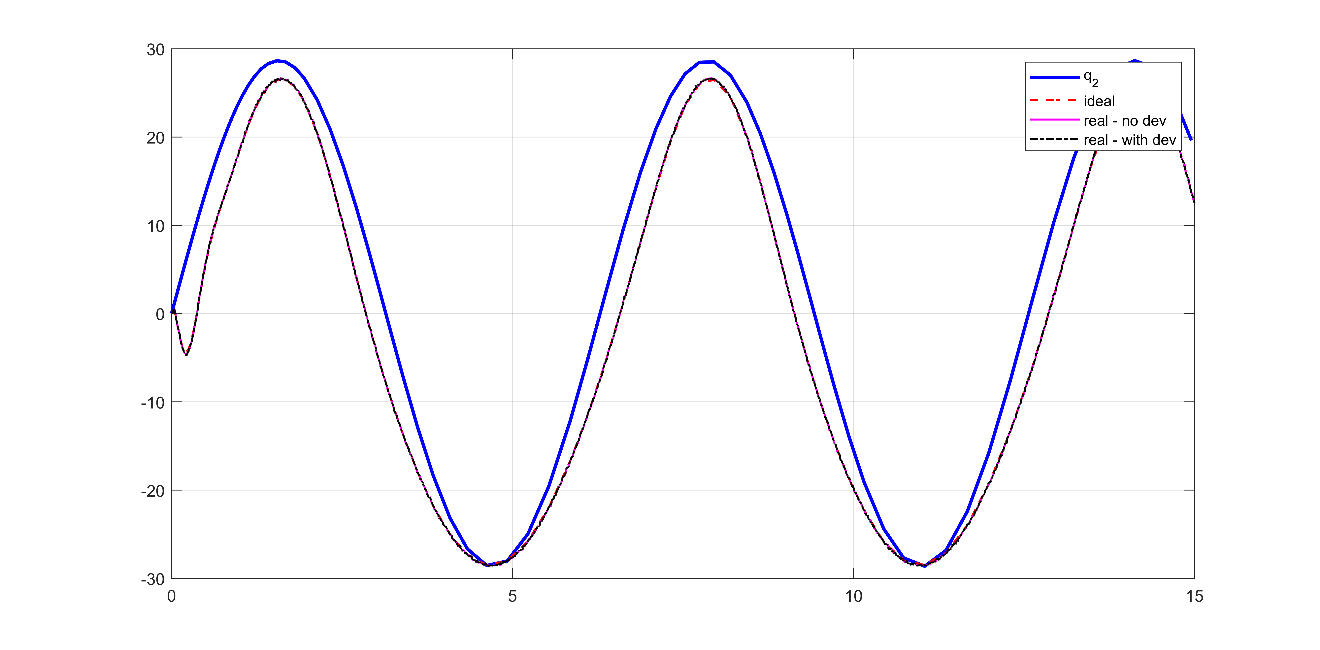
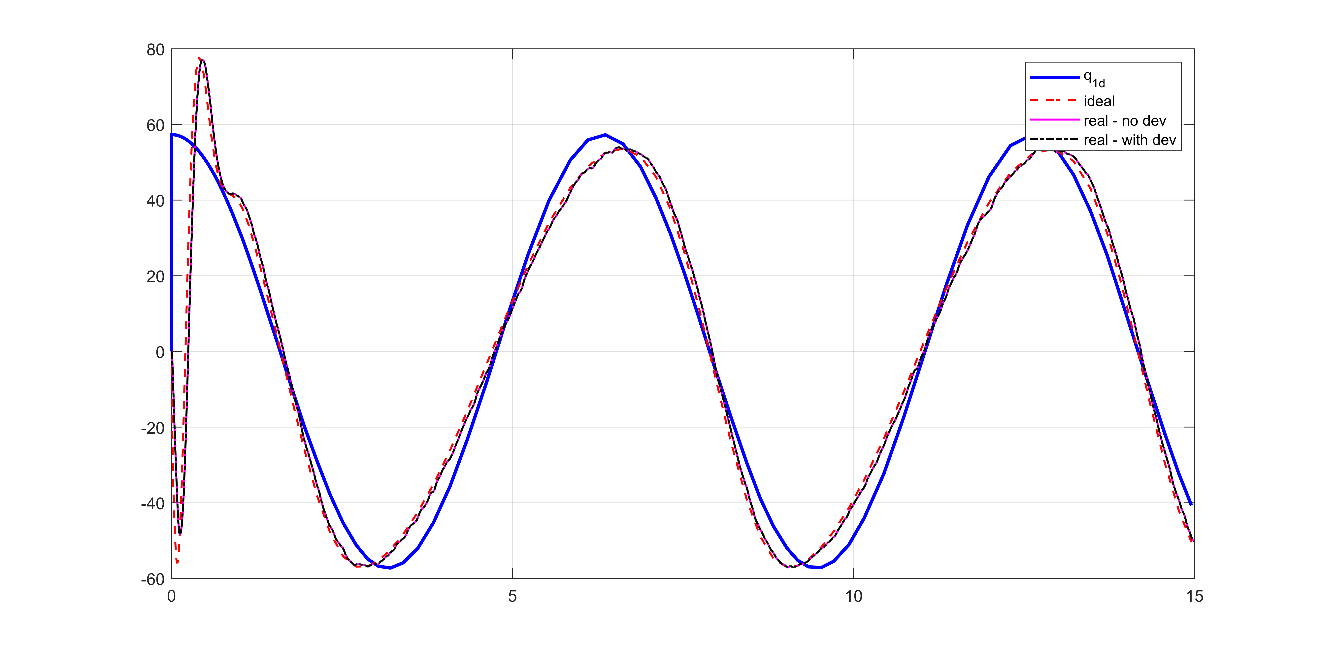


Figure 12: PD control sine wave response - joint position



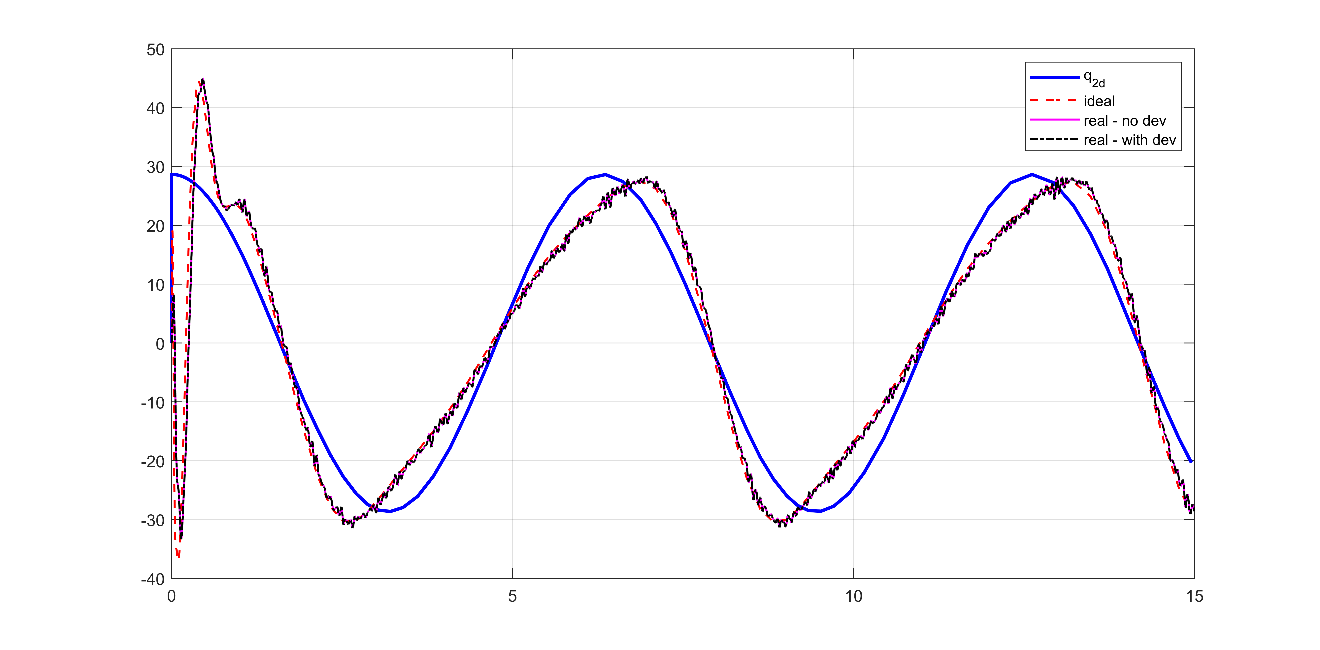


Figure 13: PD control sine wave response - joint velocity

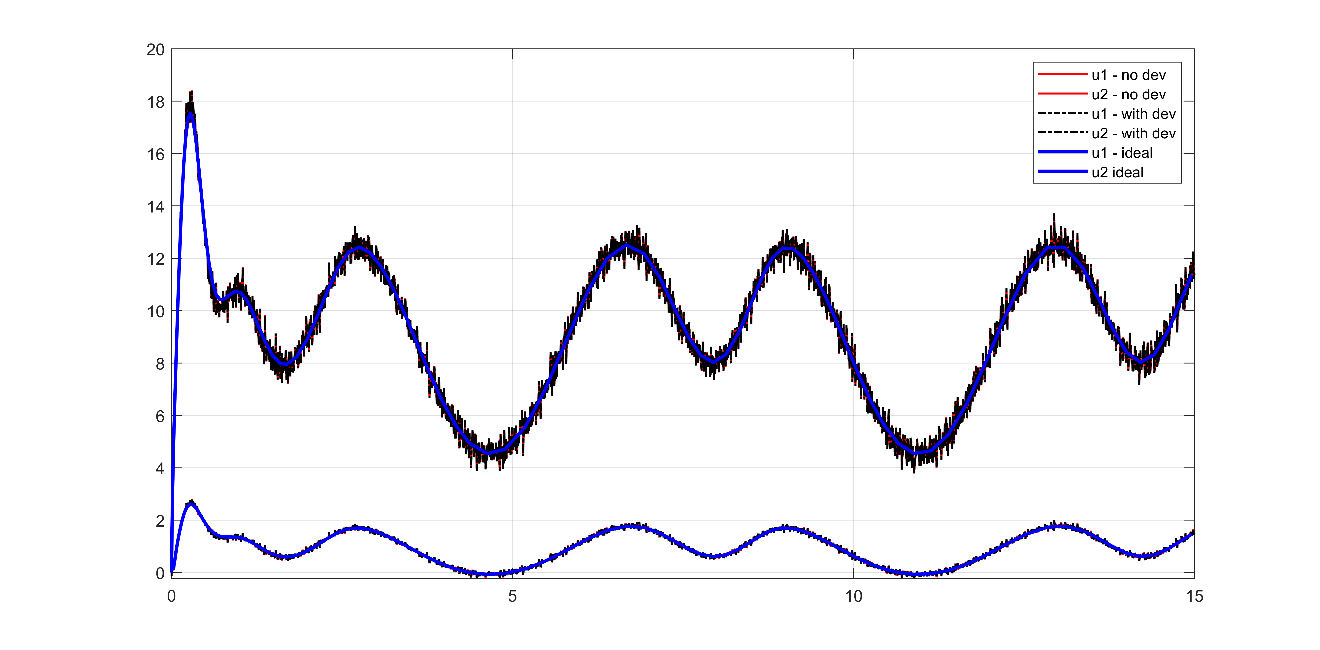


Figure 14: PD control sine wave response - joint input torque

### Observations

* Nonzero steady-state and tracking errors as expected
* This controller will not be considered any further

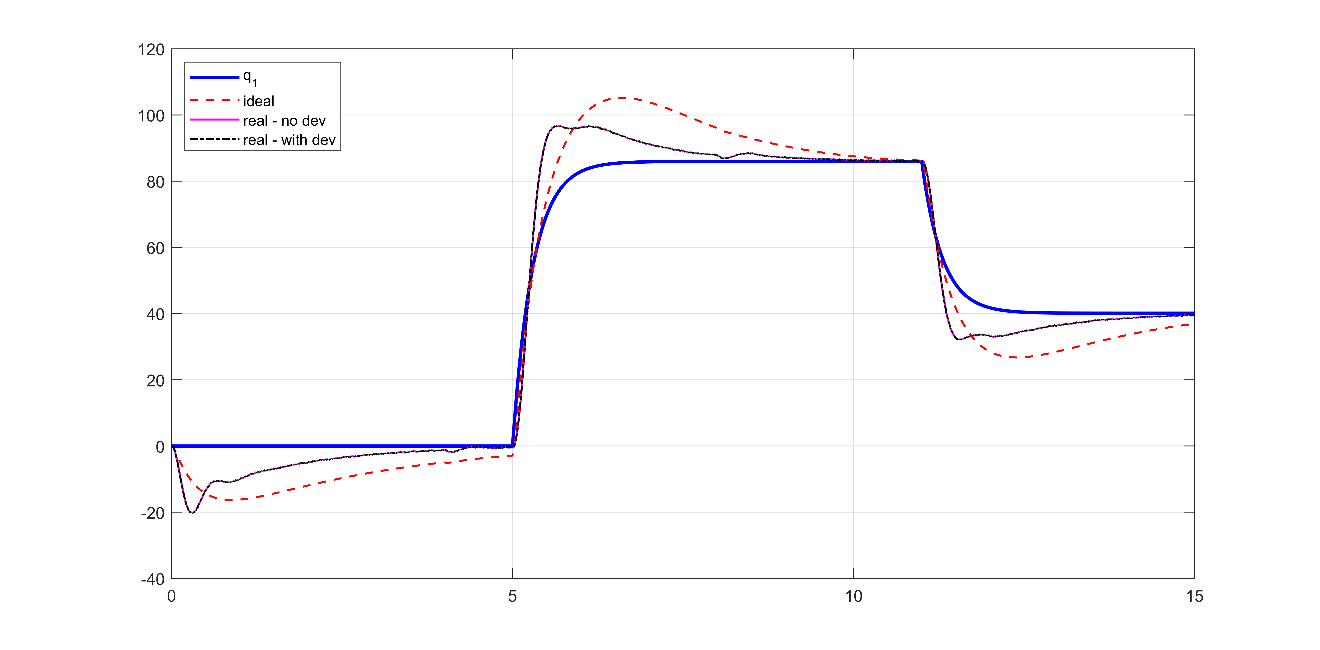
## PID Joint control

* This tuning may not be optimal. Gains were chosen such that the response is not aggressive on the actuators and the derivative gains are not large to avoid noise amplification.
* Tuning was done on the realistic case and used the same gains for the ideal case.

Table 4: PID gains

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Ideal case | | | | Realistic case | | | |
| Parameter | Value | Parameter | Value | Parameter | Value | Parameter | Value |
|  | 30 |  | 7 |  | 30 |  | 7 |
|  | 2.5 |  | 0.5 |  | 2.5 |  | 0.5 |
|  | 20 |  | 15 |  | 20 |  | 15 |

### Step response



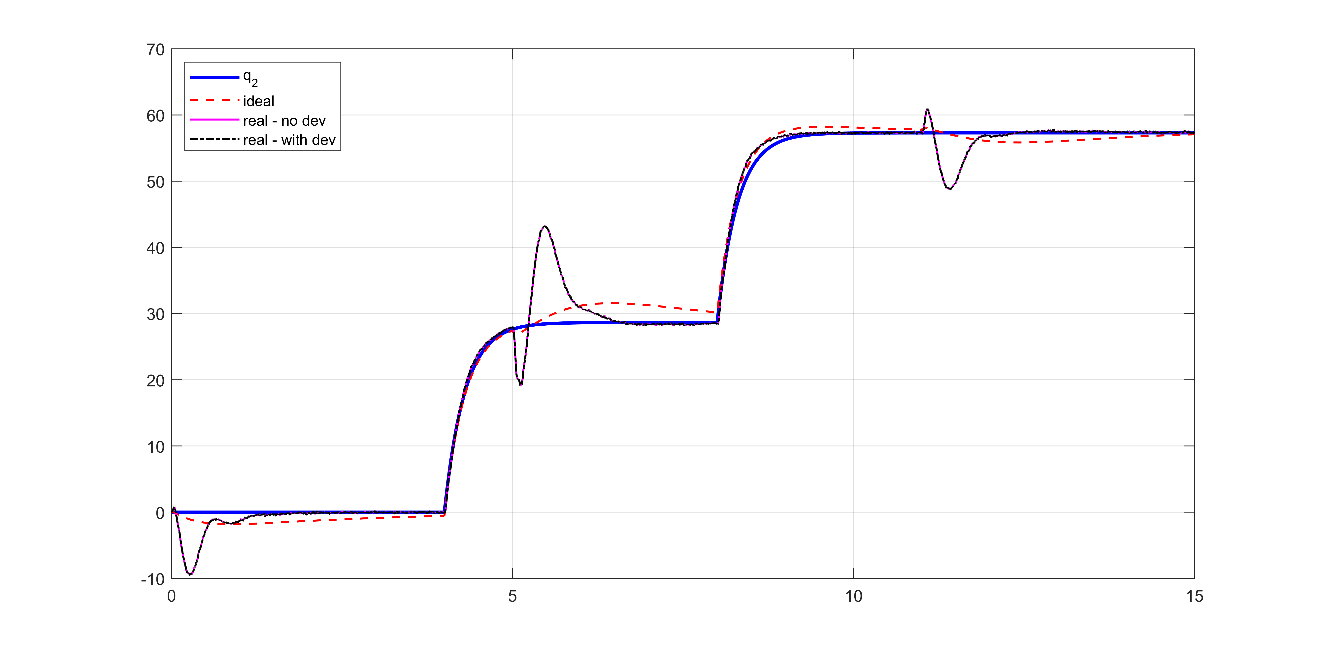
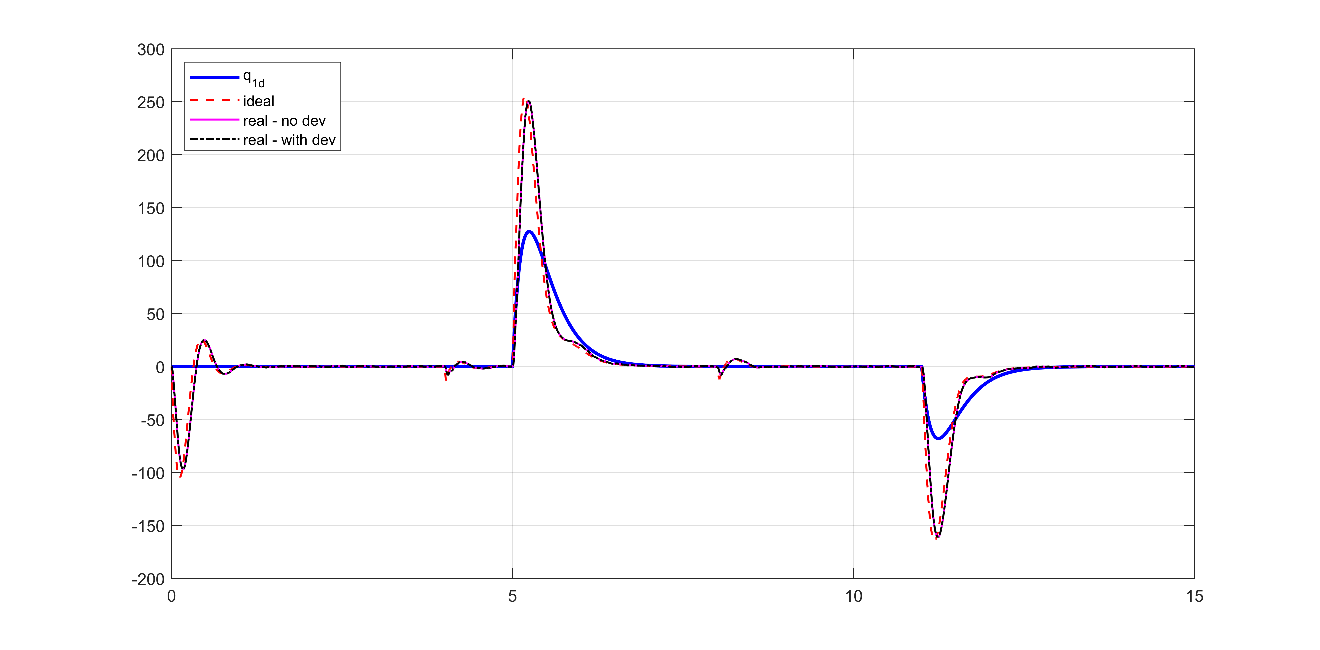


Figure 15: PID control step response - joint position



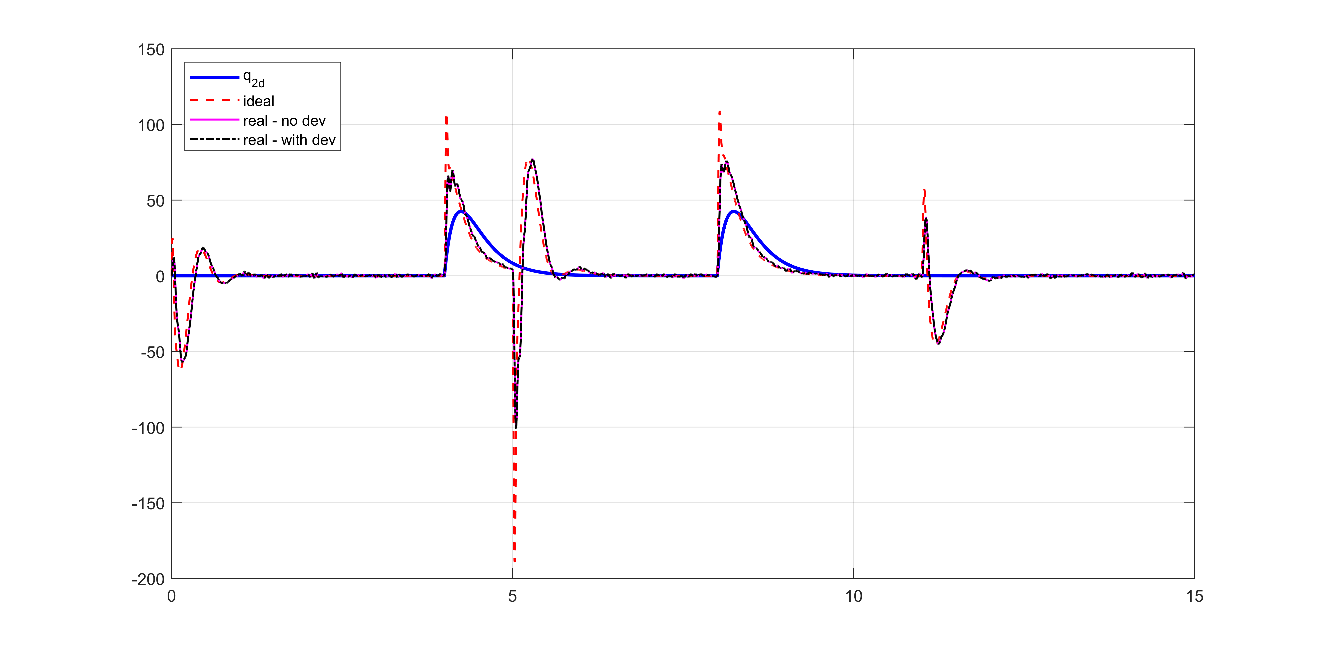


Figure 16: PID control step response - joint velocity

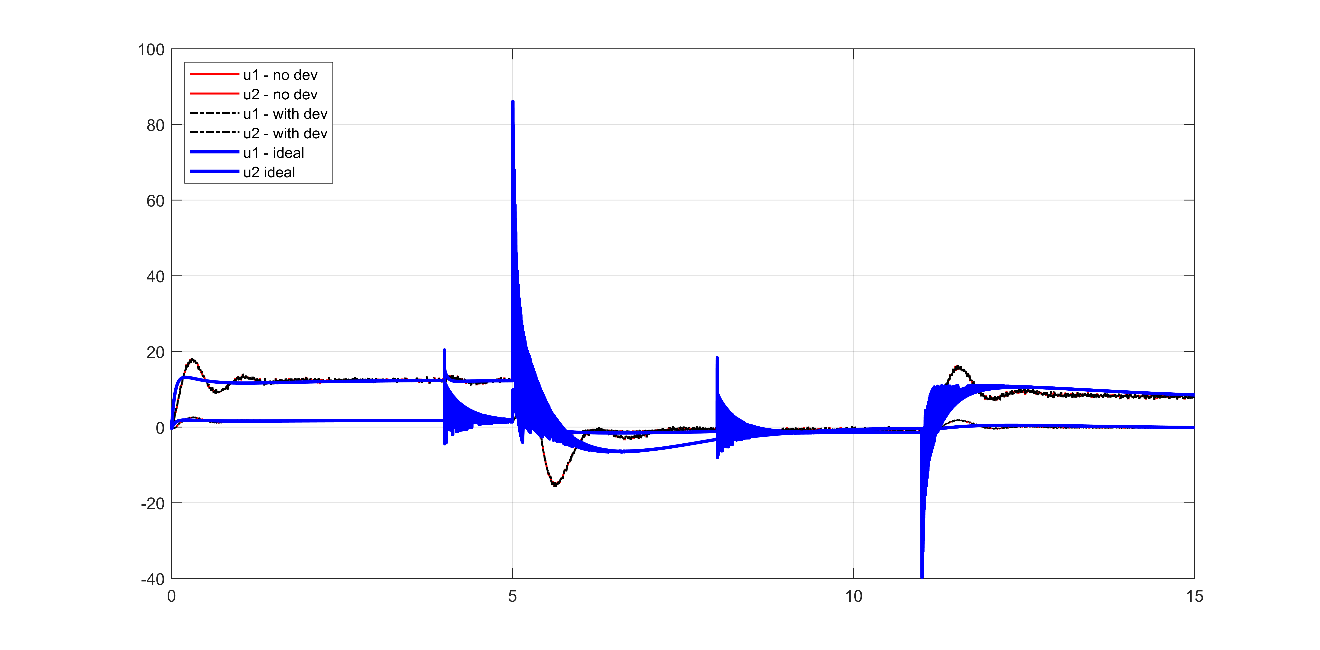
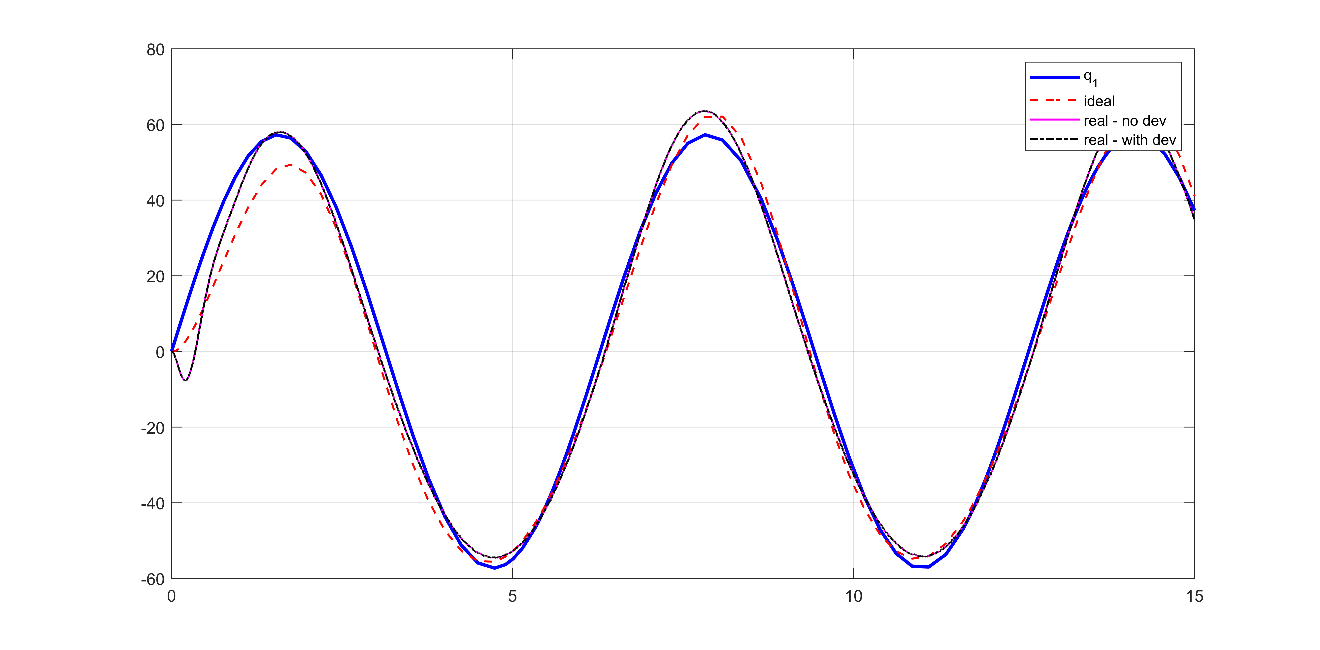


Figure 17: PID control step response - joint input torque

### Sine wave response



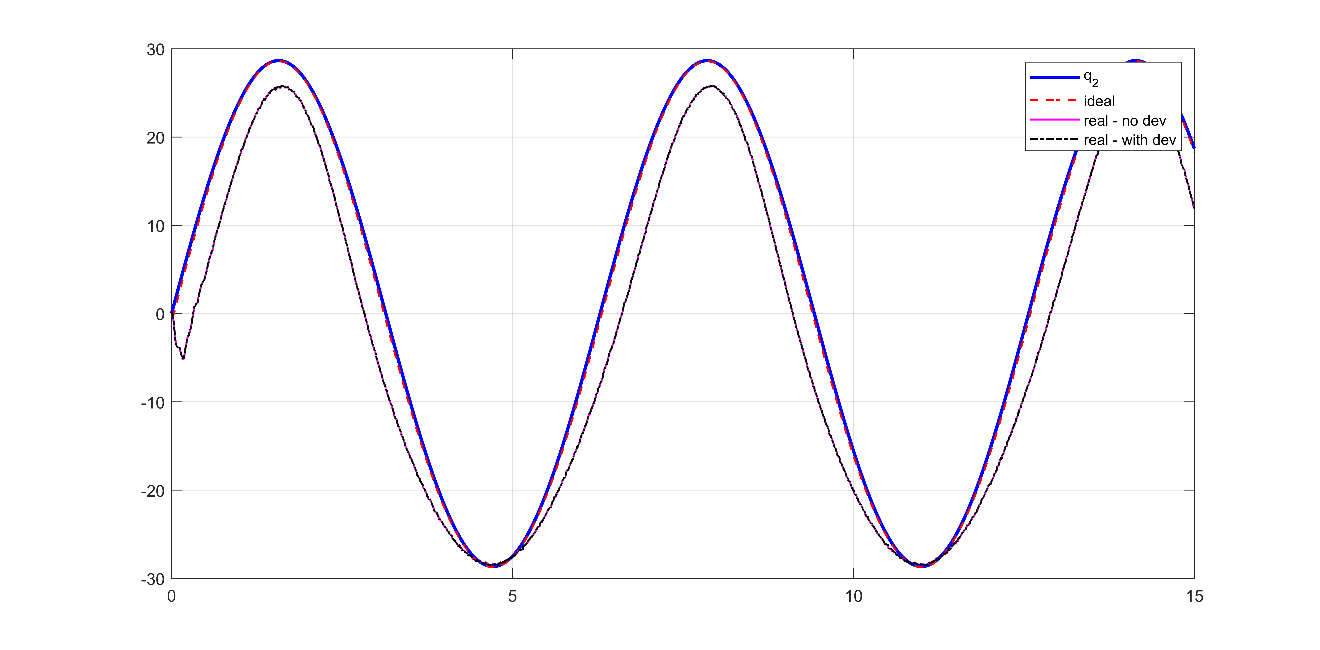
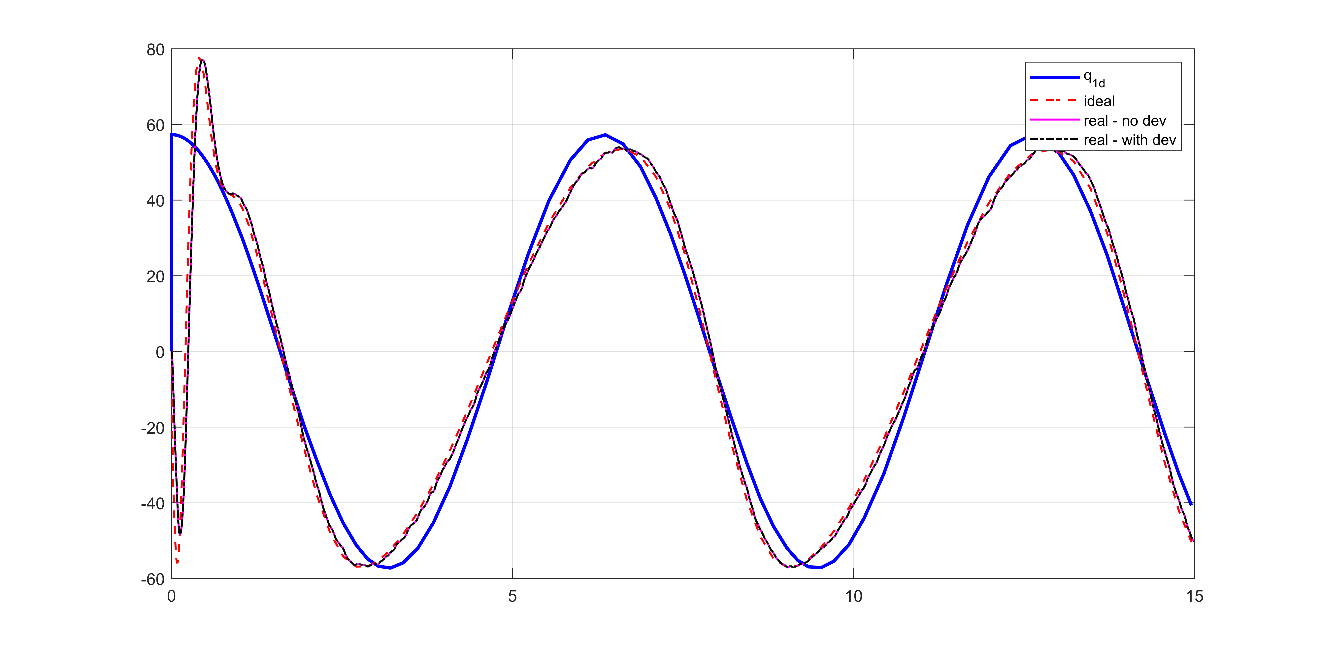


Figure 18: PID control sine wave response - joint position



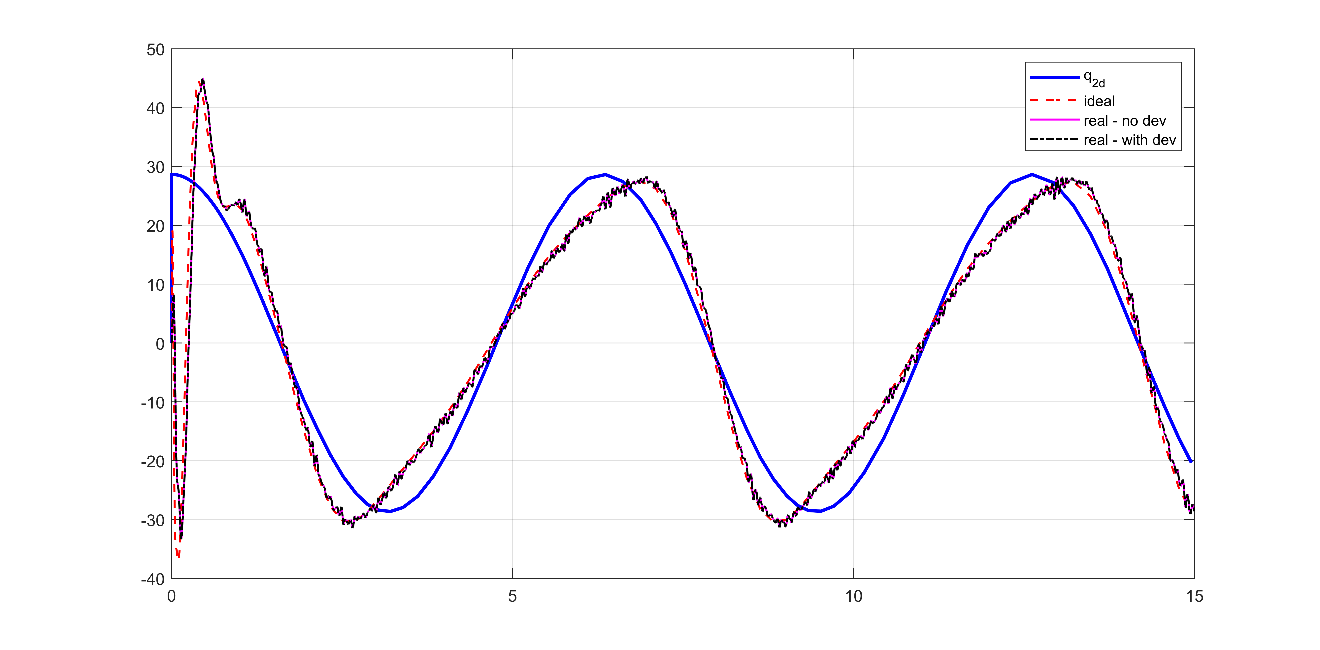


Figure 19: PID control sine wave response - joint velocity

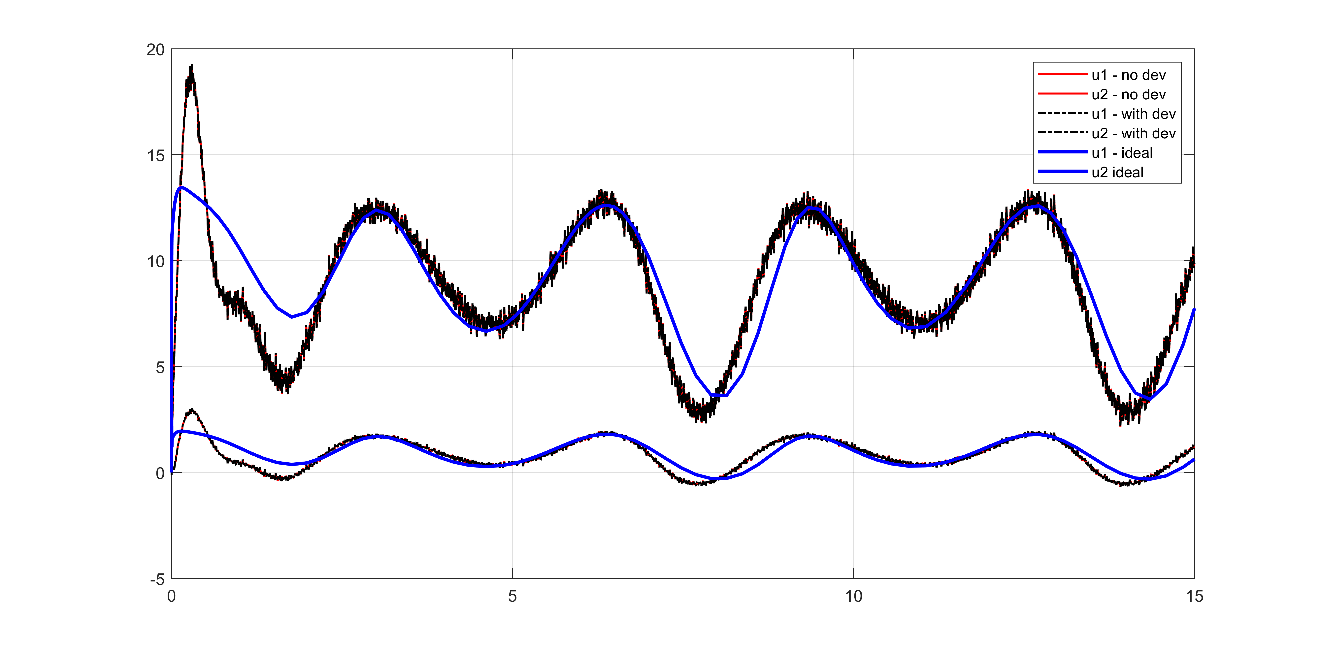


Figure 20: PID control sine wave response - joint input torque

### Observations

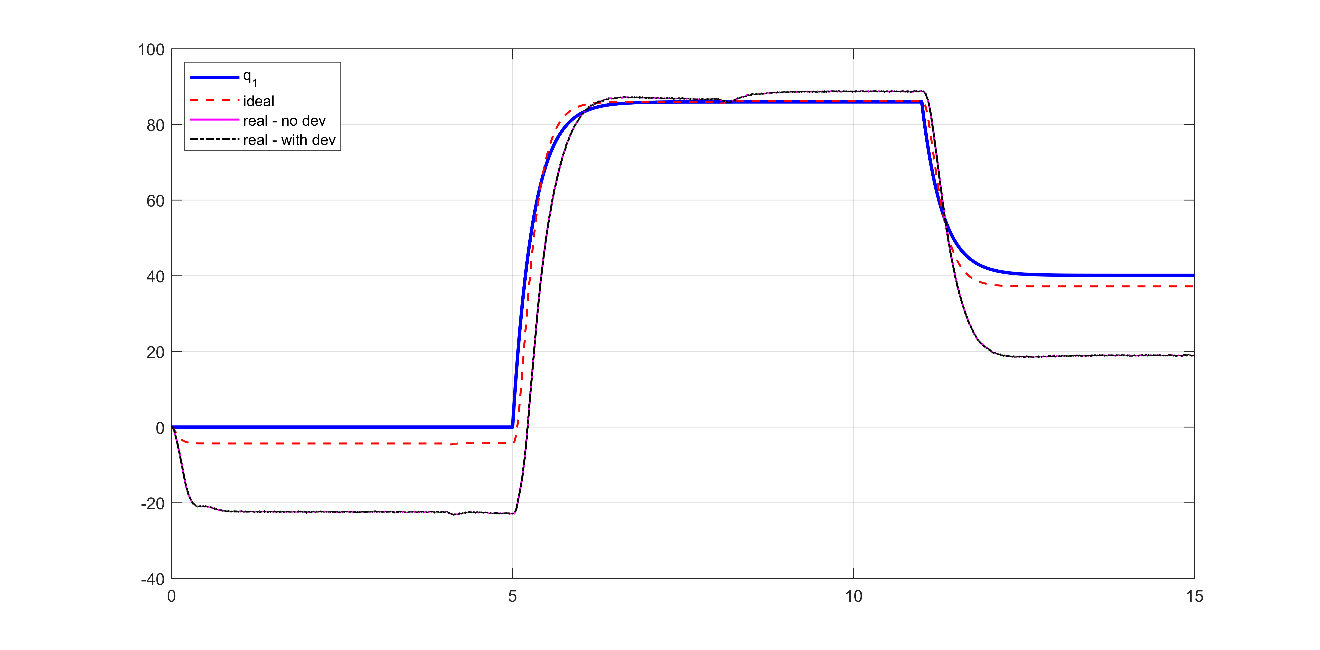
* Steady-state goes to zero when given enough time
* Tracking error is still not zero due to response lag
* High-magnitude and high frequency commanded torques at the point of step change

## Pole Placement Control

Table 5: Pole values and feedback gain- ideal

|  |  |  |  |
| --- | --- | --- | --- |
| Ideal case | | Realistic case | |
| **Parameter** | Value | **Parameter** | Value |
| **poles** | [-90, -80, -70, -10.8] | **poles** | [-25, -24, -23, -5.8] |
|  |  |  |  |

### Step response



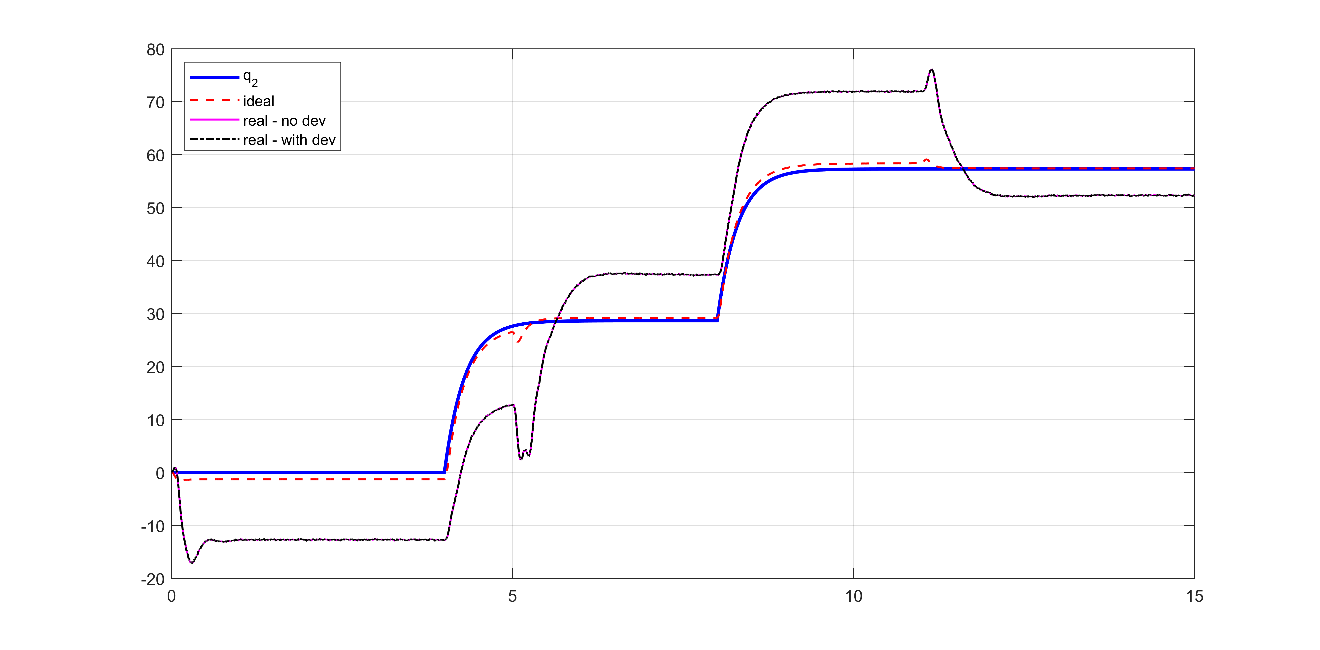
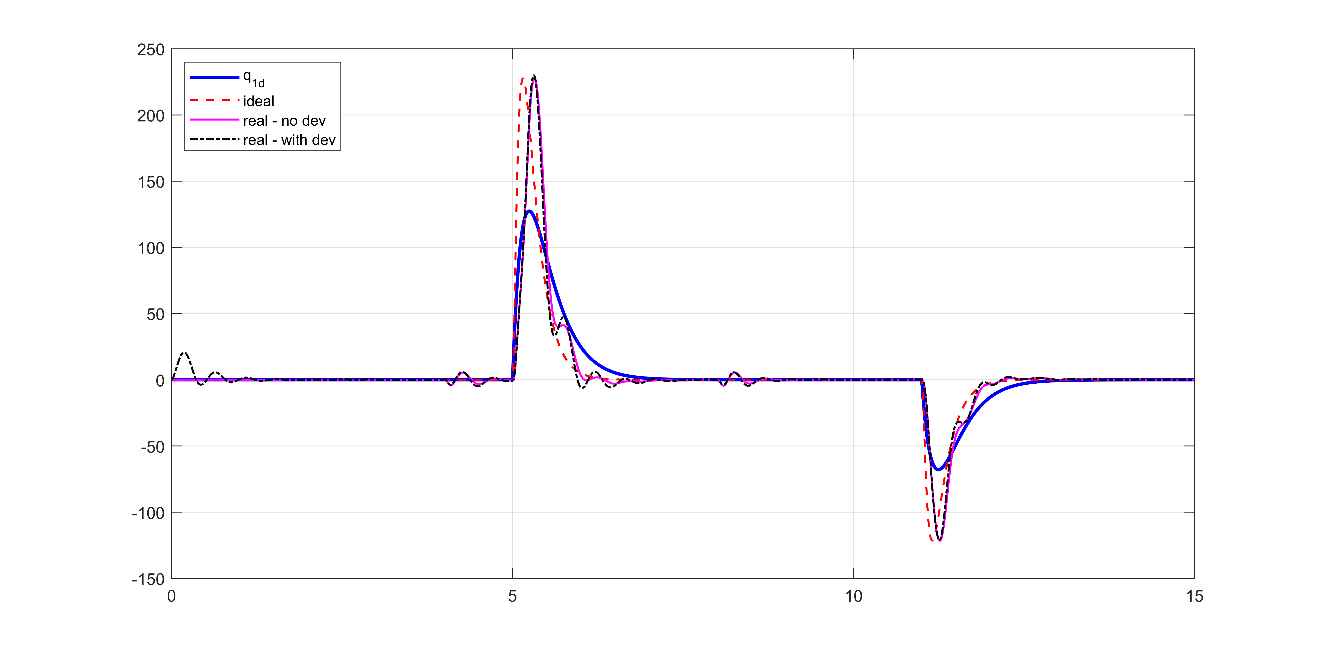


Figure 21: Pole placement control step response - joint position



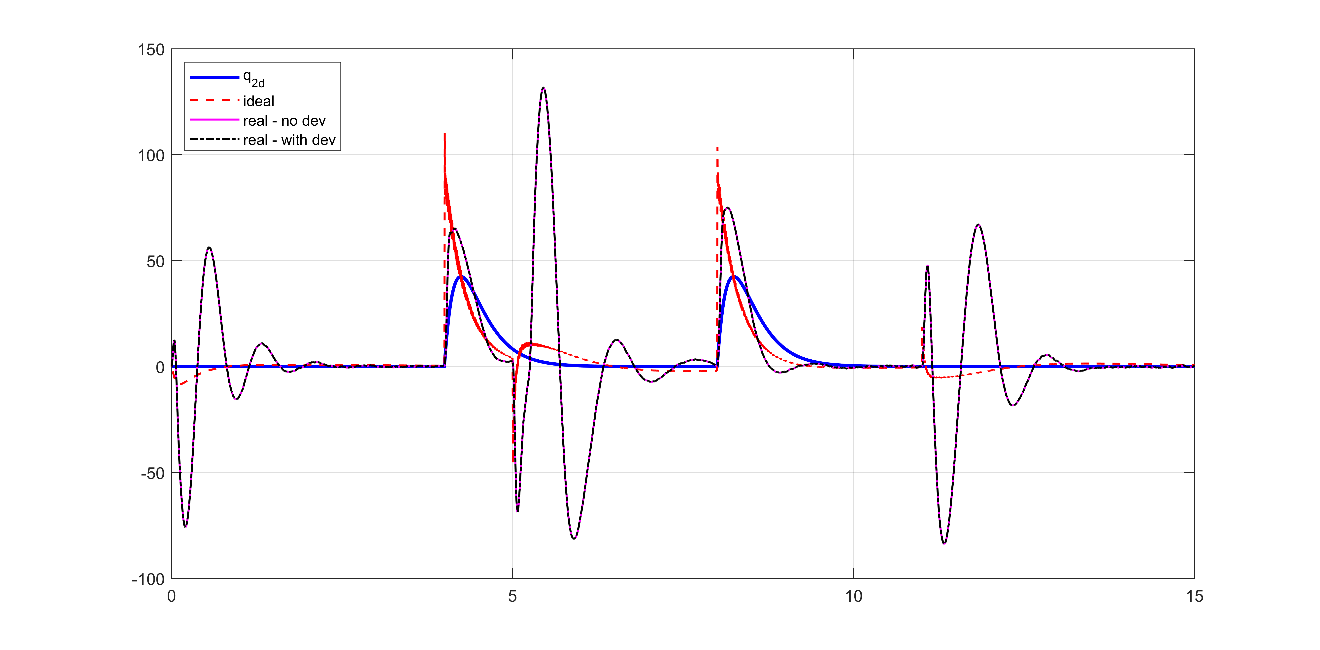


Figure 22: Pole placement control step response - joint velocity

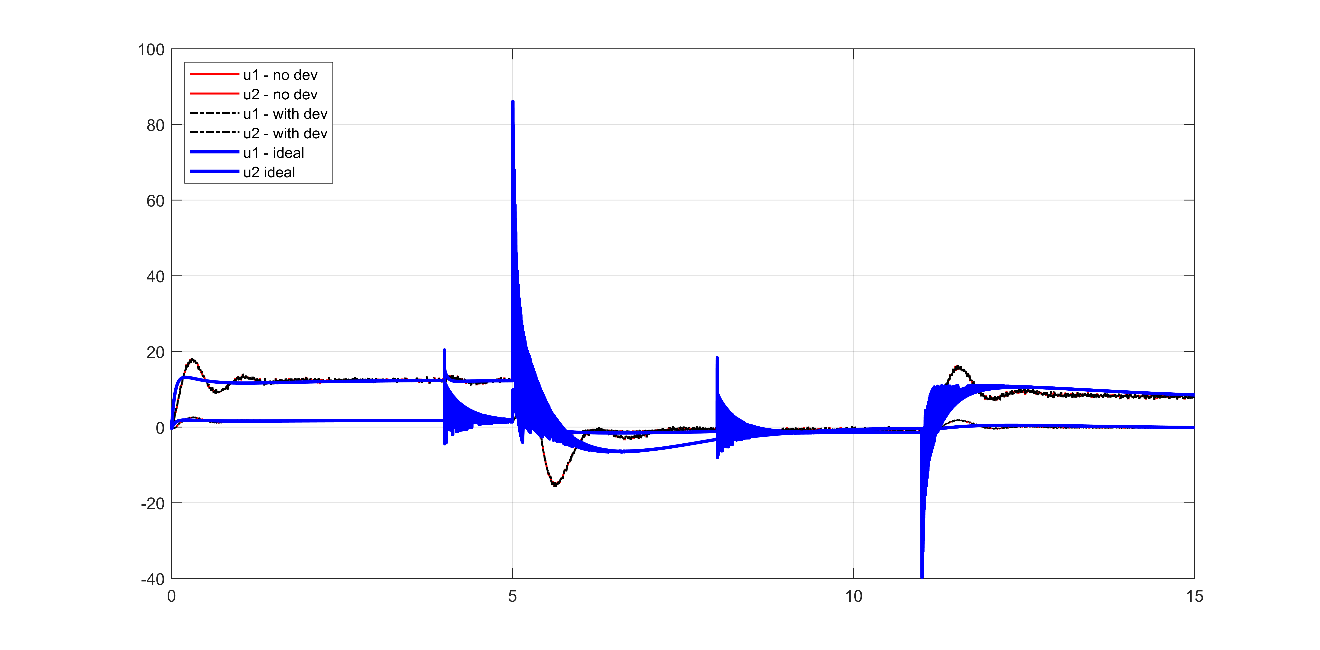
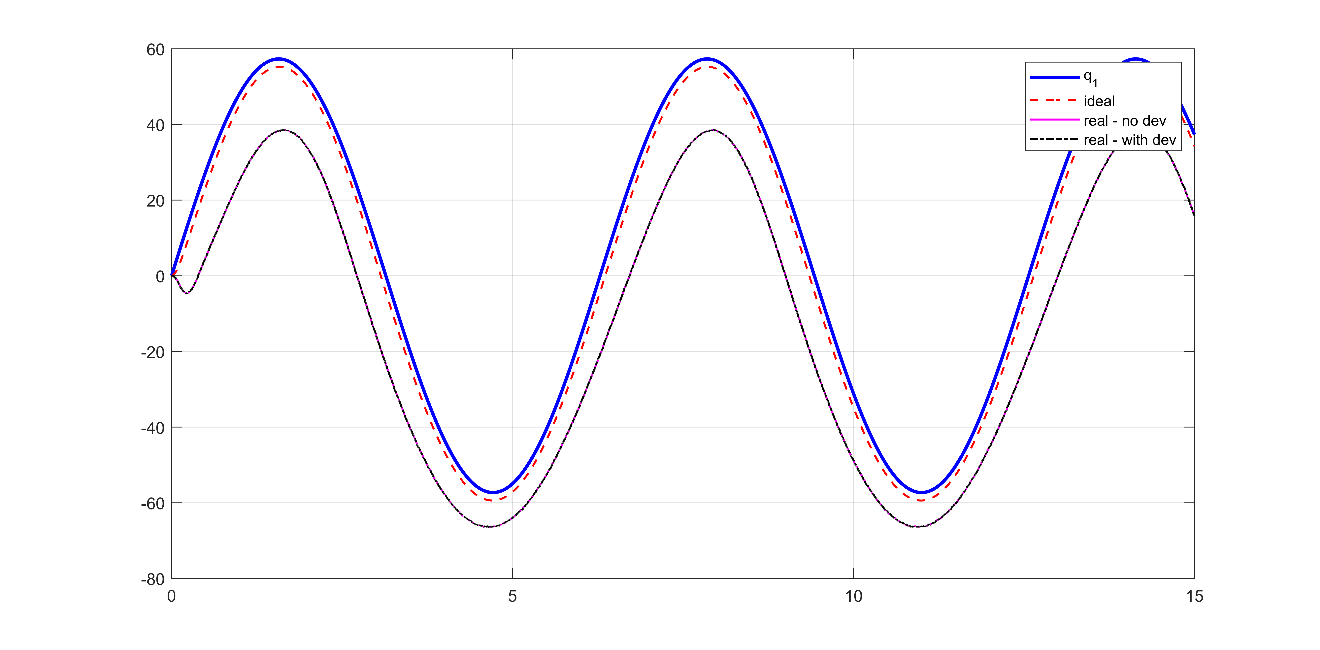


Figure 23: Pole placement control step response - joint input torque

### Sine wave response



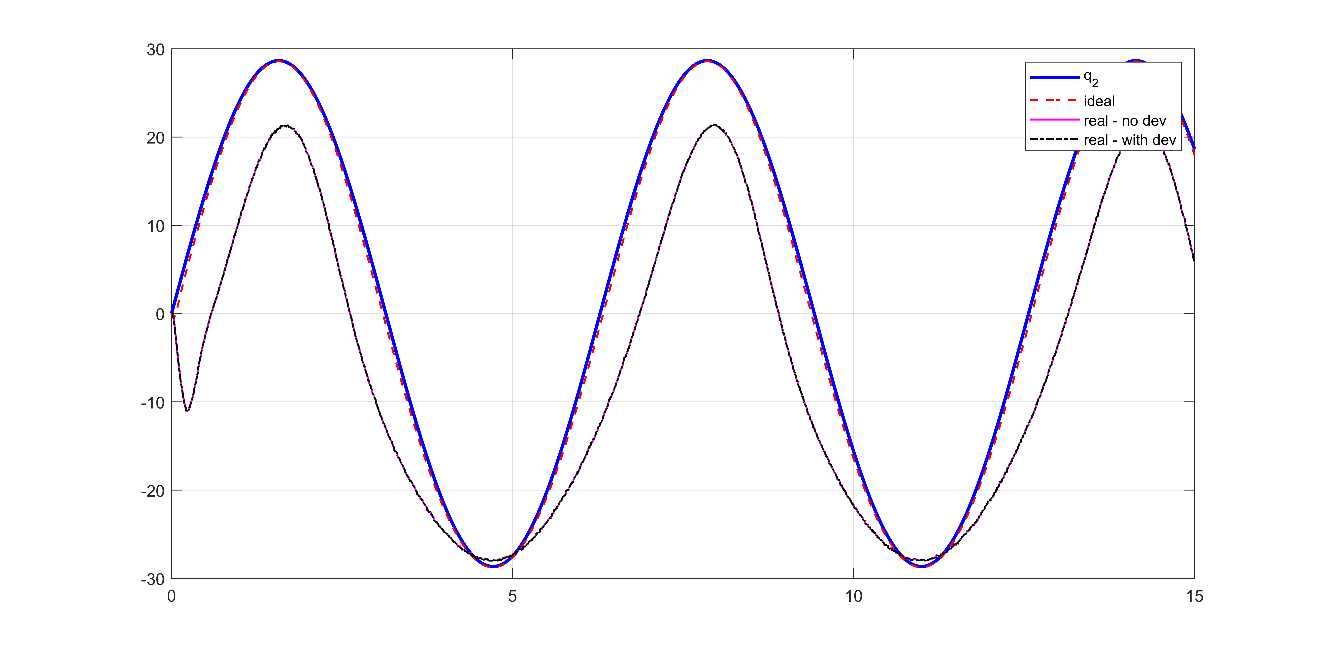
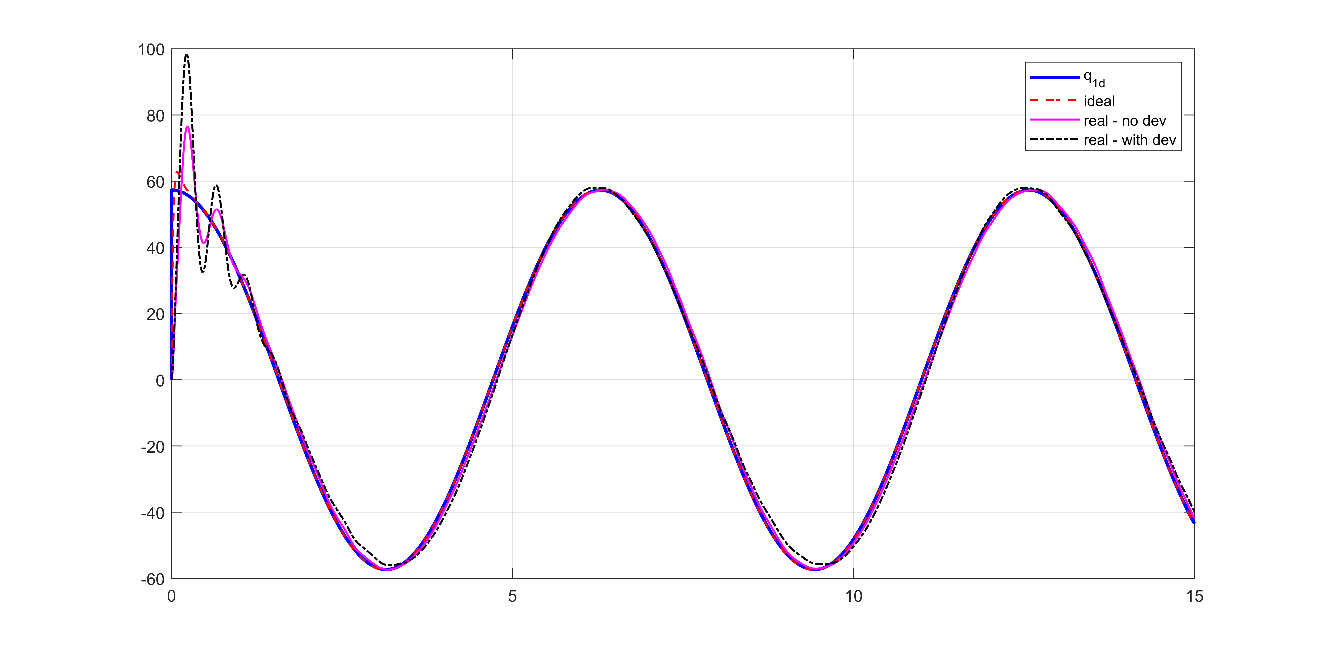


Figure 24: Pole placement control sine wave response - joint position



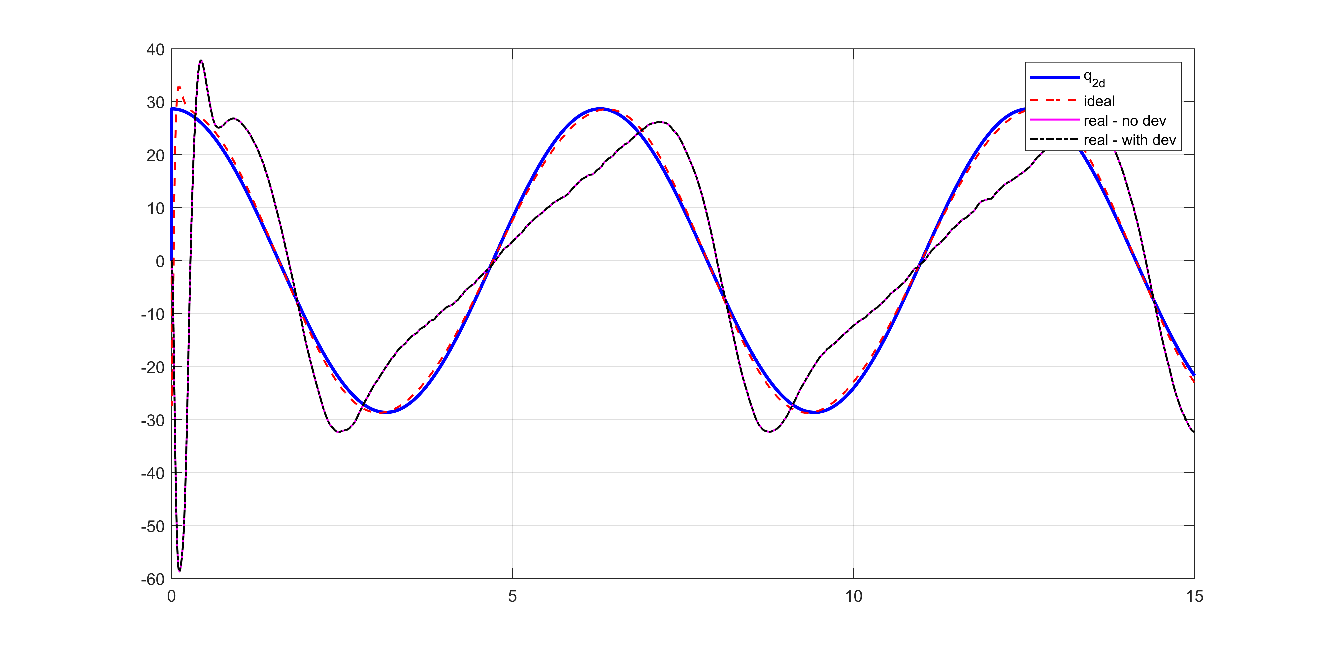


Figure 25: Pole placement control sine wave response - joint velocity

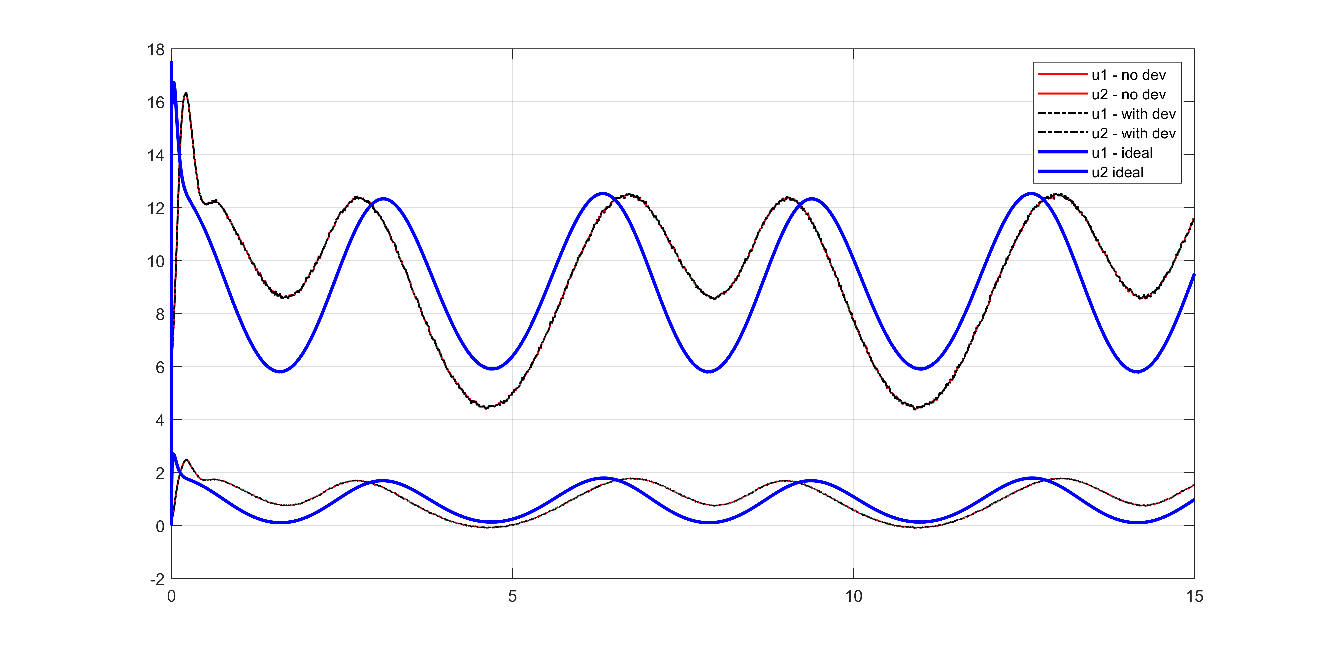


Figure 26: Pole placement control sine wave response - joint input torque

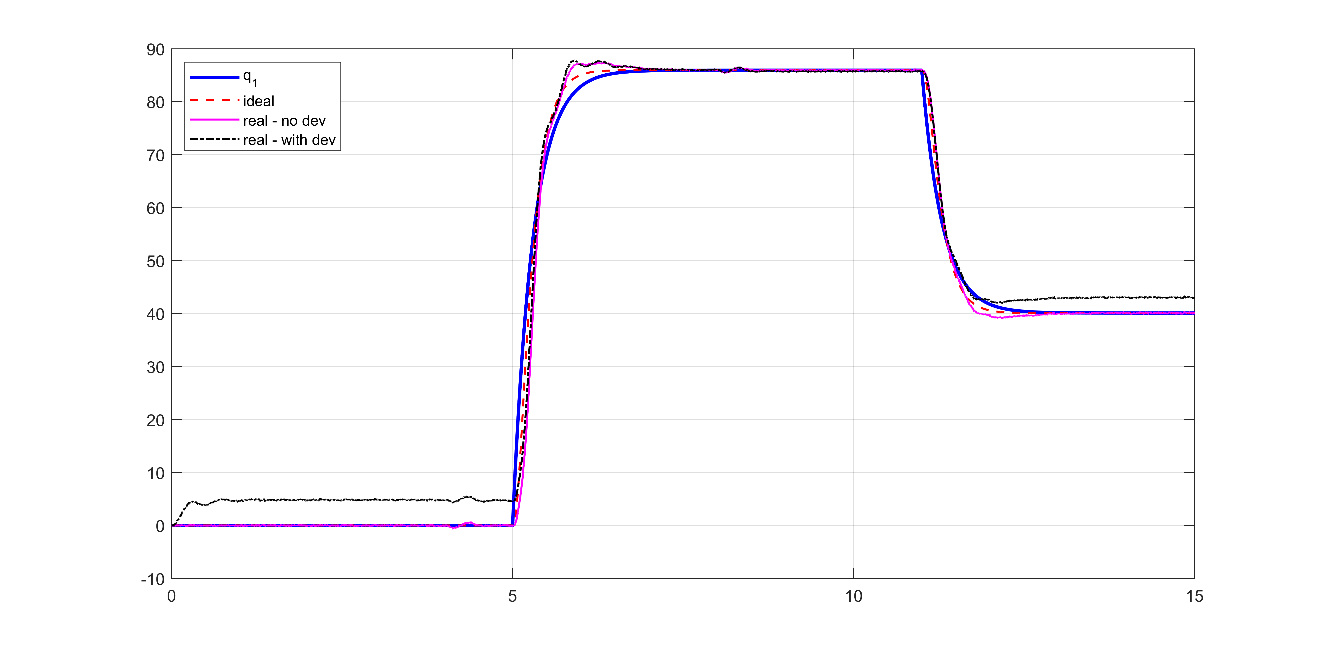
### Observations

* Nonzero Steady-state and tracking error
* This controller will not be considered for further investigation due to poor performance

## Pole Placement Control with feedforward disturbance terms

* The same gains as the Pole Placement controller

### Step response



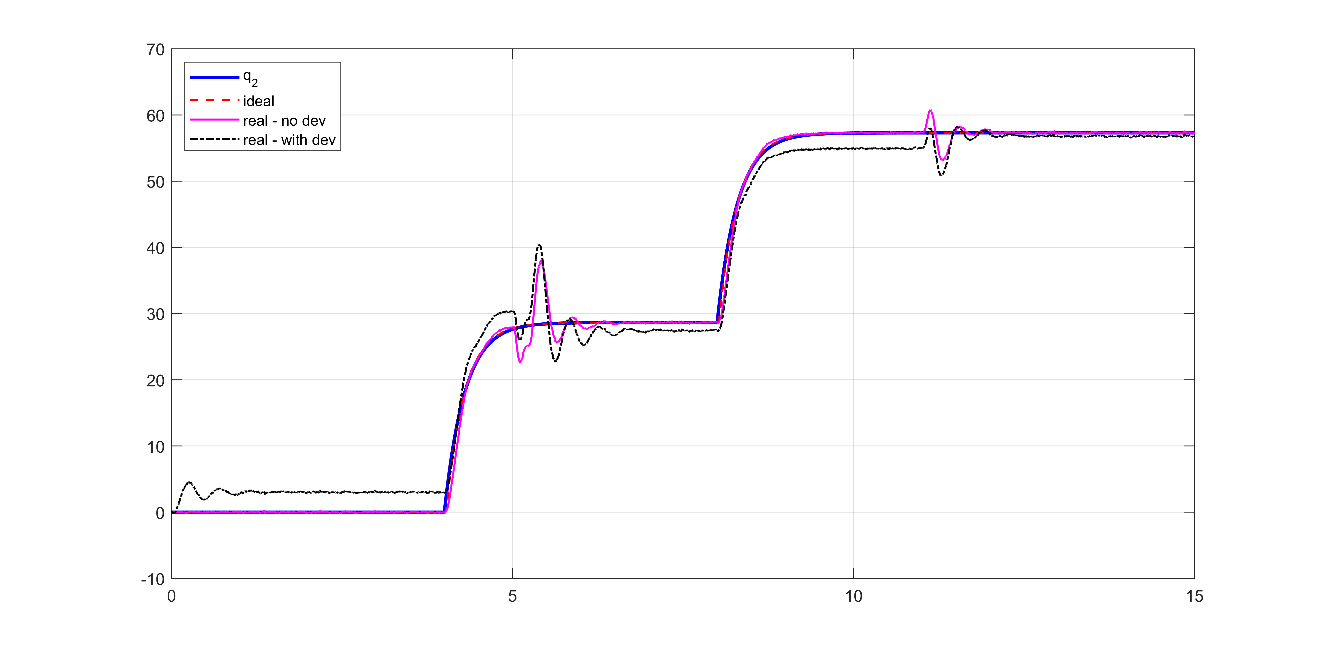
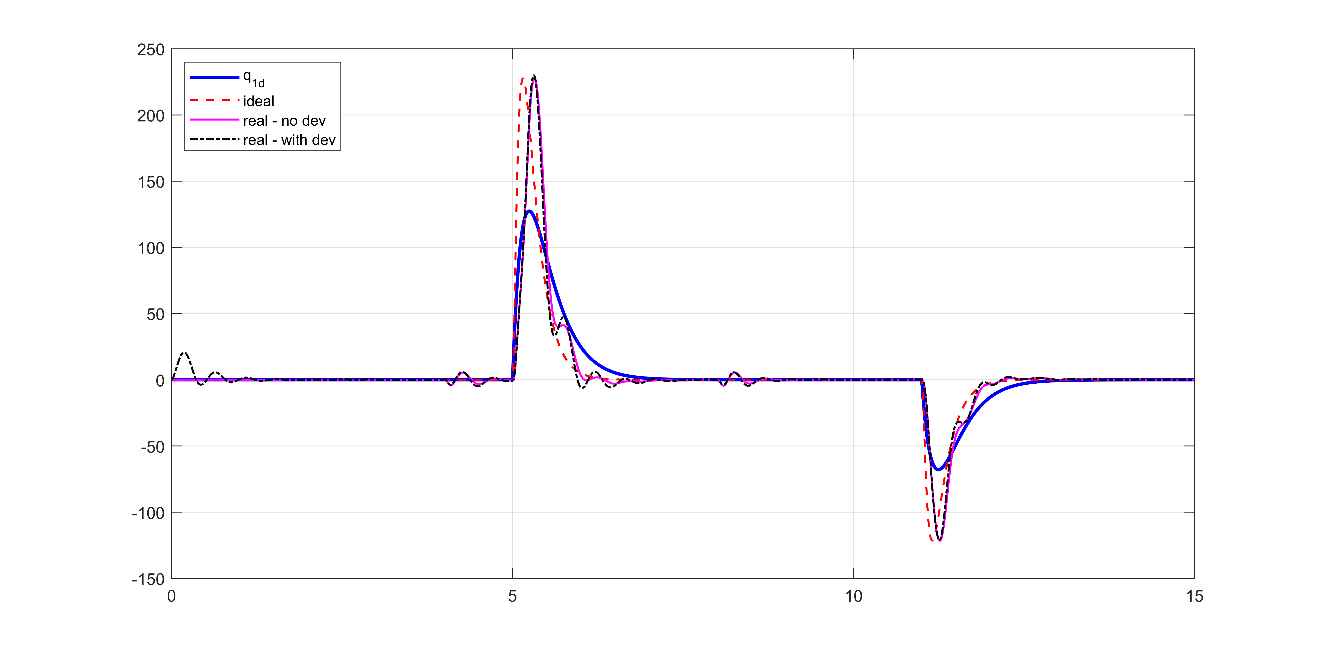


Figure 27: Pole placement+D control step response - joint position



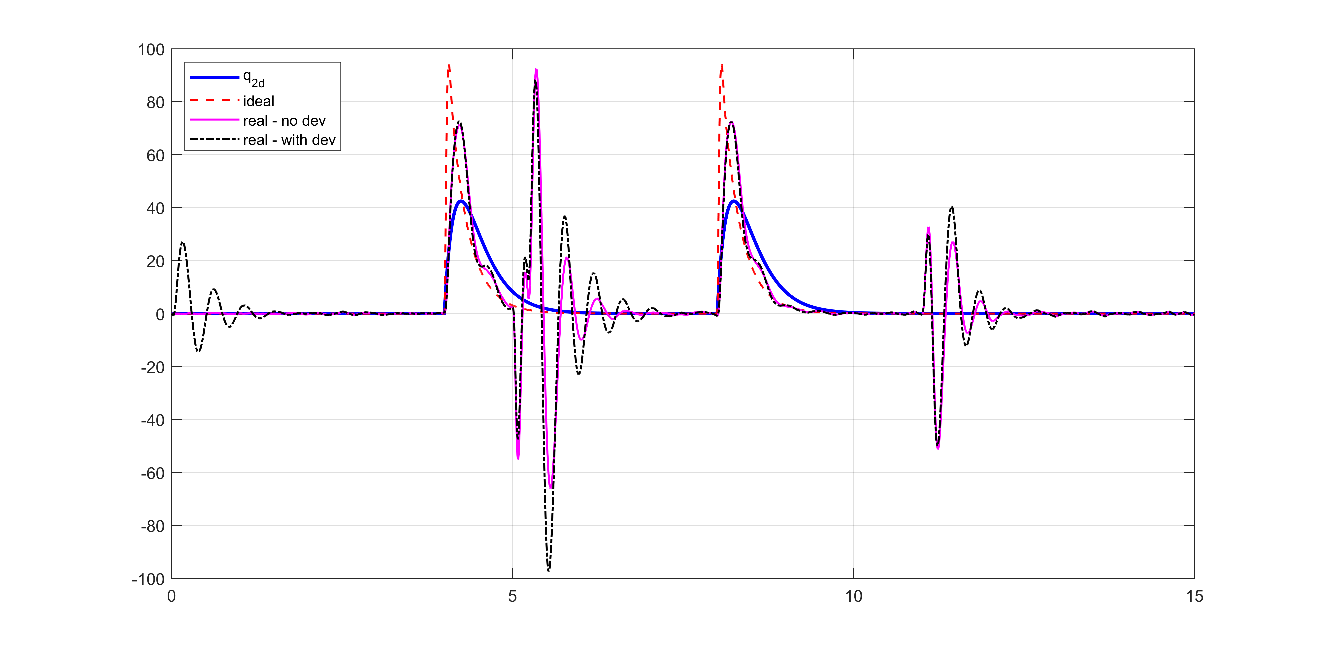


Figure 28: Pole placement+D control step response - joint velocity

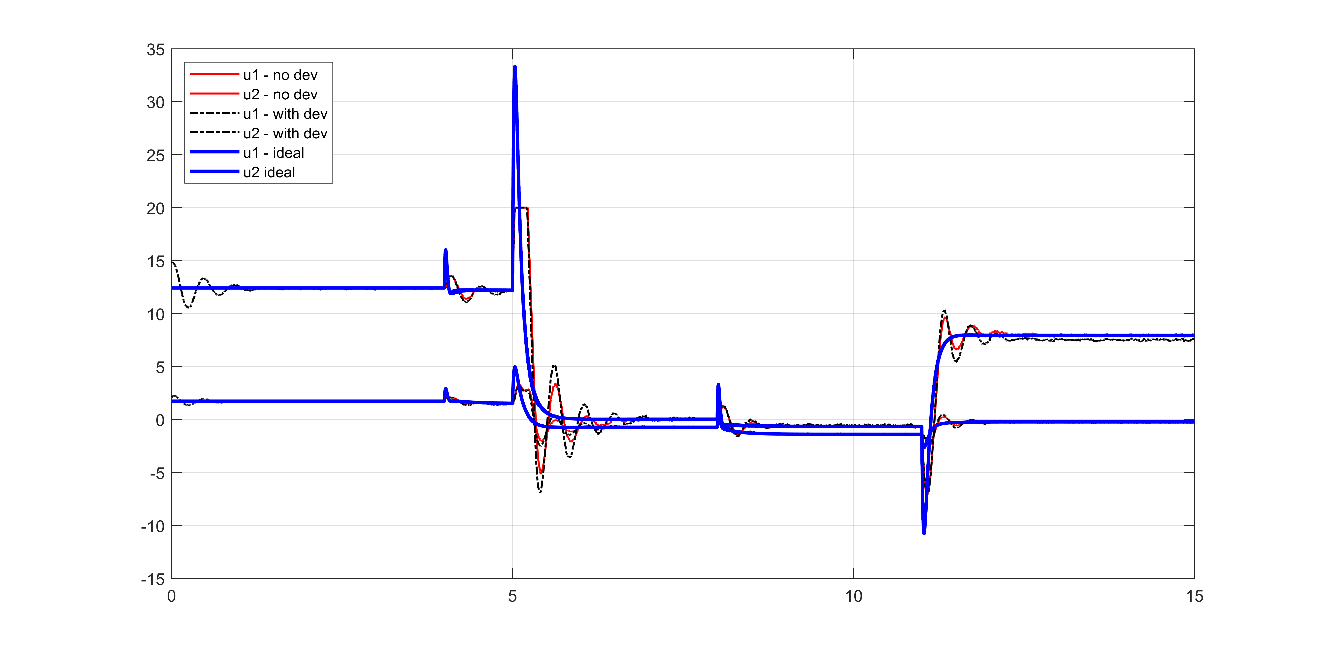
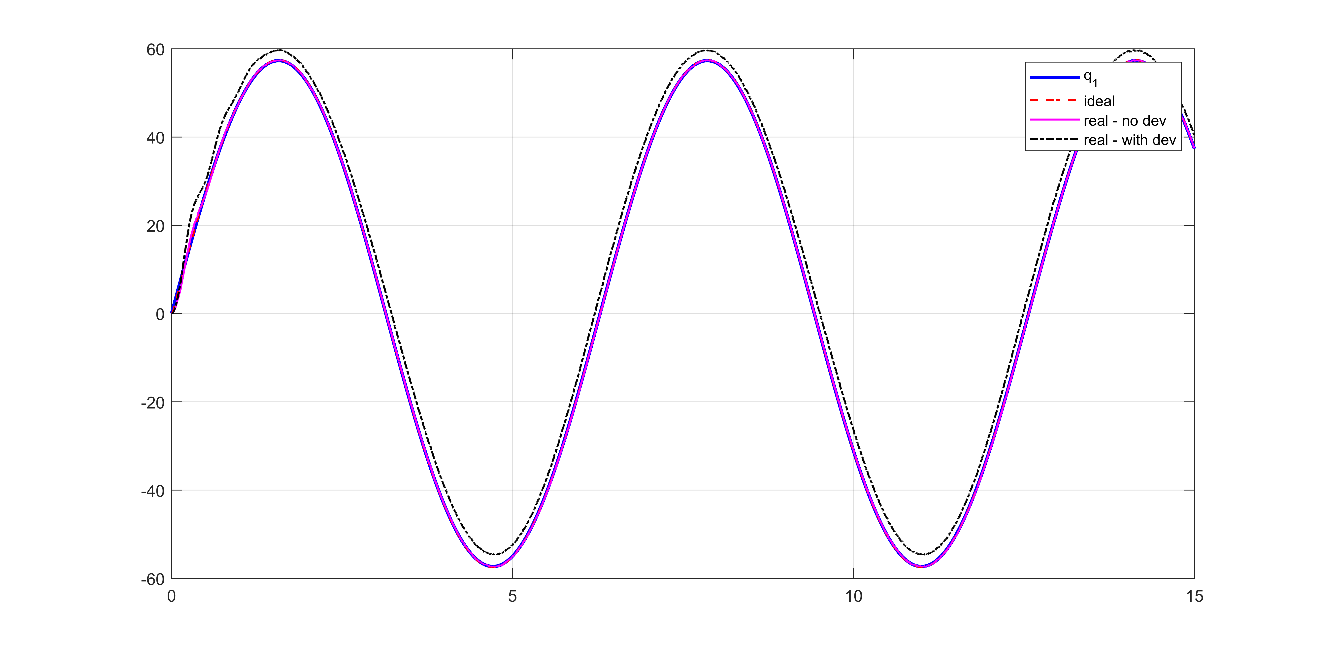


Figure 29: Pole placement+D control step response - joint input torque

### Sine wave response



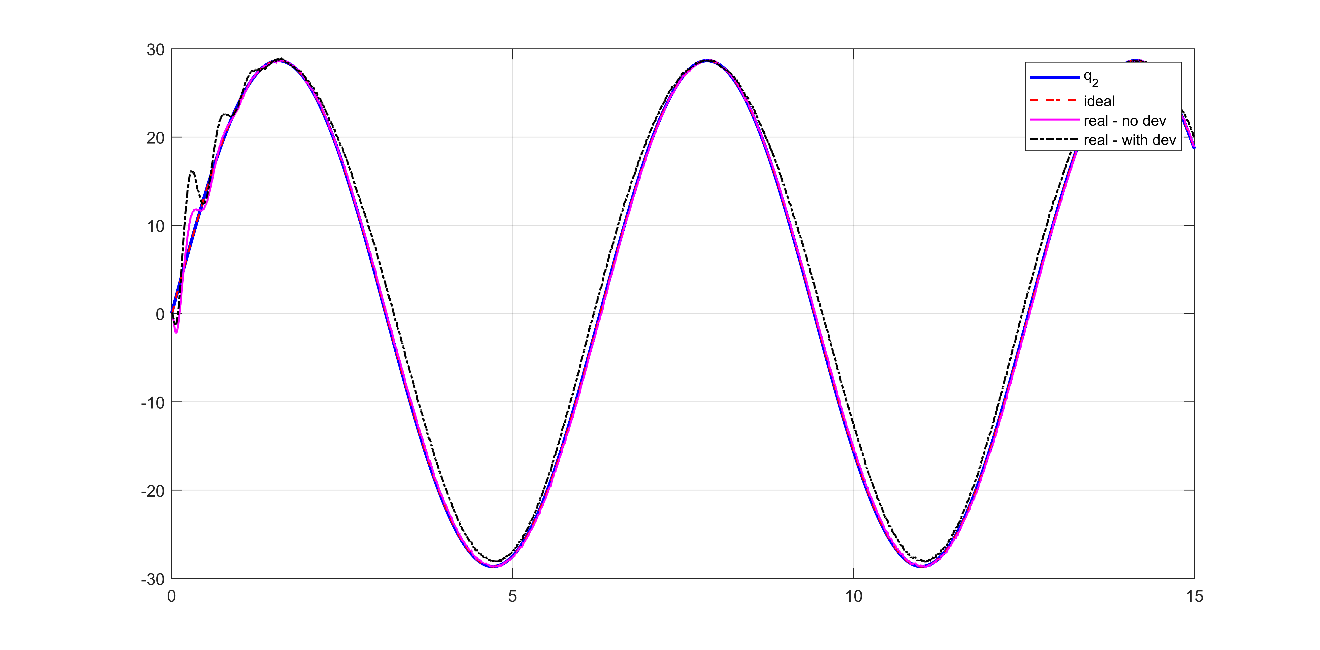
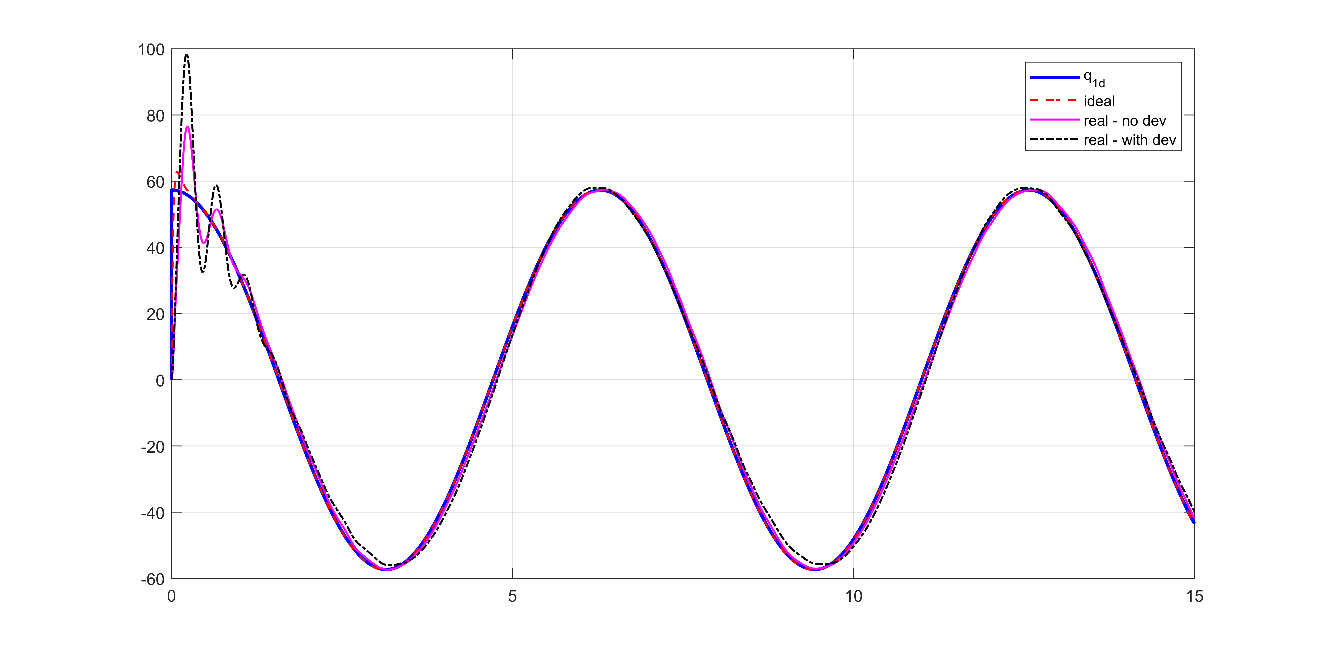


Figure 30: Pole placement+D control sine wave response - joint position



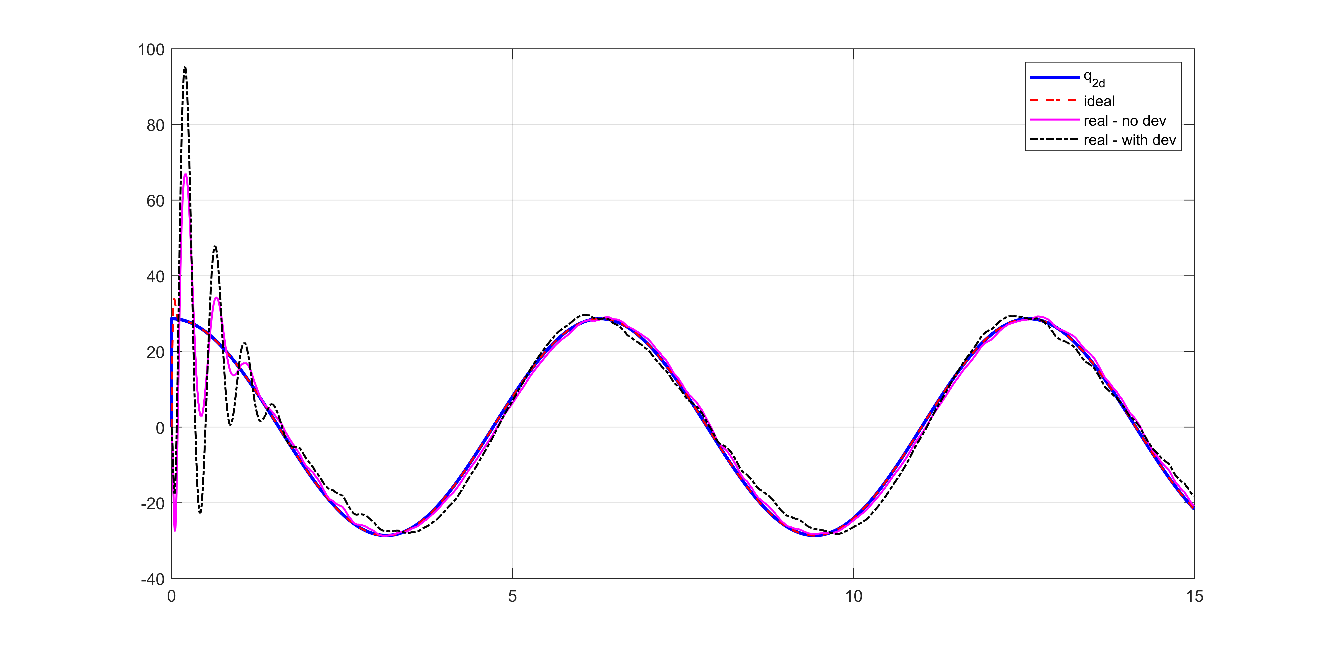


Figure 31: Pole placement+D control sine wave response - joint velocity

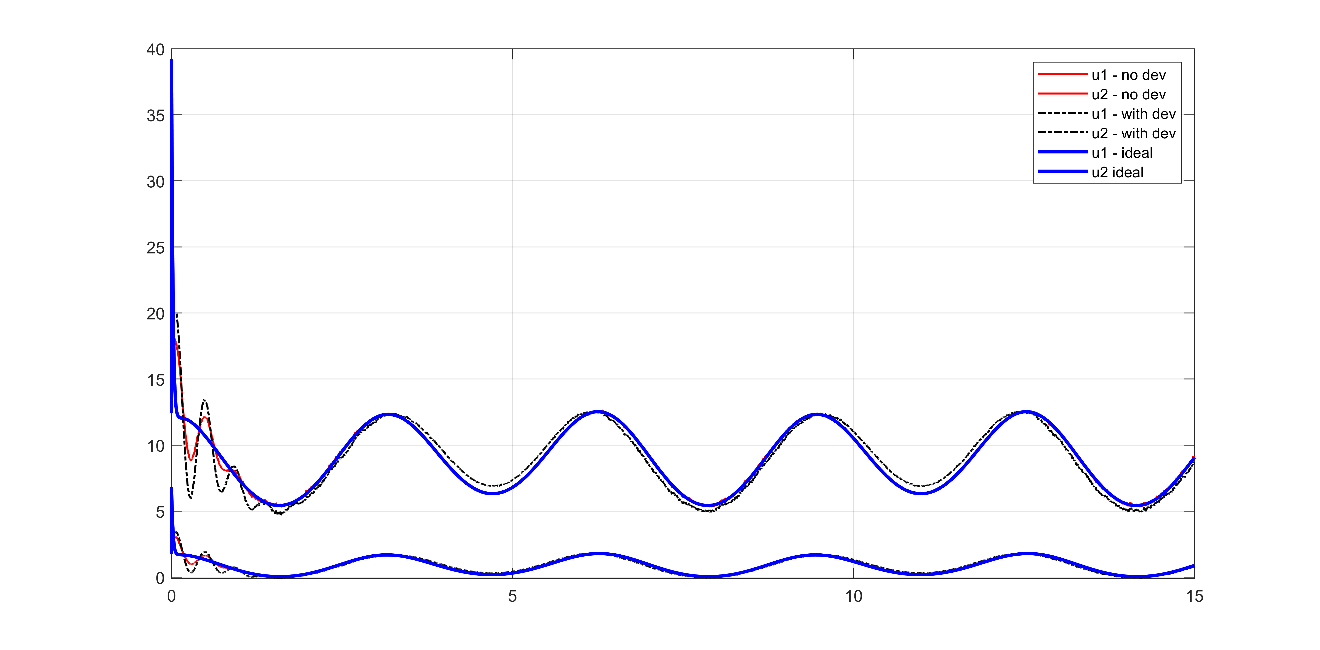


Figure 32: Pole placement+D control sine wave response - joint input torque

### Observations

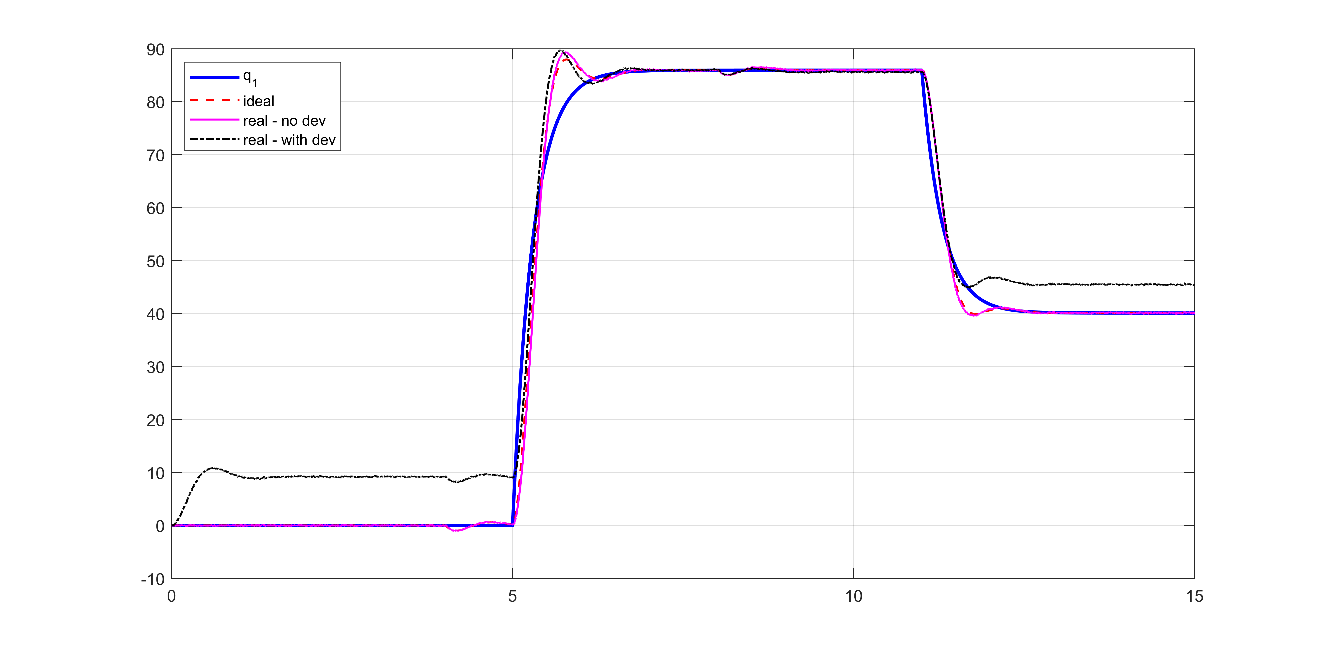
* Zero Steady-state and tracking errors in ideal case but errors are considerable when dynamic parameters deviate.
* Slight overshoot in position and big overshoot in velocity response
* Clear coupling effects between the joints

## PD+Gravity compensation Joint control

Table 6: PD gains for Gravity+PD control

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Ideal case | | | | Realistic case | | | |
| Parameter | Value | Parameter | Value | Parameter | Value | Parameter | Value |
|  | 15 |  | 3.5 |  | 15 |  | 3.5 |
|  | 2.5 |  | 0.5 |  | 2.5 |  | 0.5 |

### Step response



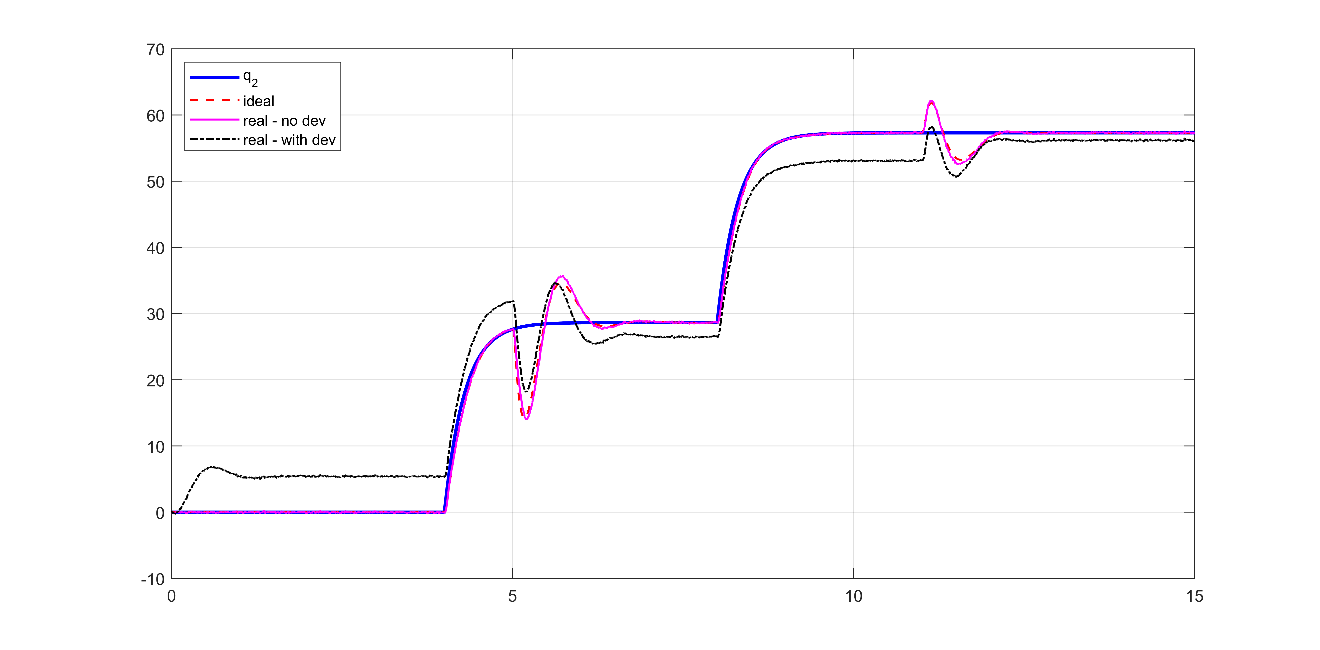
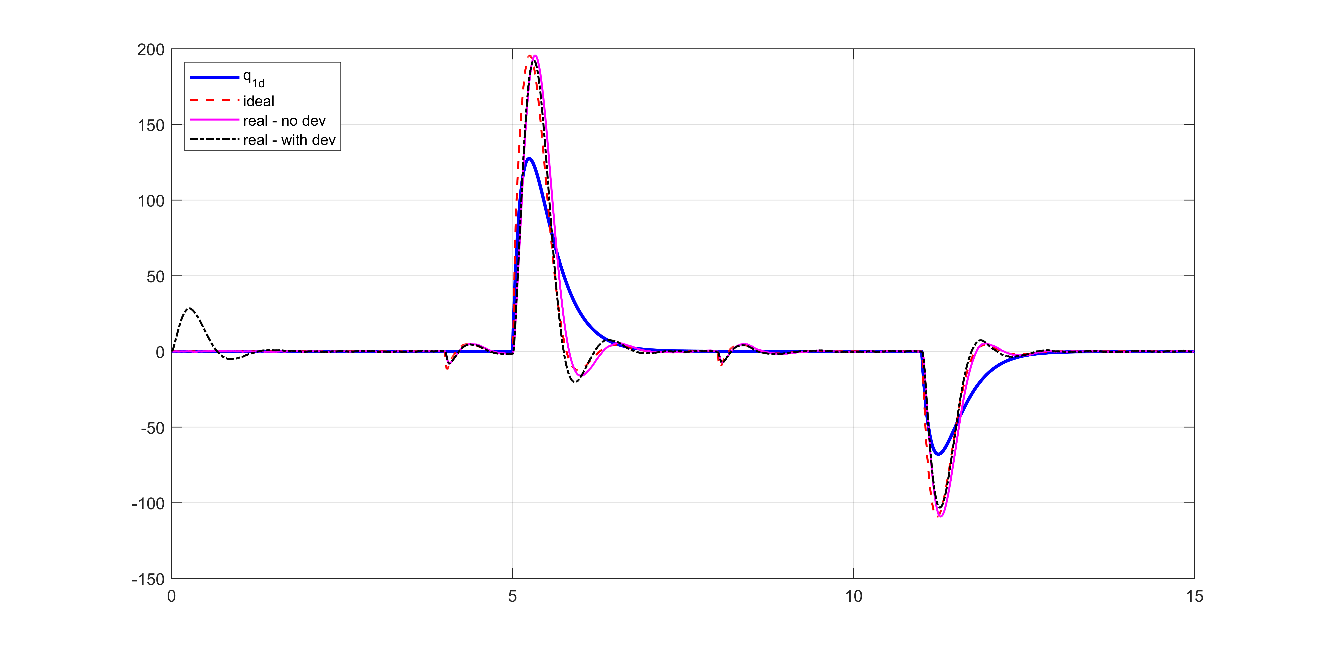


Figure 33: GRAVITY+PD control step response - joint position



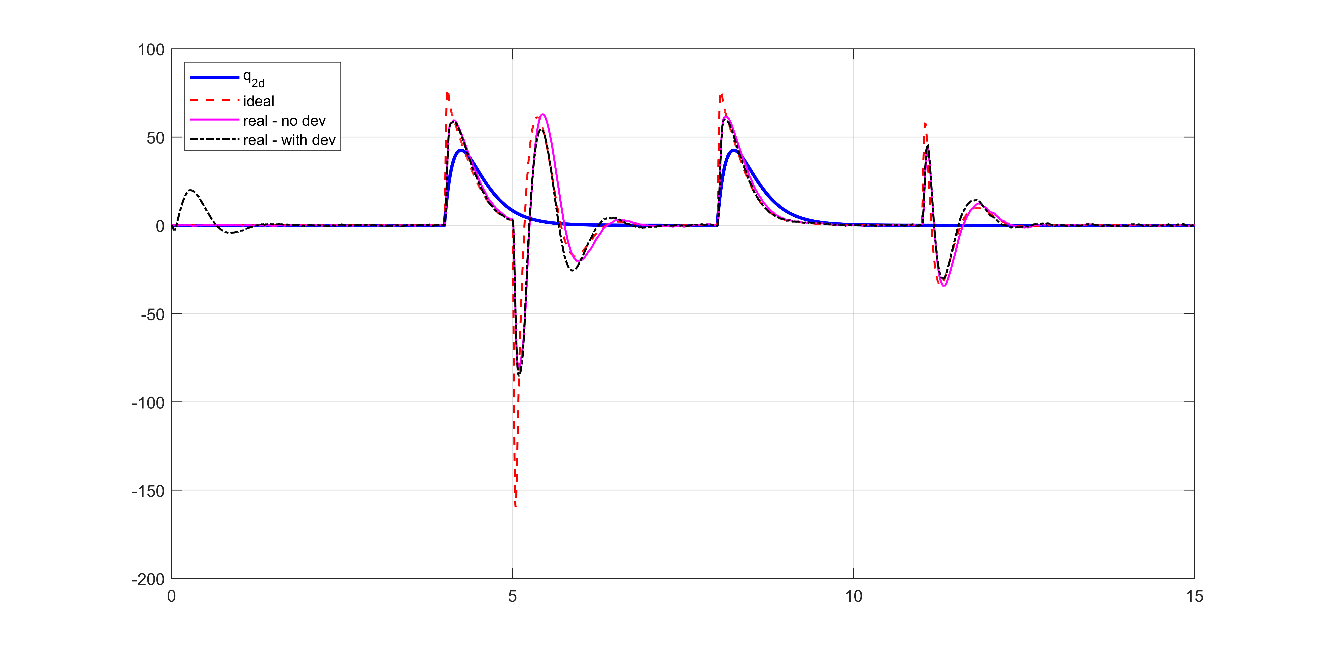


Figure 34: GRAVITY+PD control step response - joint velocity

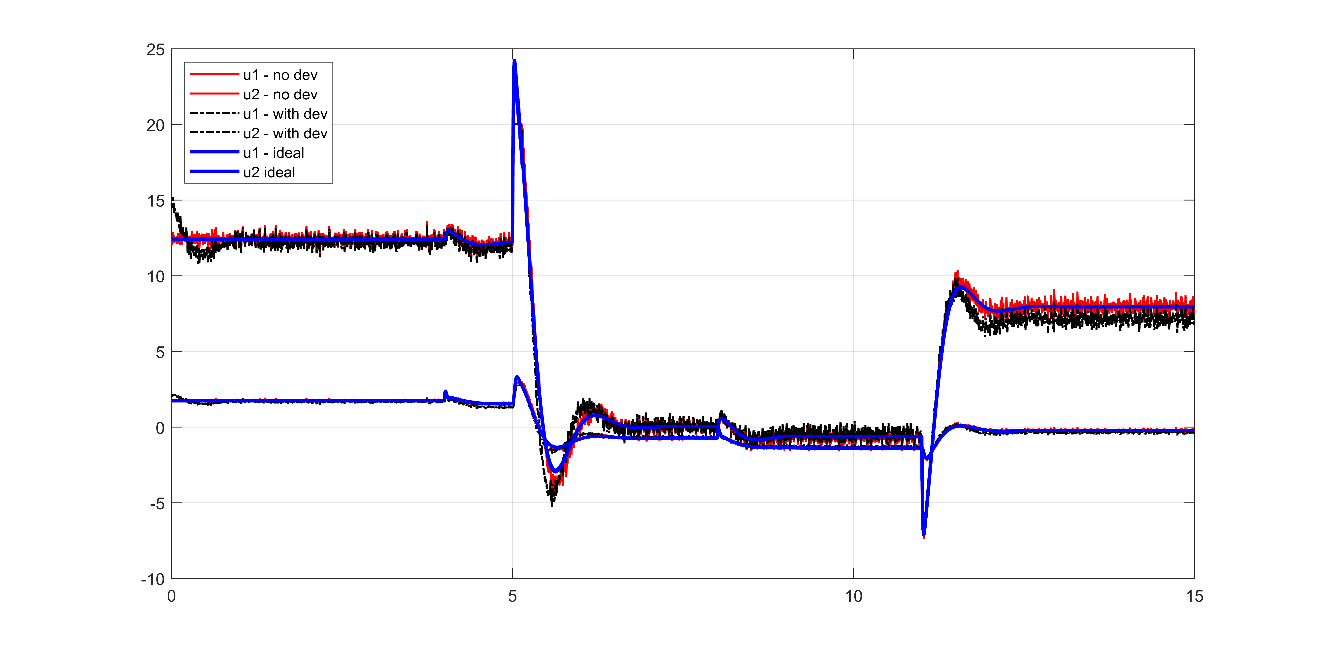
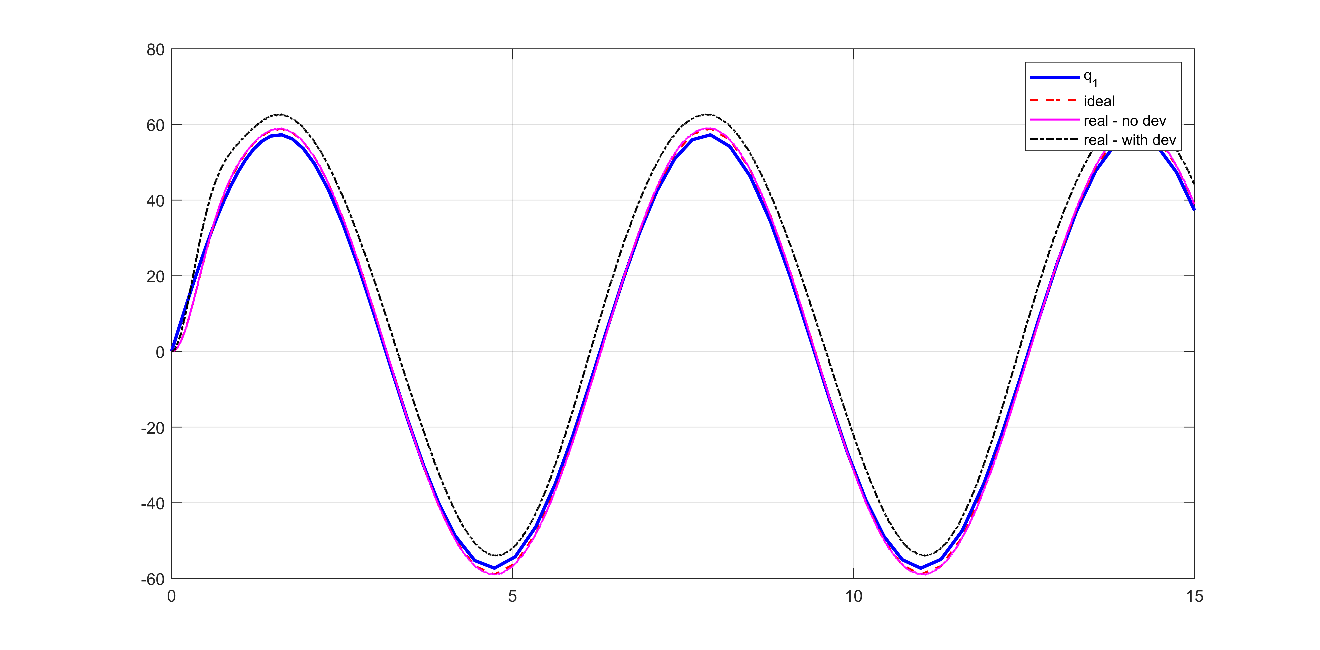


Figure 35: GRAVITY+PD control step response - joint input torque

### Sine wave response



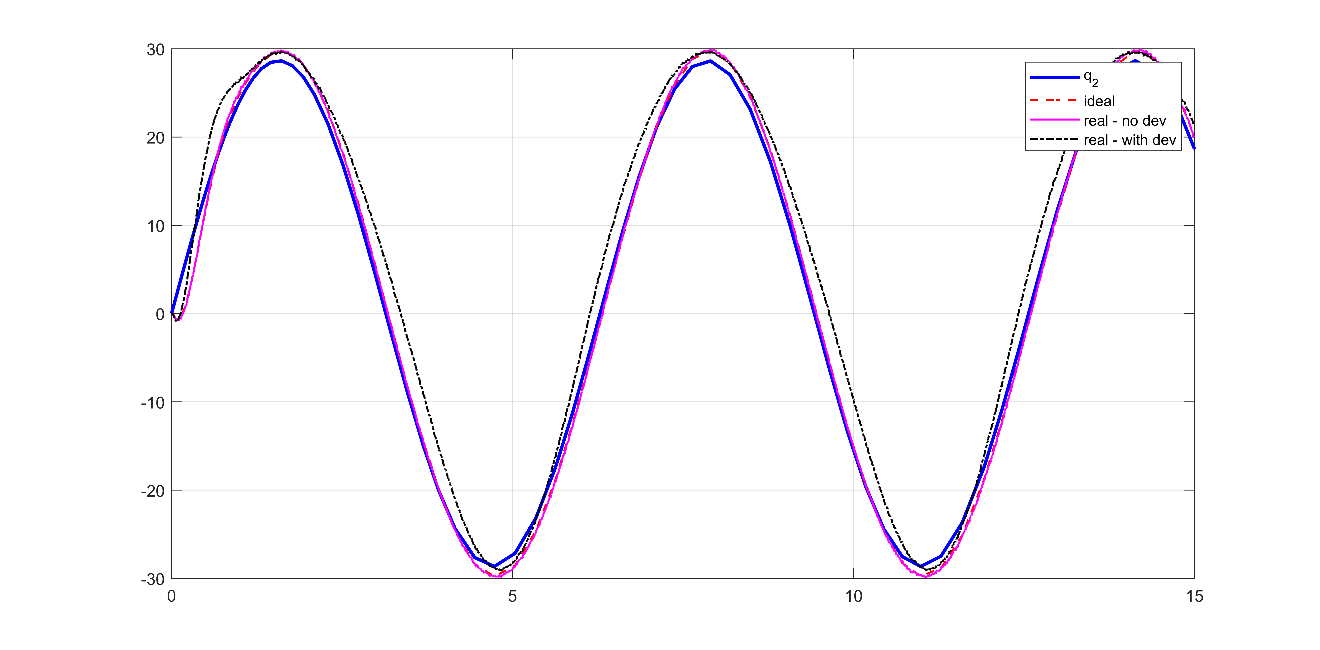
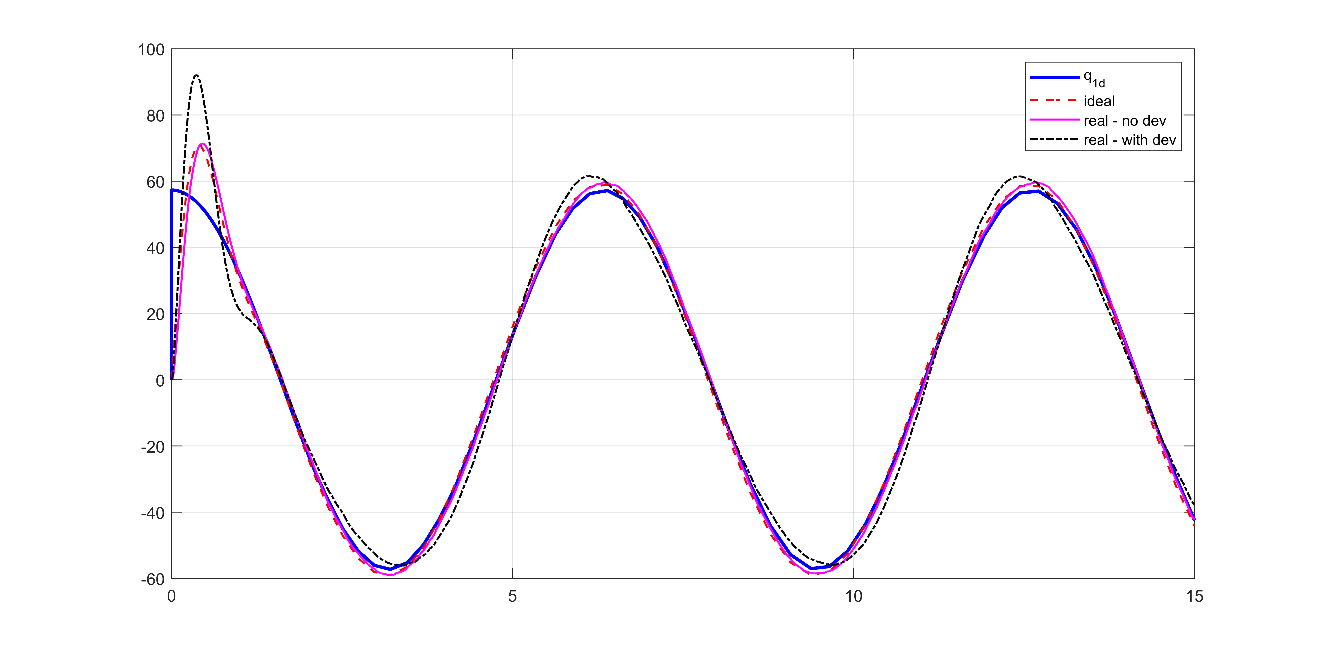


Figure 36: GRAVITY+PD control sine wave response - joint position



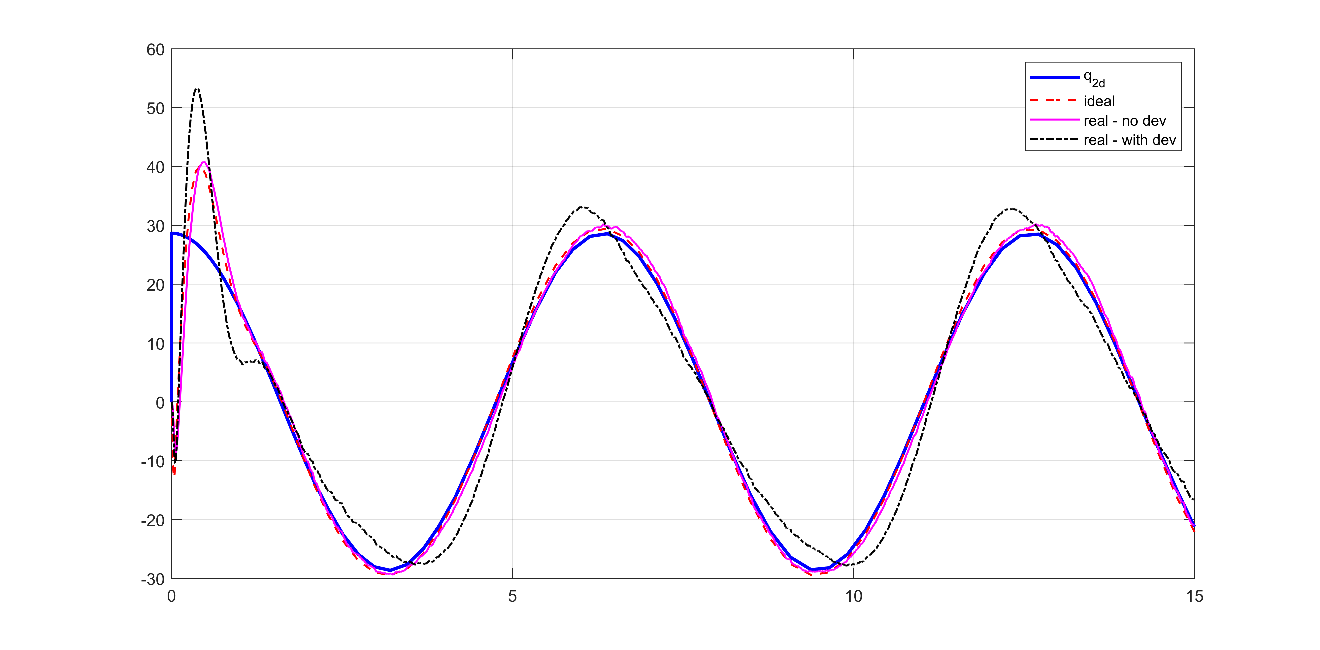


Figure 37: GRAVITY+PD control sine wave response - joint velocity

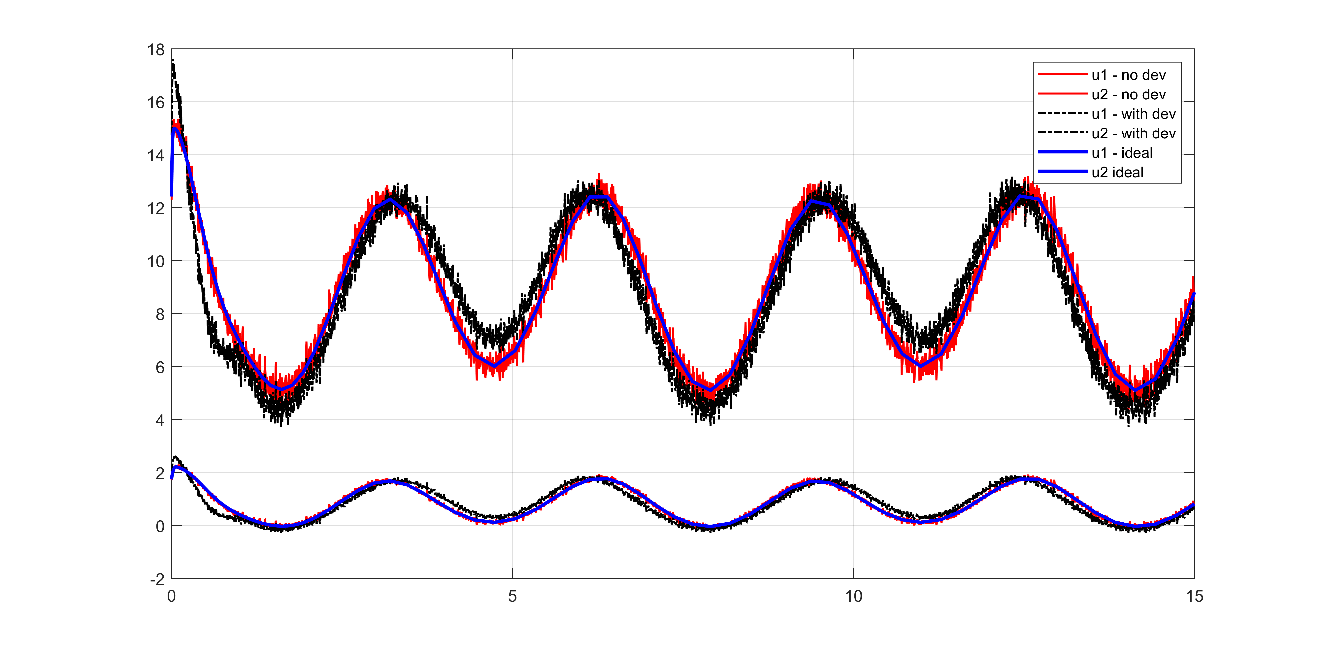


Figure 38: GRAVITY+PD control sine wave response - joint input torque

### Observations

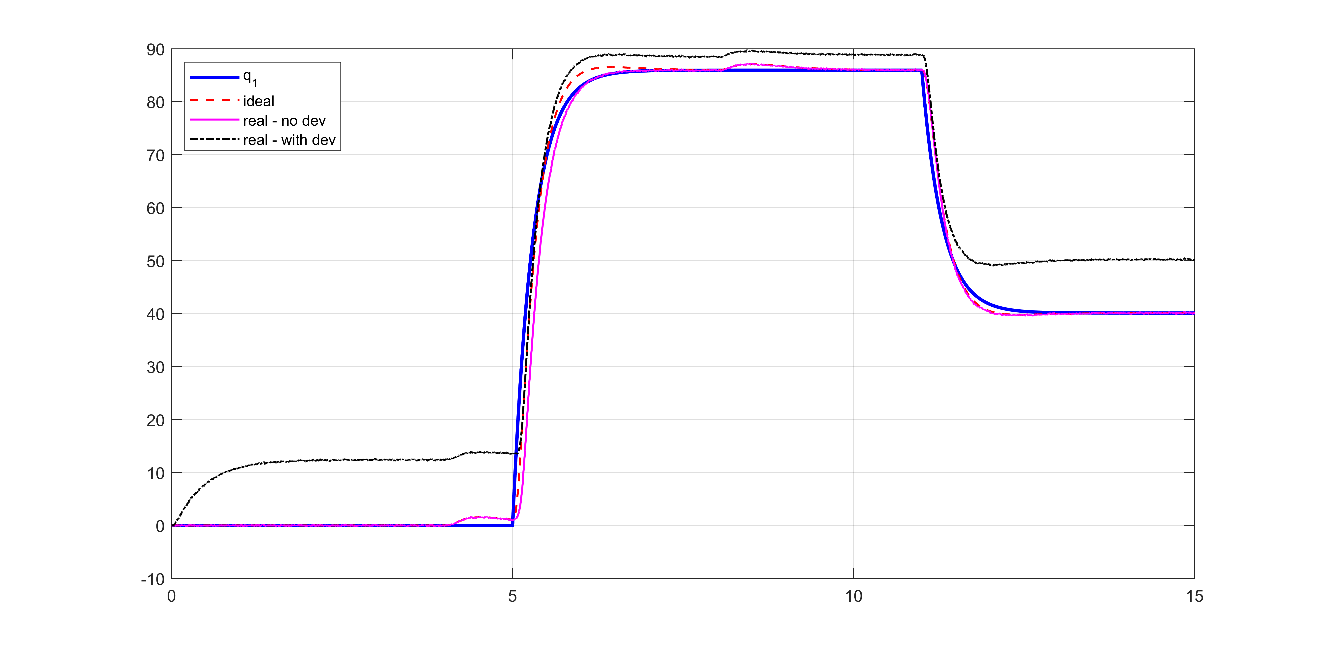
* Steady-state converges close to zero and tracking error is within 1 degree if dynamic parameters are accurate.
* Noticeable steady-state and tracking errors when dynamic parameters deviate.
* Huge improvements compared to PD-only control
* Increasing PD gains will result in fast settling time and zero steady-state errors but will introduce high-frequency oscillations in commanded torques.

## Inverse dynamics control

Table 7: PD gains – Inverse control

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Ideal case | | | | Realistic case | | | |
| Parameter | Value | Parameter | Value | Parameter | Value | Parameter | Value |
|  | 25 |  | 50 |  | 25 |  | 50 |
|  | 12 |  | 15 |  | 12 |  | 15 |

### Step response



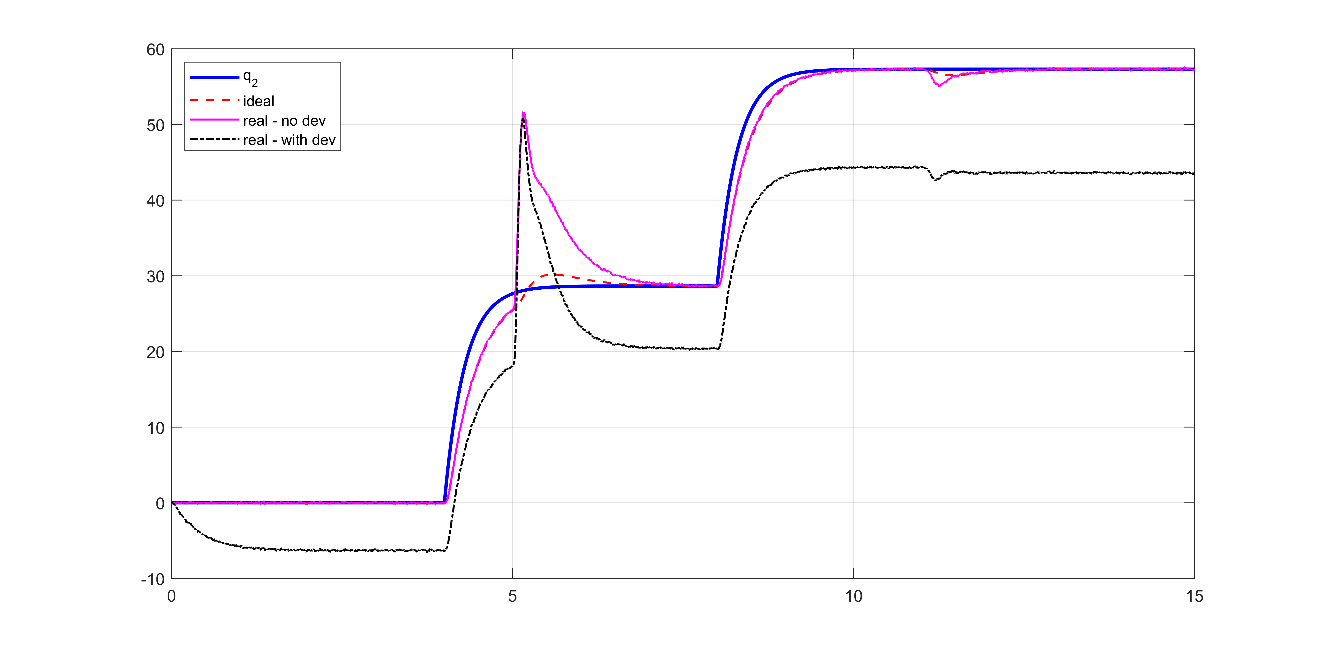
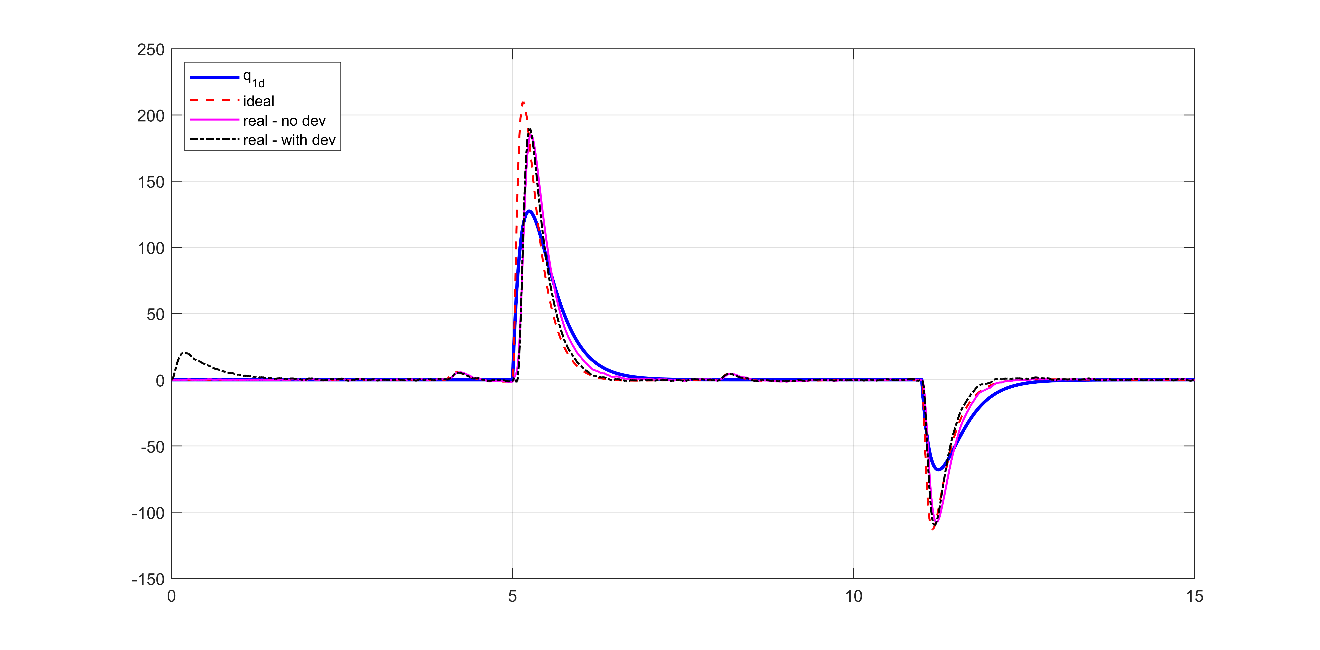


Figure 39: Inverse dynamics control step response - joint position



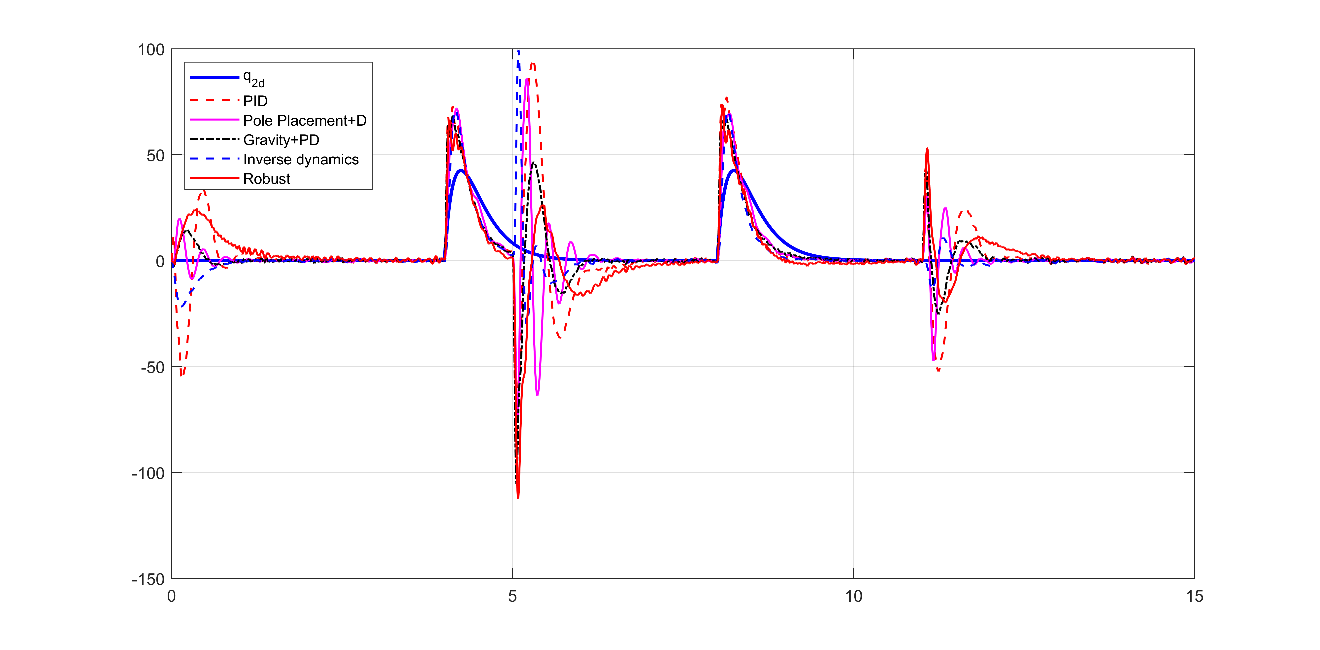


Figure 40: Inverse dynamics control step response - joint velocity

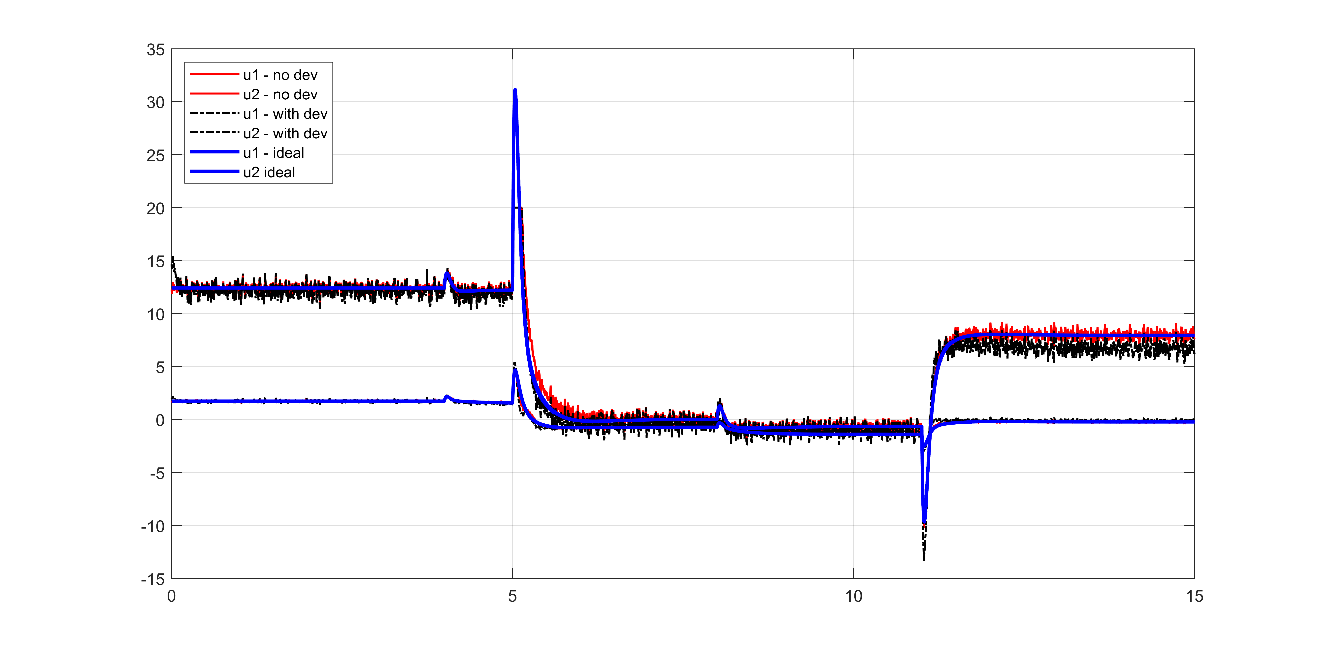
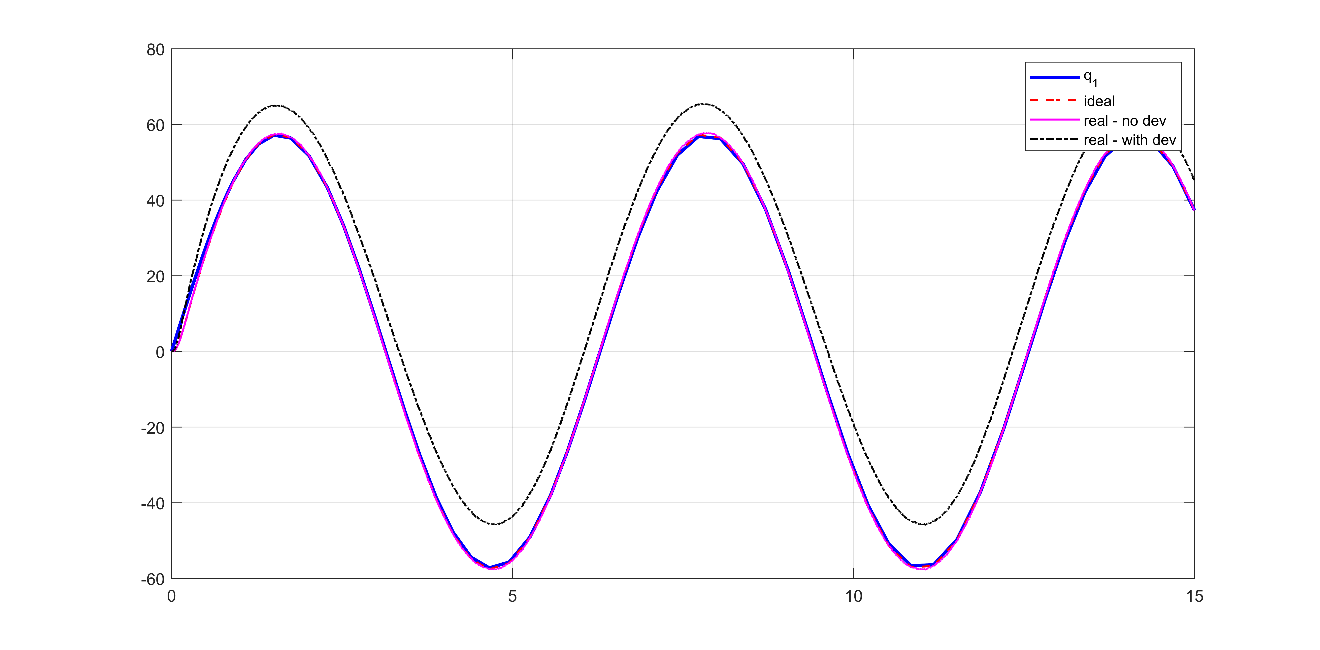


Figure 41: Inverse dynamics control step response - joint input torque

### Sine wave response



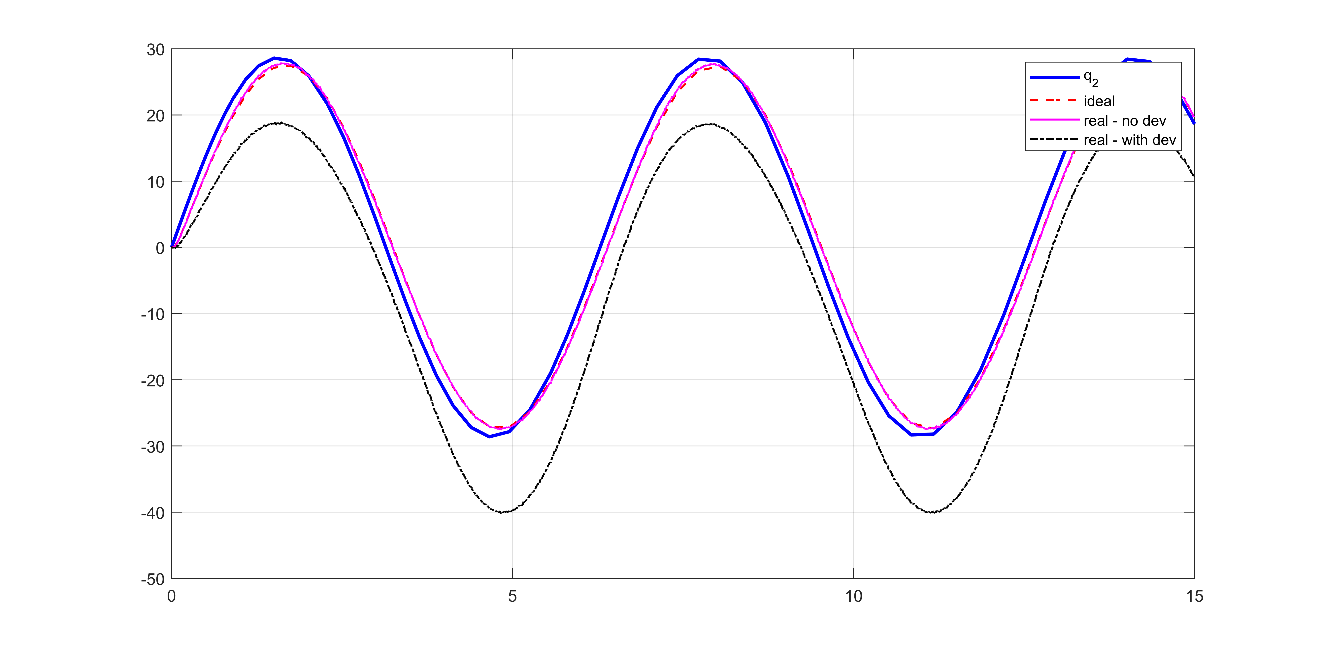
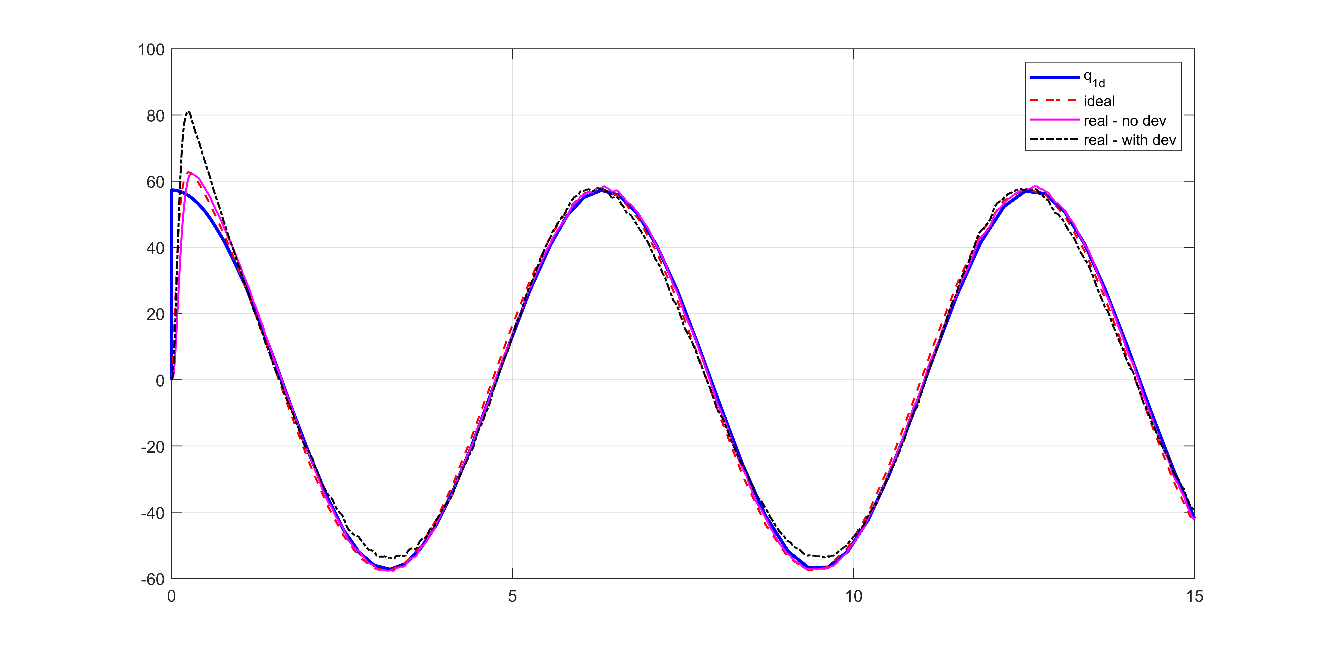


Figure 42: Inverse dynamics control sine wave response - joint position



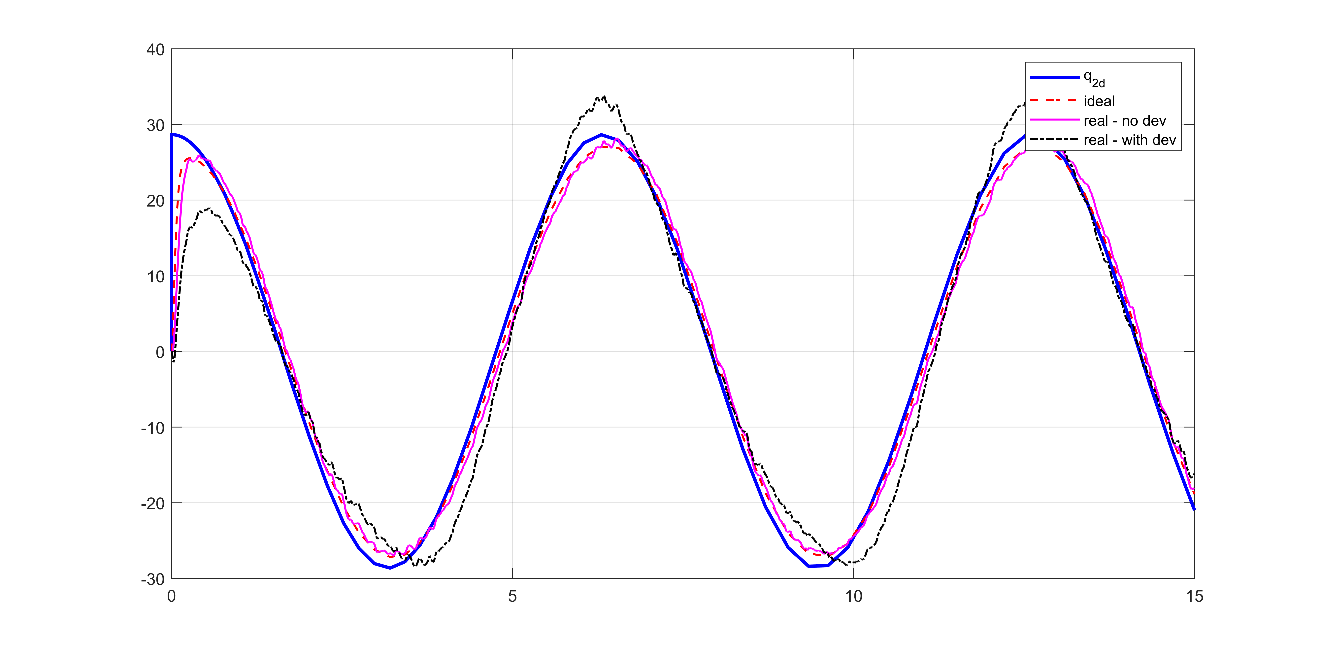


Figure 43: Inverse dynamics control sine wave response - joint velocity

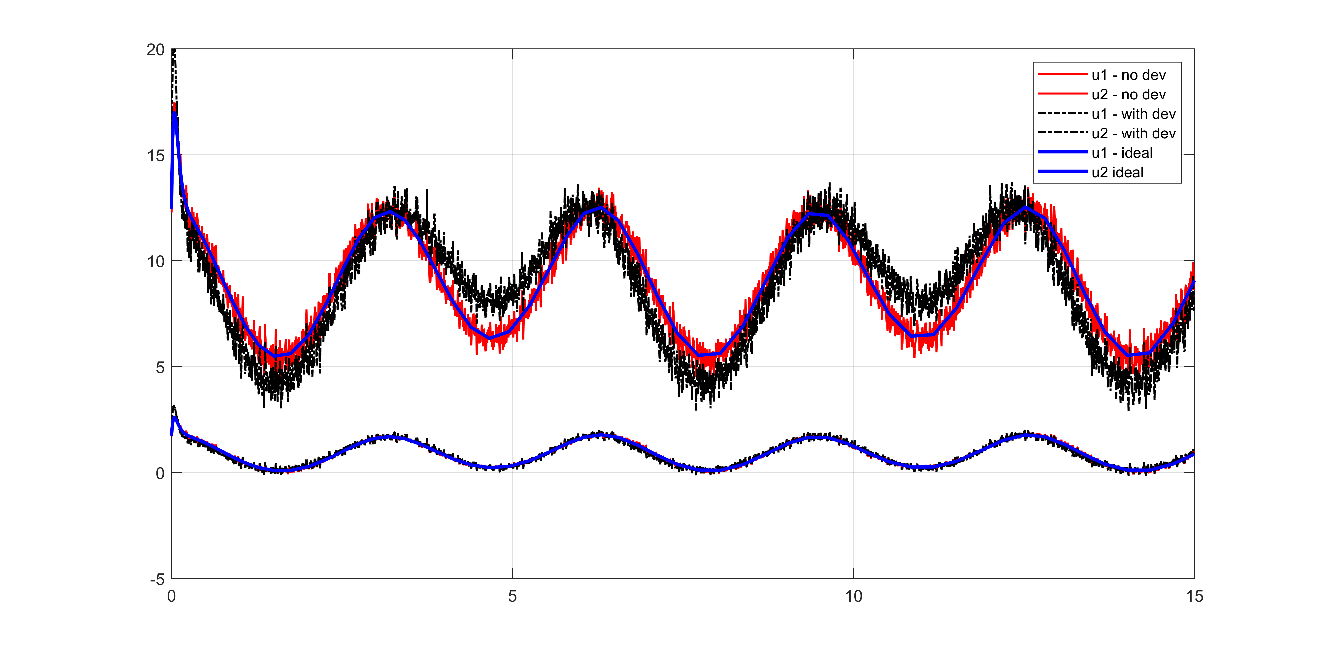


Figure 44: Inverse dynamics control sine wave response - joint input torque

### Observations

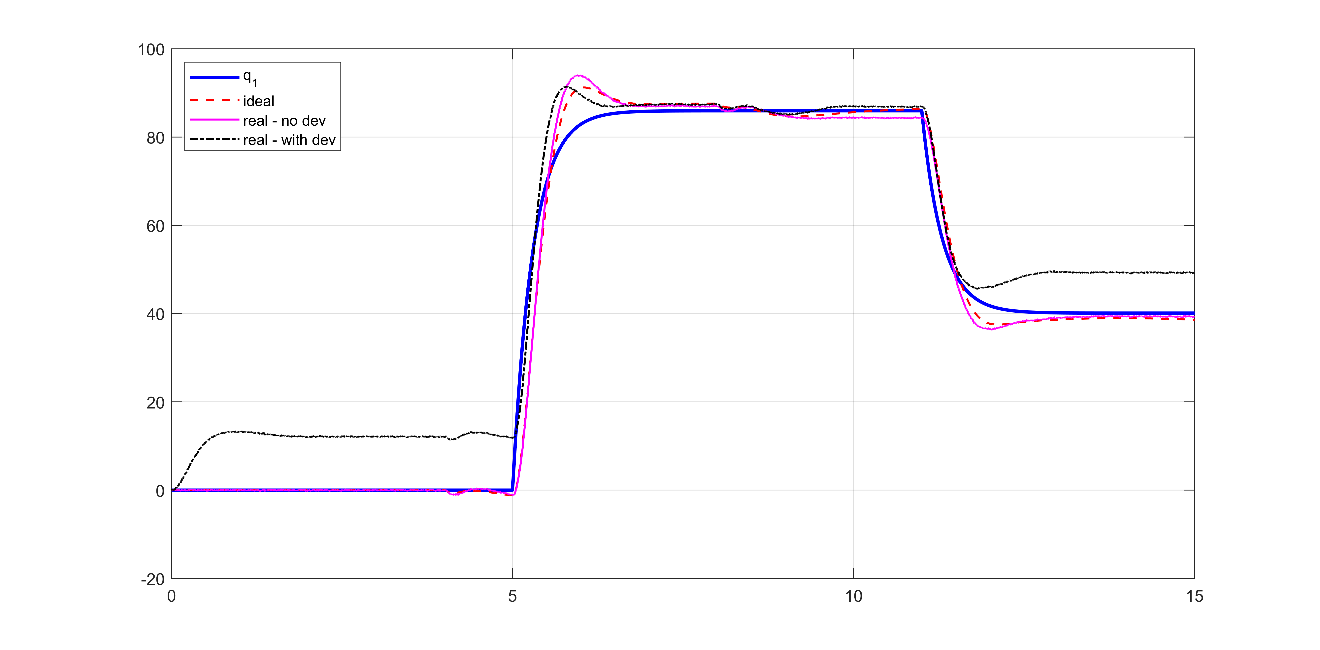
* Steady-state converges to zero and tracking error is much better than Gravity+PD control in ideal case
* Huge improvements compared to PD-only control and slight improvement compared to gravity+PD control when there is accurate parameter estimation (no deviation)
* Noticeable steady-state and tracking errors when dynamic parameters deviate.
* Increasing PD gains may result in fast settling time and zero steady-state errors but will introduce high-frequency oscillations in commanded torques.

## Robust control

Table 8: PD gains for robust control – Realistic system

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Value | Parameter | Value |
|  | 25 |  | 45 |
|  | 7 |  | 35 |
|  | 0.1 |  | 1 |
|  |  |  |  |

### Step response



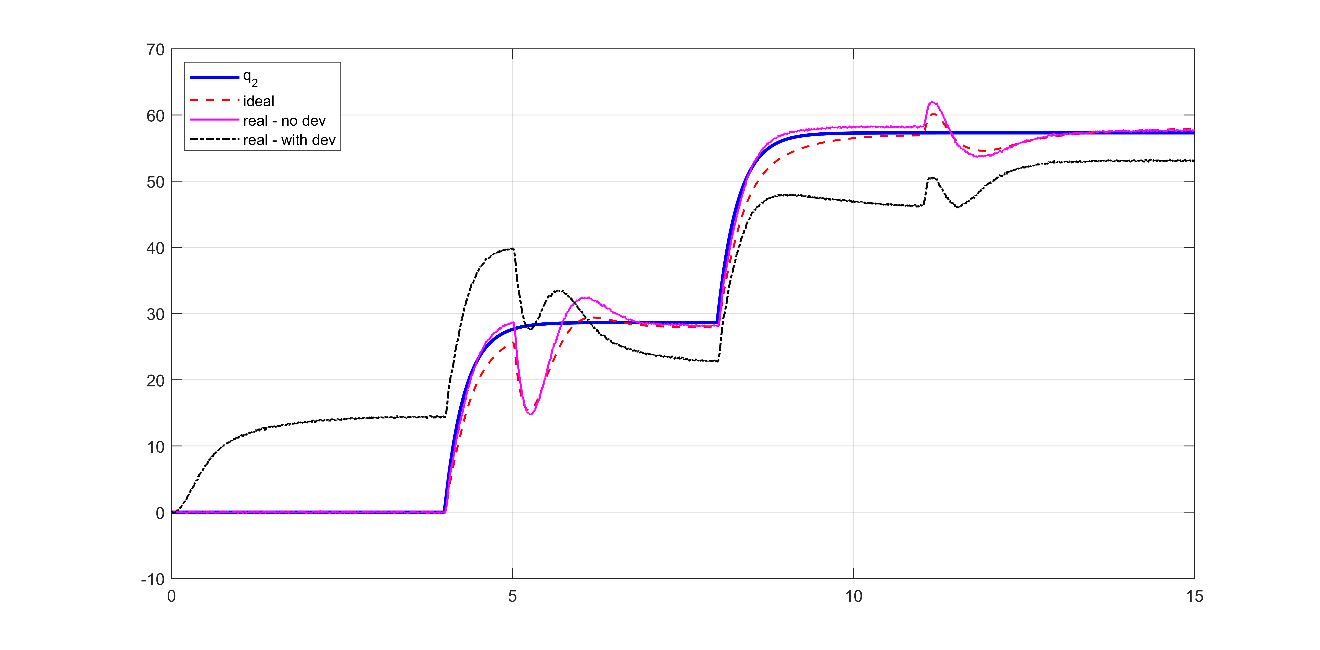
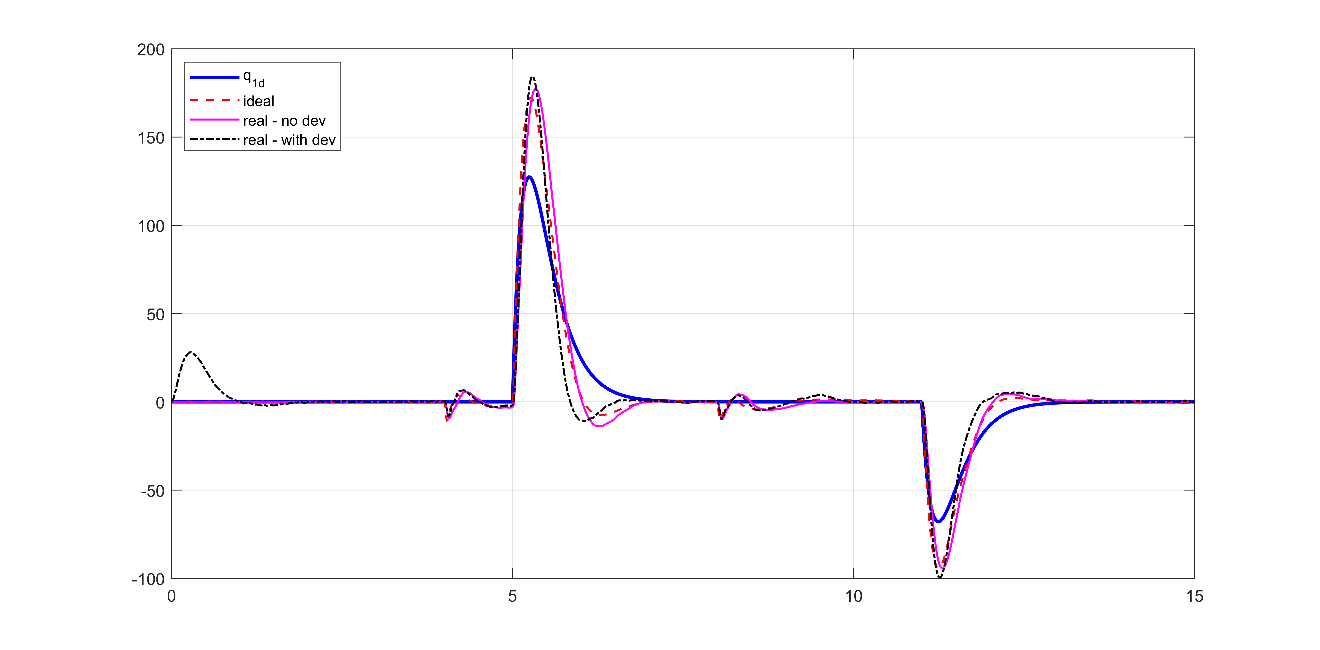


Figure 45: Robust control realistic system step response - joint position



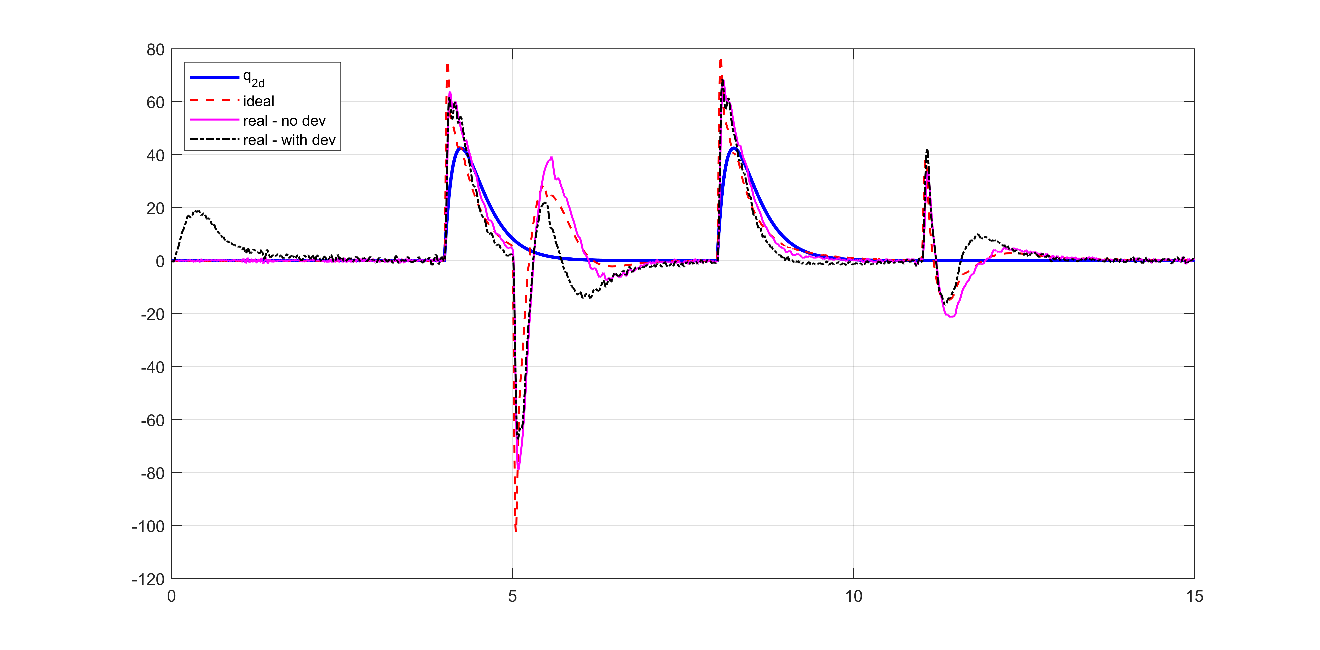


Figure 46: Robust control realistic system step response - joint velocity

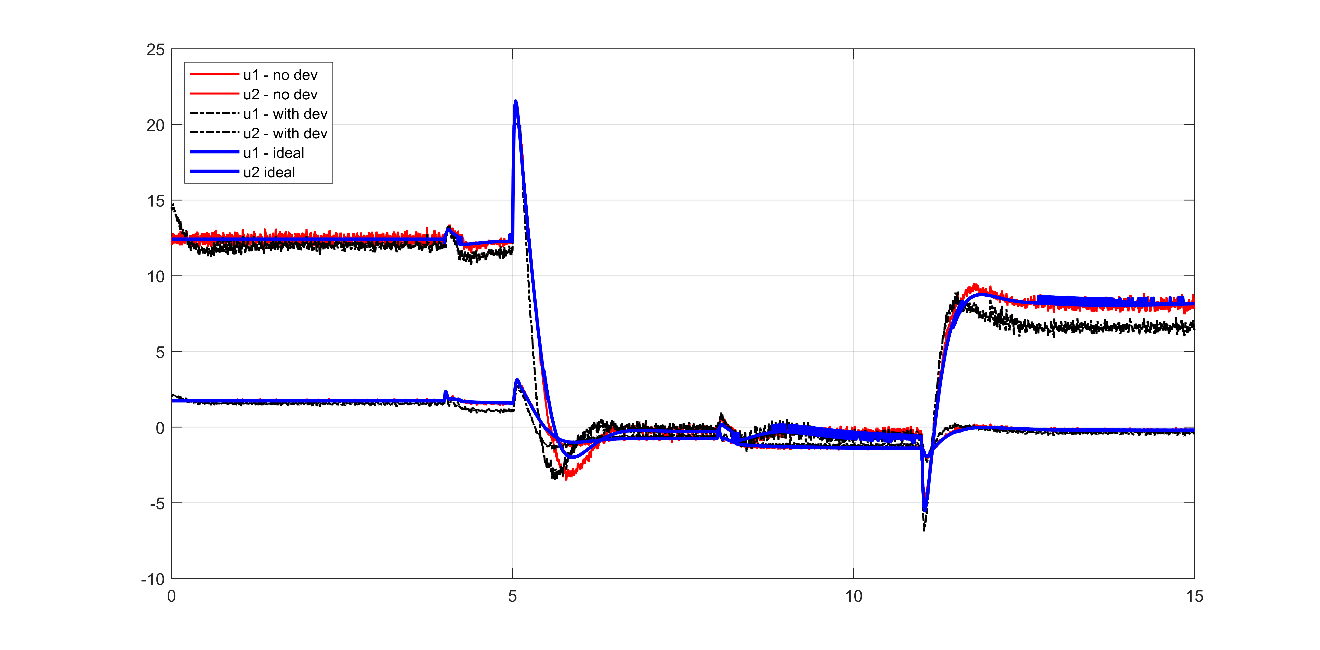
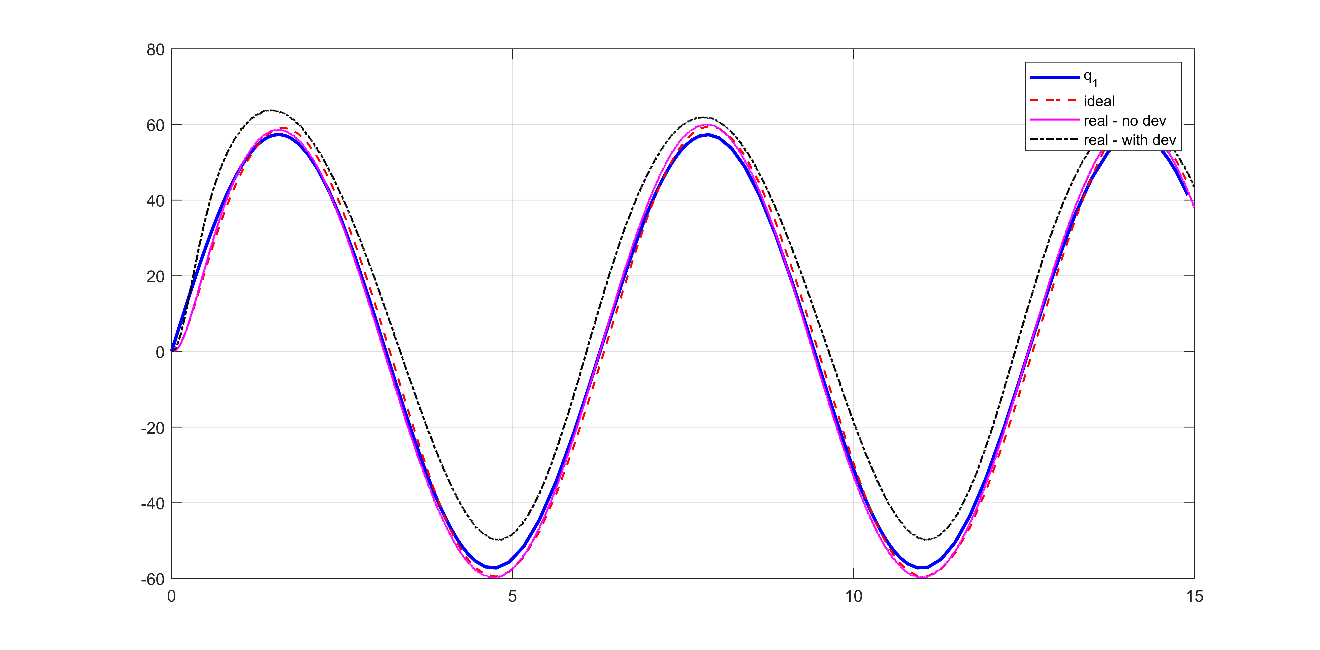


Figure 47: Robust control realistic system step response - joint input torque

### Sine wave response



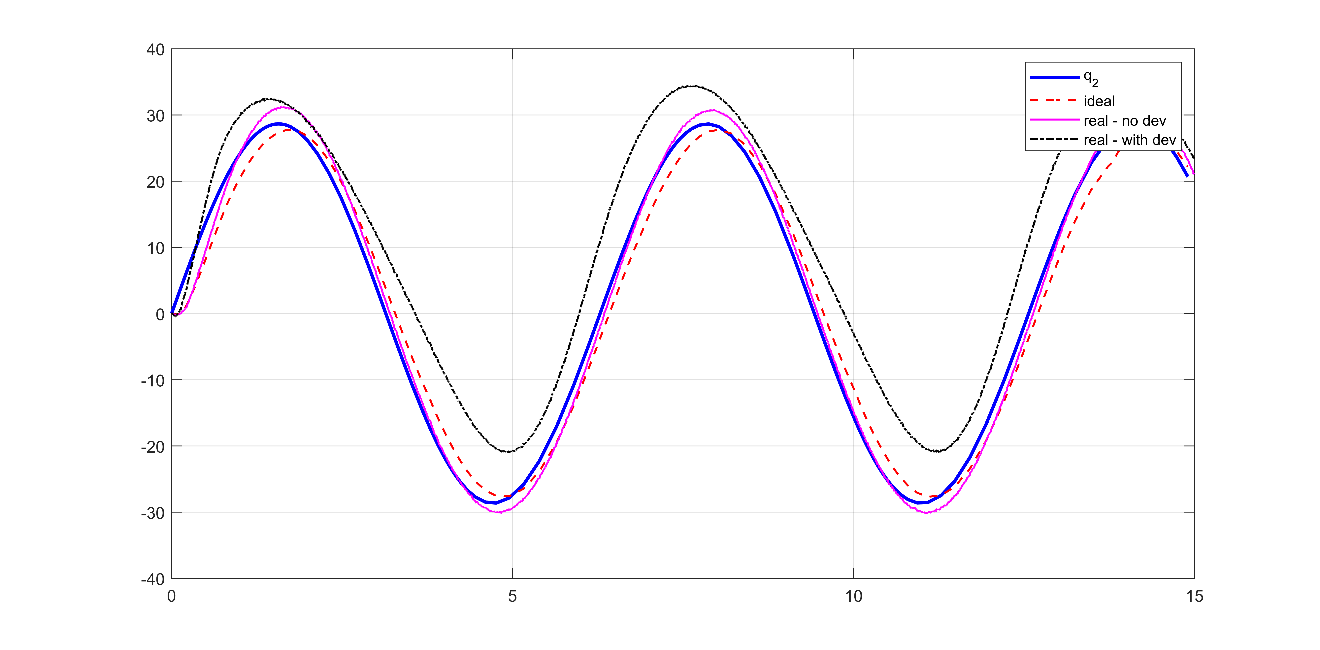
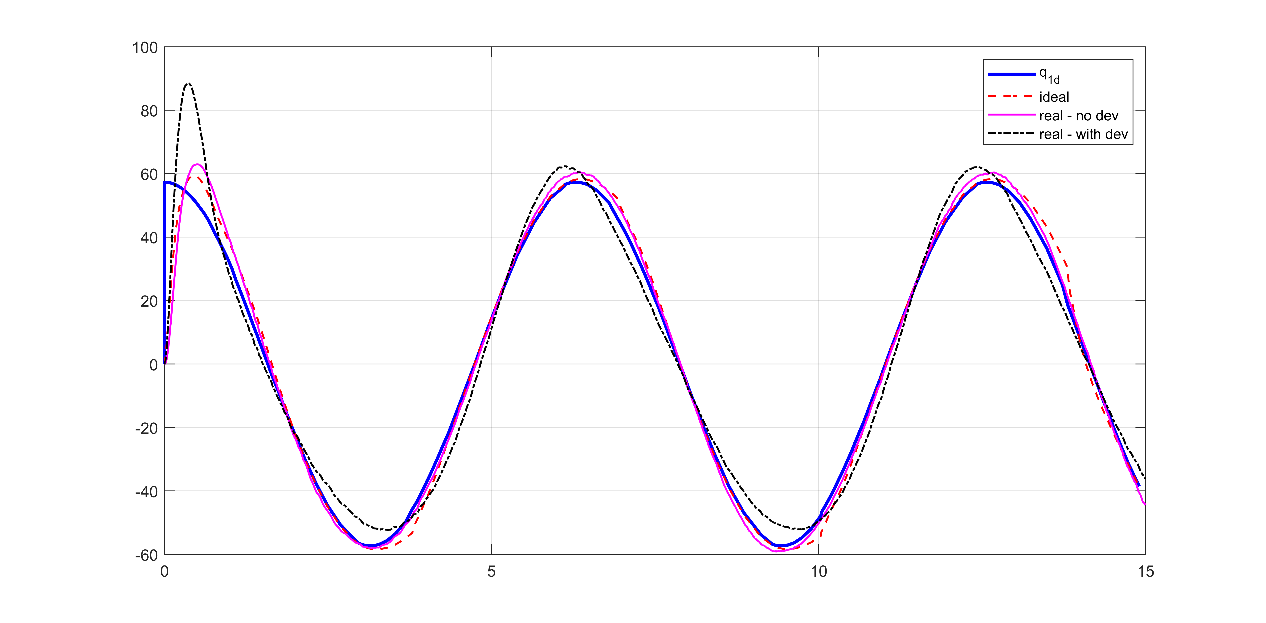


Figure 48: Robust control realistic system sine wave response - joint position



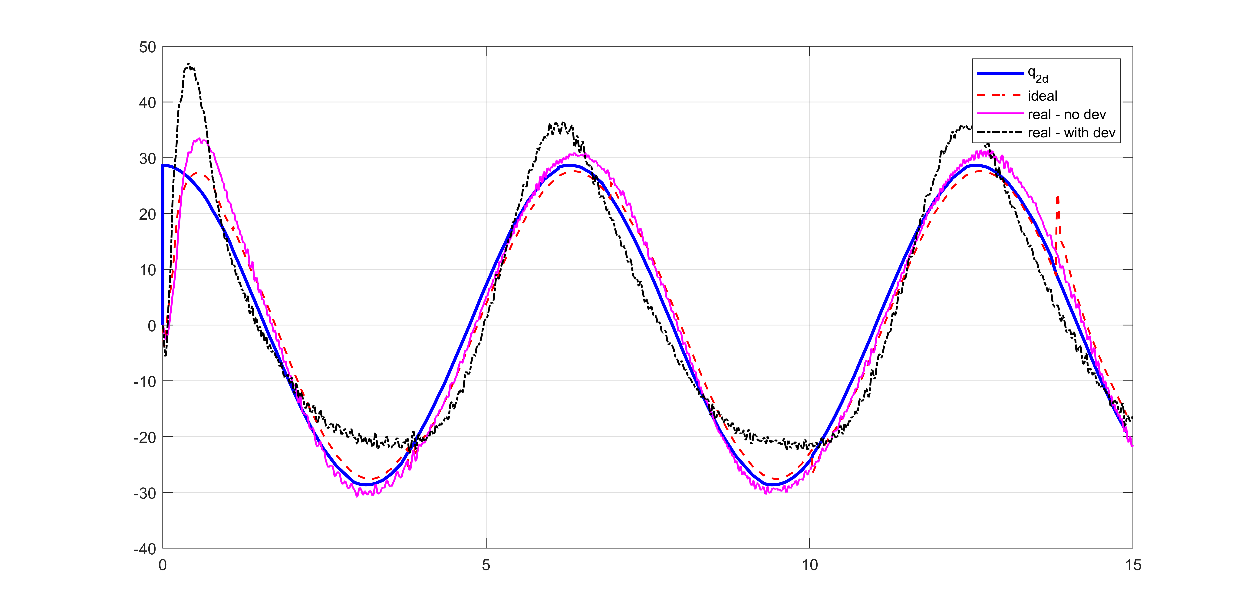


Figure 49: Robust control realistic system sine wave response - joint velocity

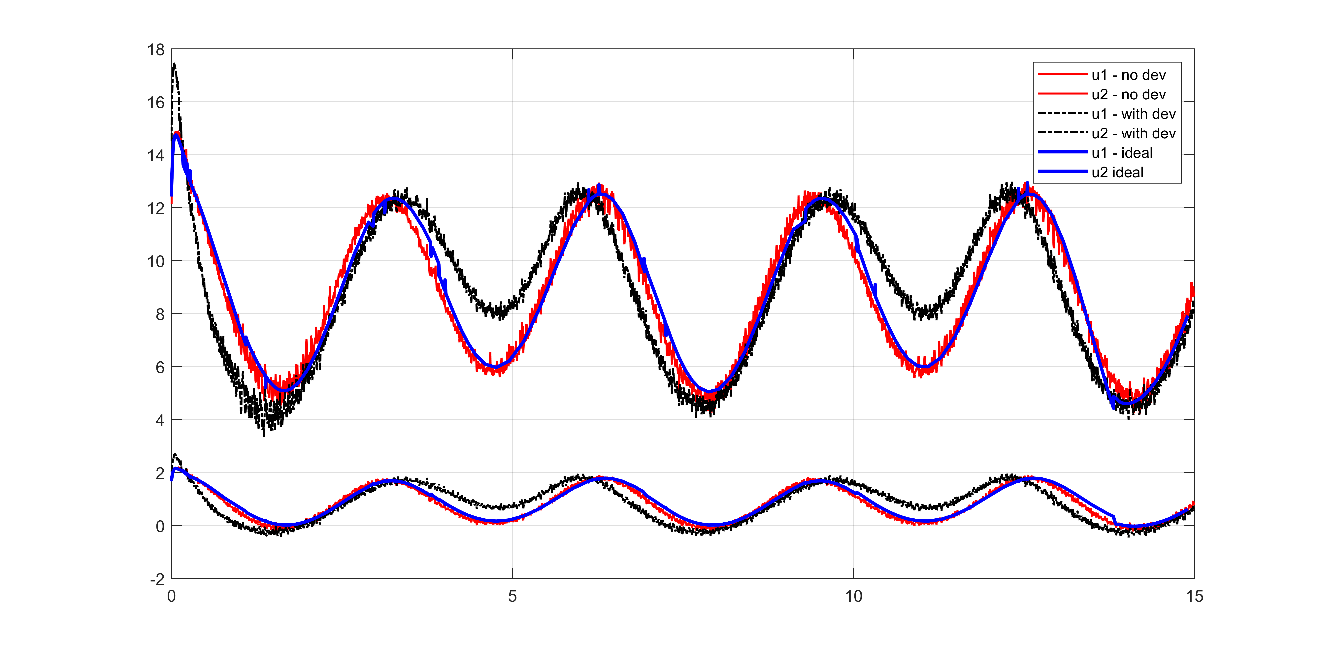


Figure 50: Robust control realistic system sine wave response - joint input torque

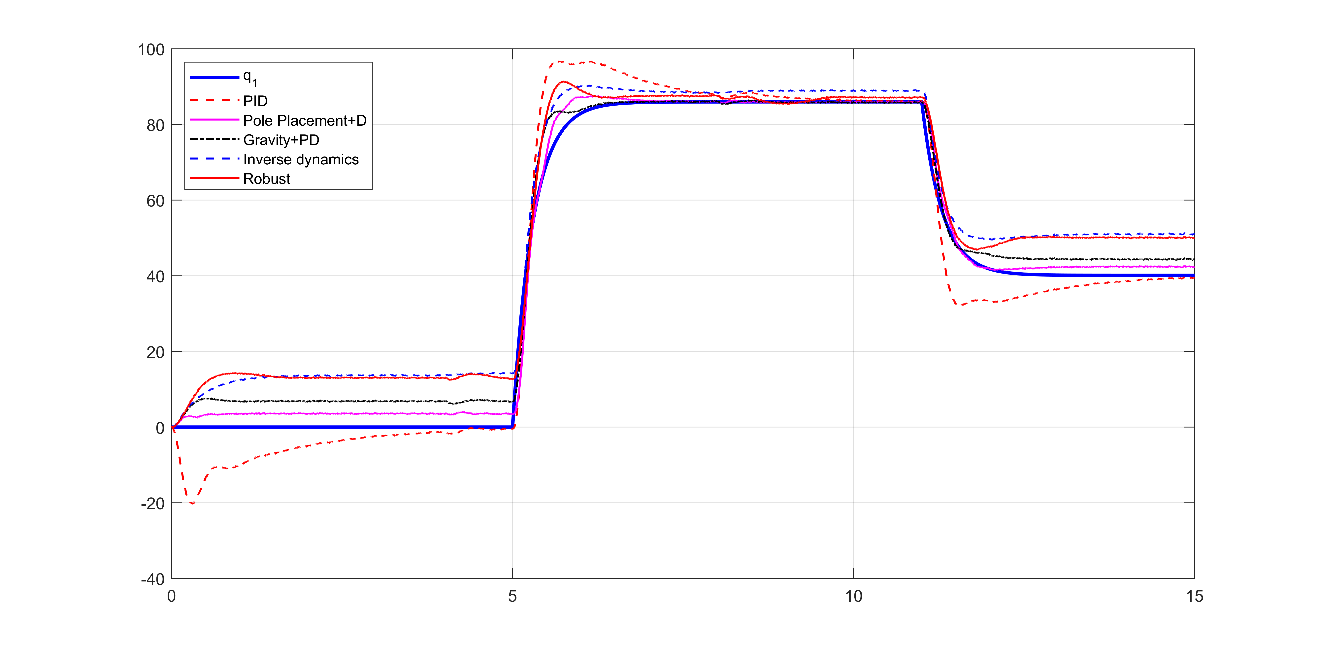
### Observations

* Worse in performance than inverse dynamics control and gravity+PD control
* Very hard to tune and robust terms don’t seem to provide robustness to deviations

# Simulation results – Comparison

* This section compares potentially good controllers from previous section
* This section continues with realistic case and dynamic parameter deviation feeding to the controllers
* The plots are exactly the same as previous section but combined in one figure for clearer comparison
* The controllers that are considered are:
  + PID control
  + Pole-Placement + Disturbance control
  + Gravity+PD control
  + Inverse dynamics control
  + Robust control
* Best controllers will be evaluated further in next section

### **Step response**



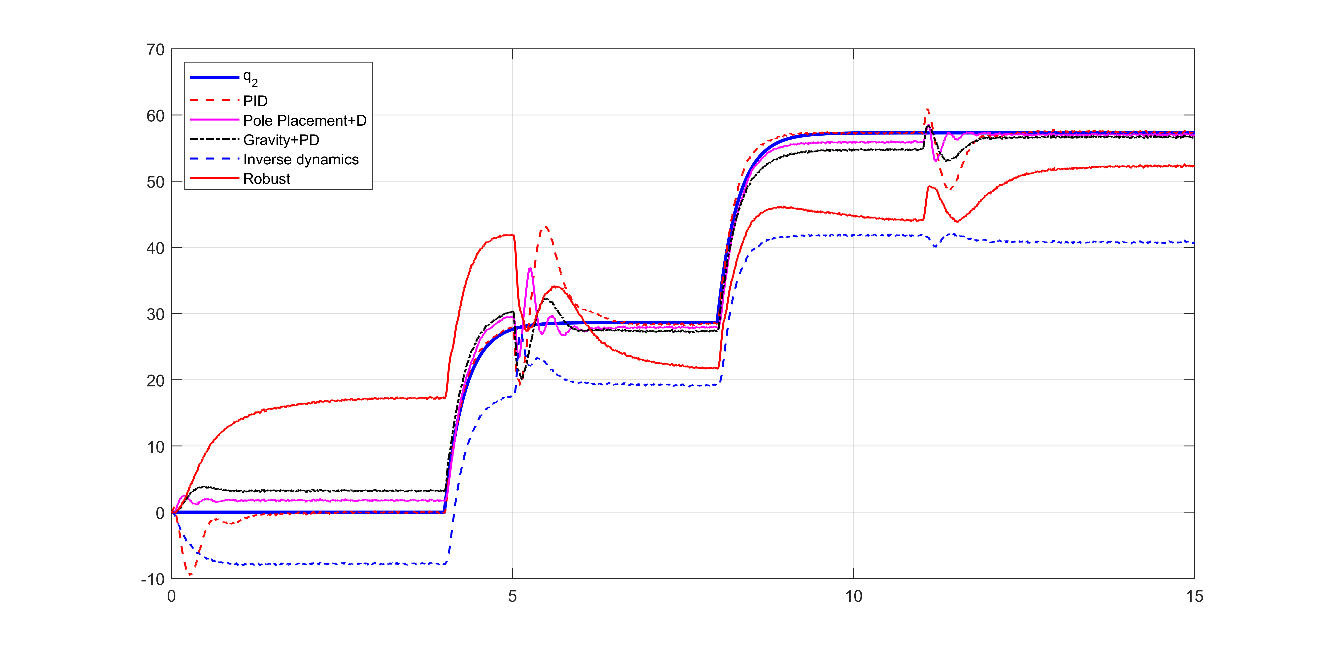
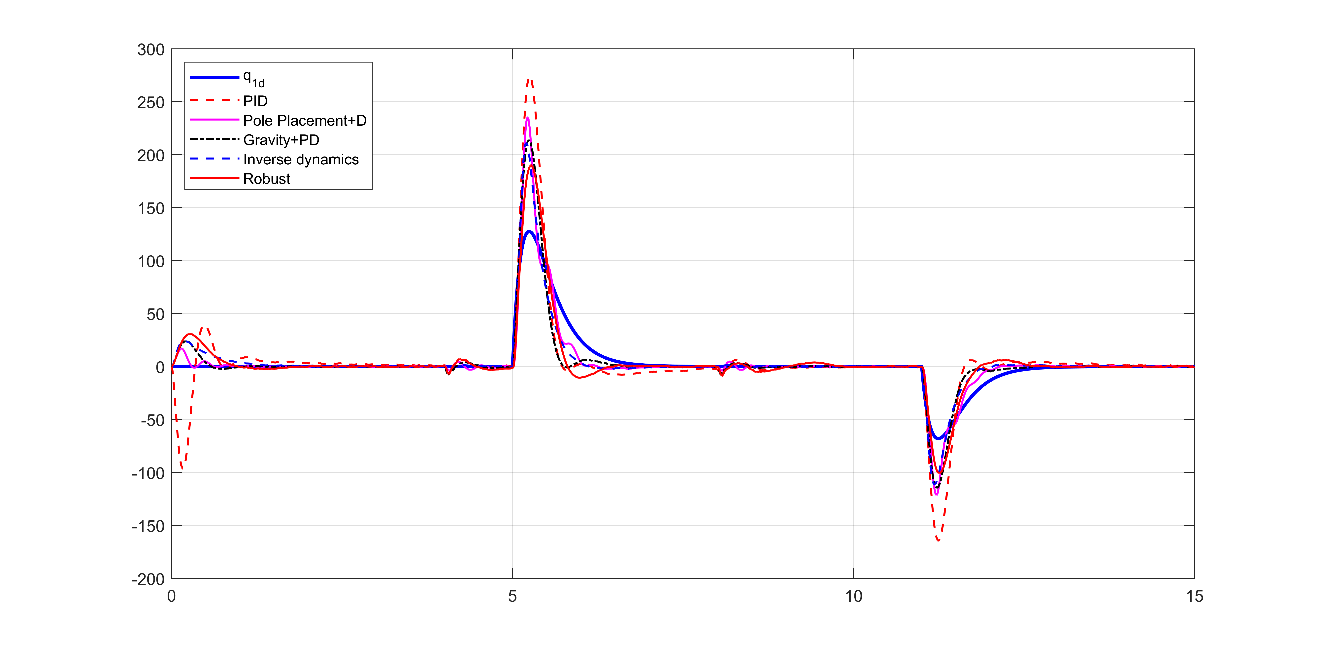


Figure 51: step response comparison - joint position



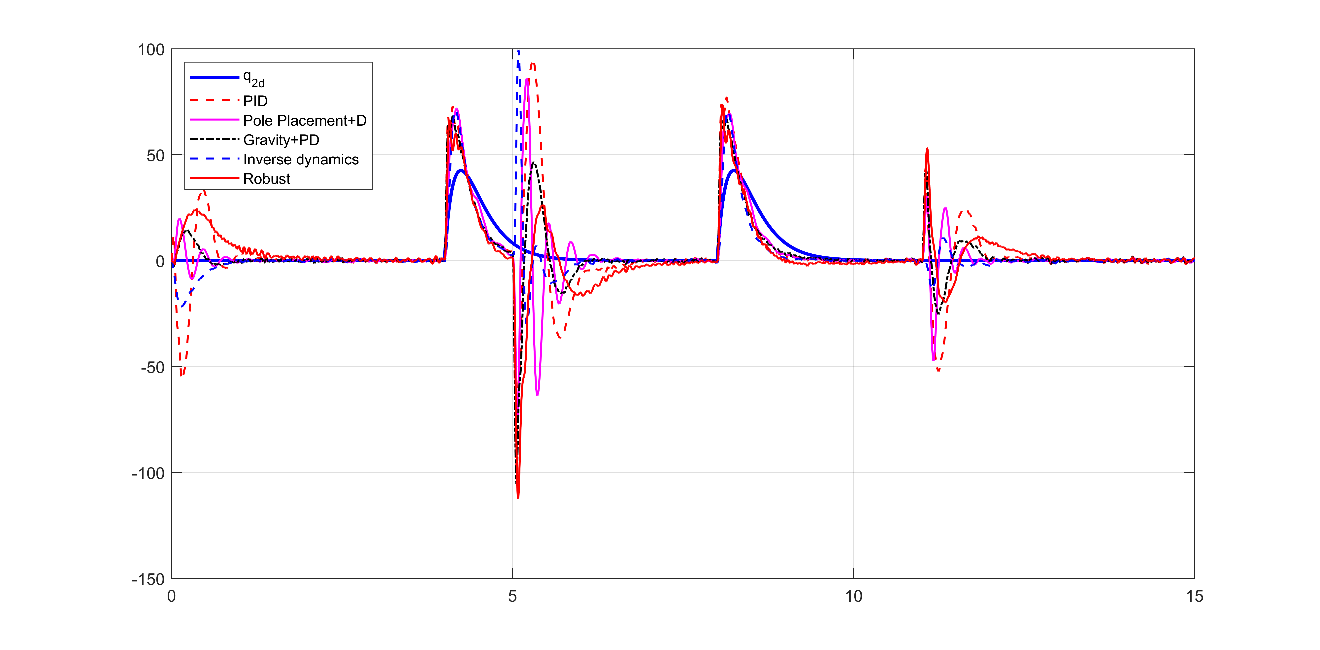
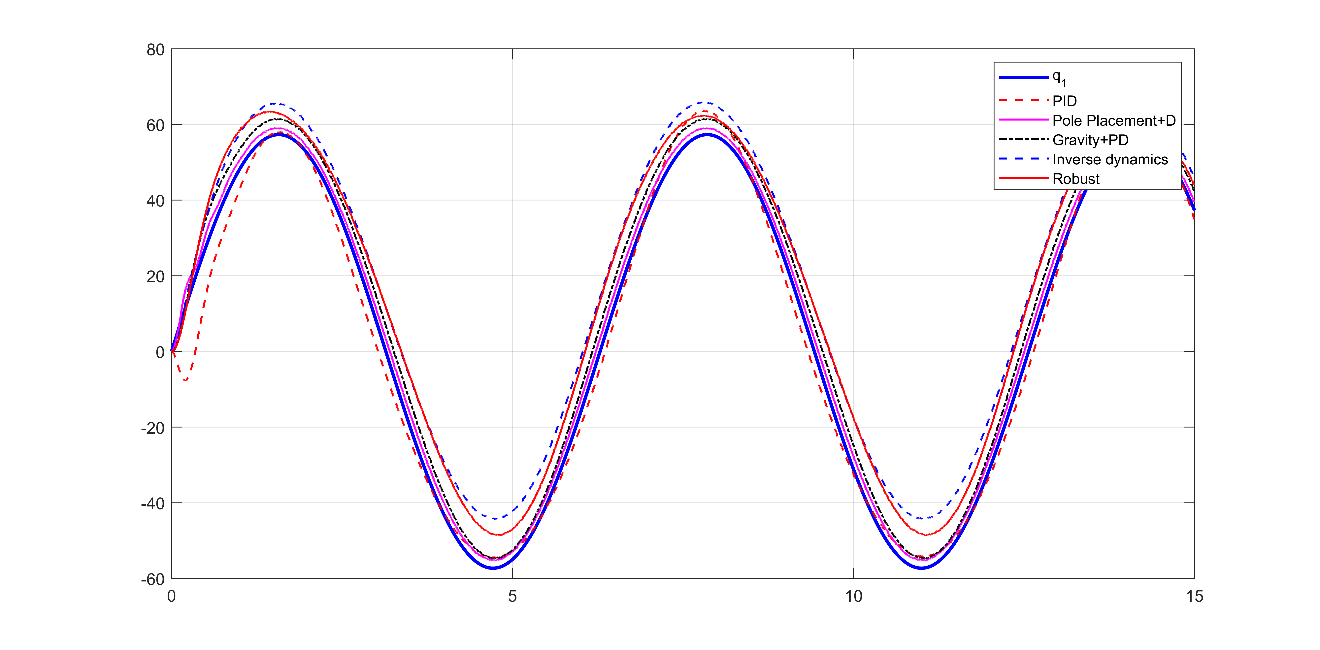


Figure 52: step response comparison - joint velocity

### **Sine wave response**



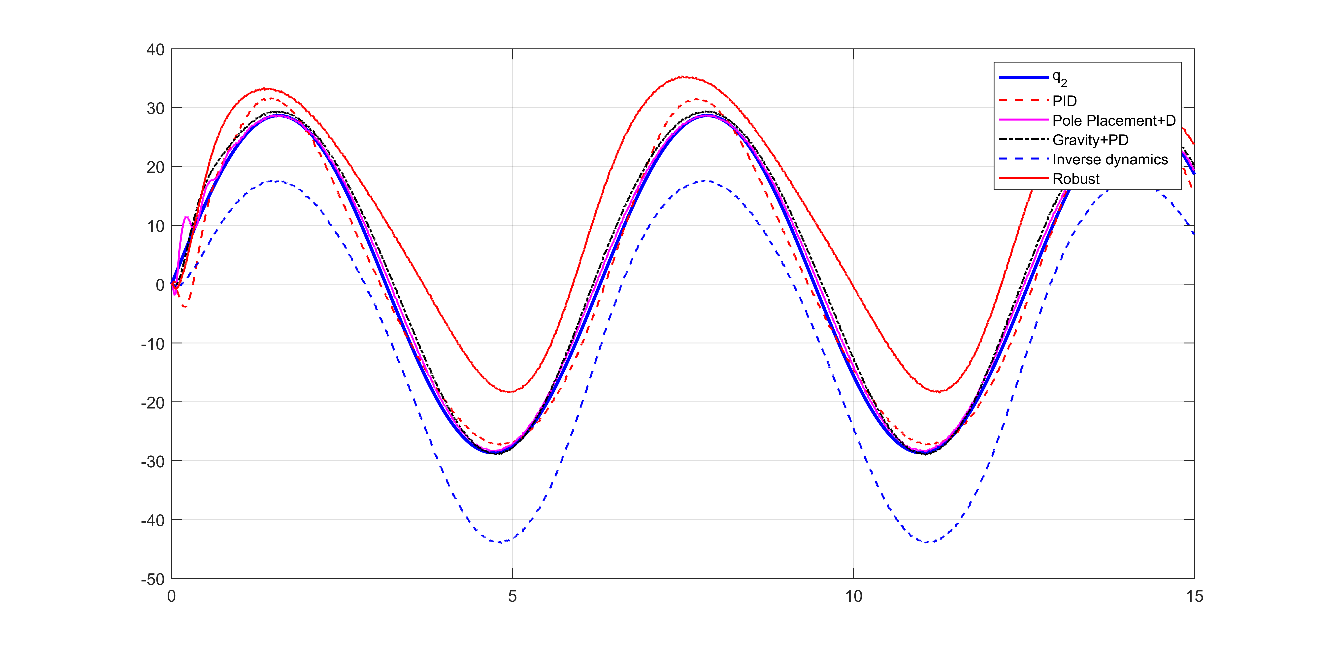


Figure 53: sine wave response comparison - joint position

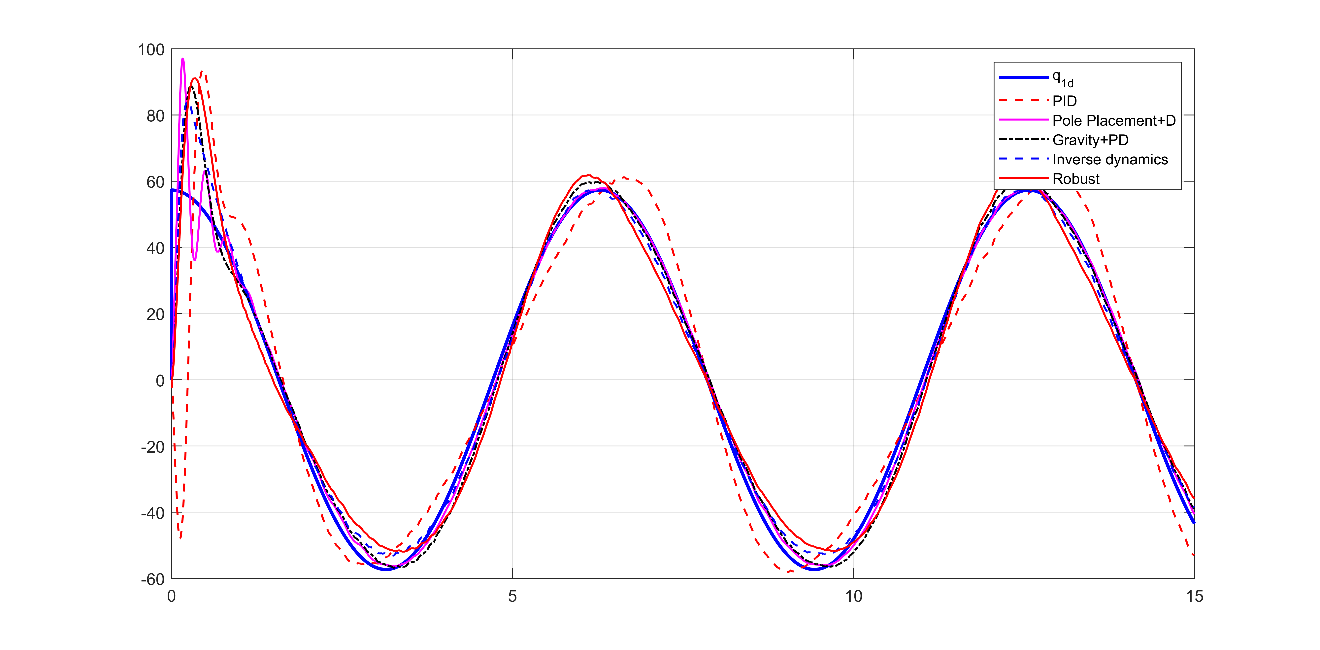




Figure 54: sine wave response comparison - joint velocity

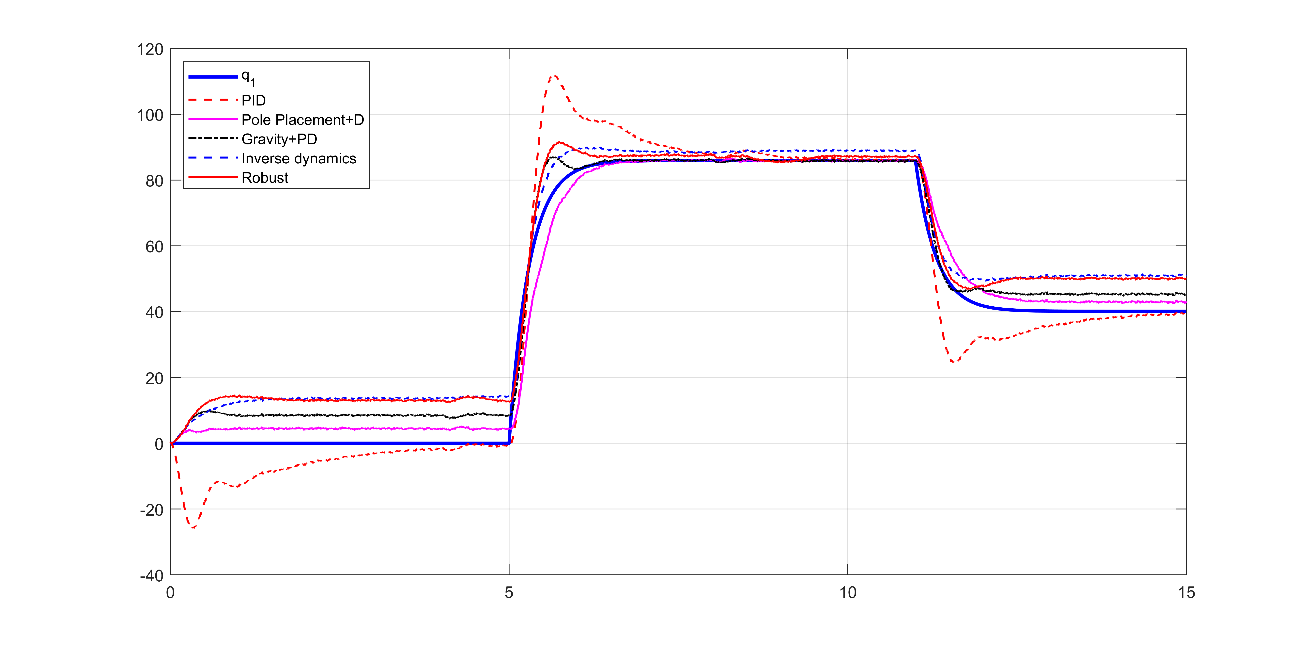
### Observations

* Steady-state error is present in all controllers except PID, however PID has slower settling time and higher overshoots
* Pole-Placement+D controller has the best tracking error followed by PID, gravity-PD control, robust control and inverse dynamics control, when parameters deviate from actual.
* Robust controller seems to perform better than inverse dynamics controller, though both are generally performing bad in both steady-state and tracking errors with poor dynamic parameters estimation.
* All controllers have zero constant velocity errors (step response) and large overshoots
* PID performs badly in velocity tracking (sine wave) compared to other controllers
* Velocity tracking of the first joint is better than the second joint
* Pole-Placement+D and robust controllers have the least velocity tracking errors for the first joint followed by gravity and inverse dynamics control. However, robust controller doesn’t perform well for the second joint.
* It is worth evaluating these controllers further under more dynamic parameter variations i.e. under load.
* LQR+D control produces similar results to Pole-Placement+D controller, hence it was omitted. LQR control is much easier to tune than Pole placement though.

# Simulation results – Comparison with parameter variation

* This section continues with realistic case and dynamic parameter deviation feeding to the controllers
* Additionally, the actual robot model is changed as follows:
  + Additional 25% mass to the links to simulate a load
  + Increase noise level by 100%.
* The controllers that are considered are:
  + PID control
  + Pole-Placement + Disturbance control
  + Gravity+PD control
  + Inverse dynamics control
  + Robust control

### **Step response**



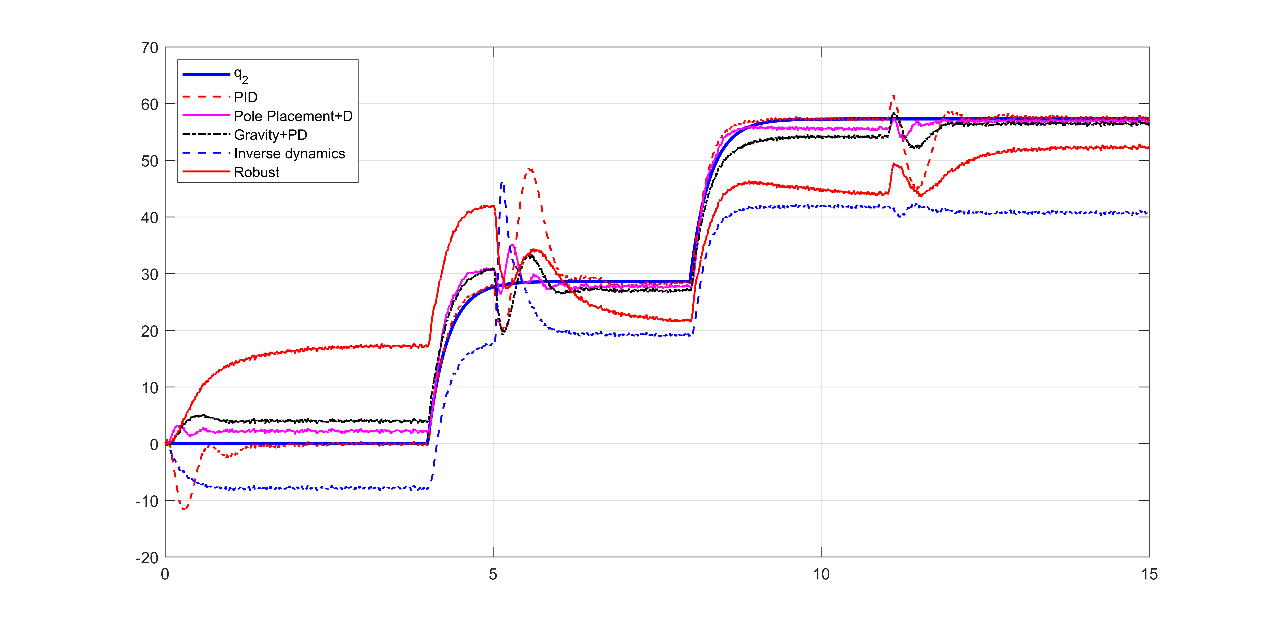
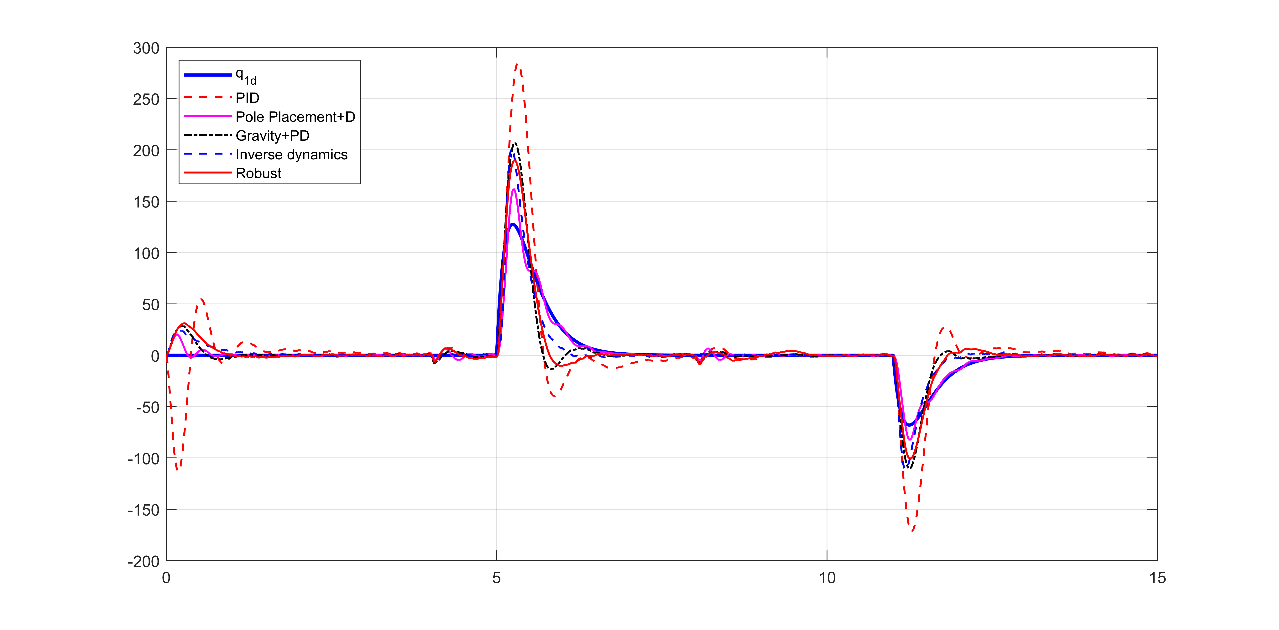


Figure 55: step response comparison - joint position



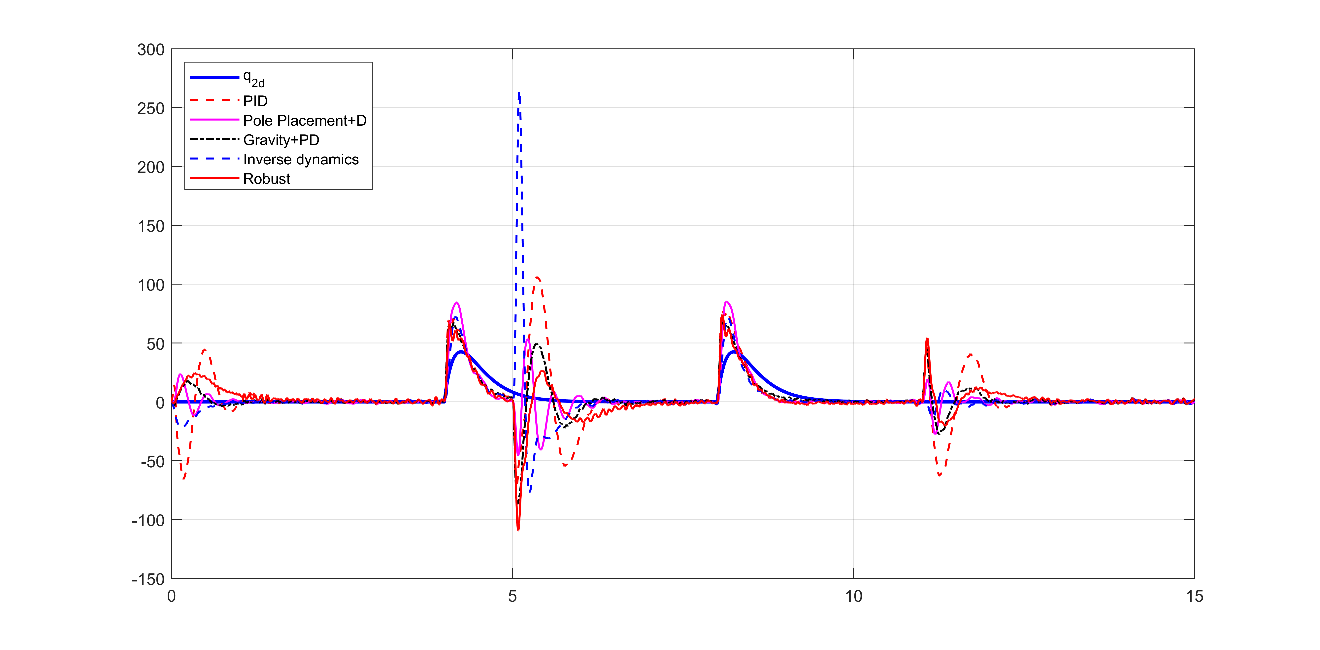


Figure 56: step response comparison - joint velocity

### **Sine wave response**

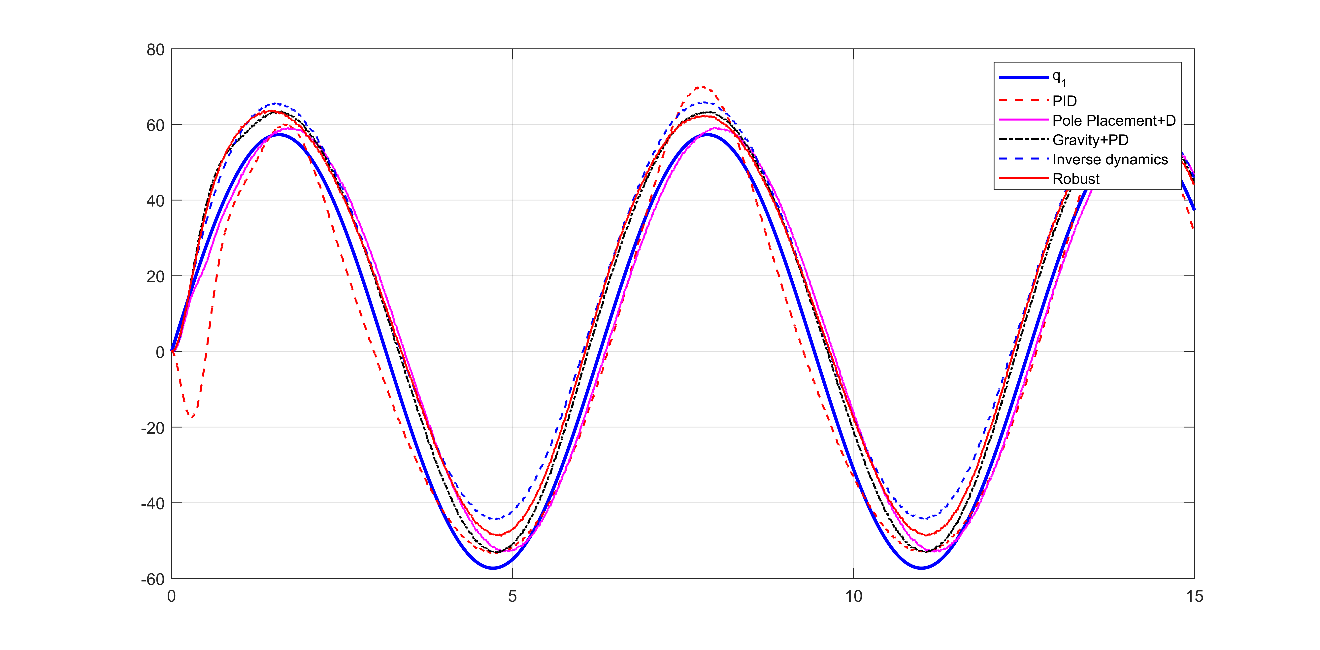
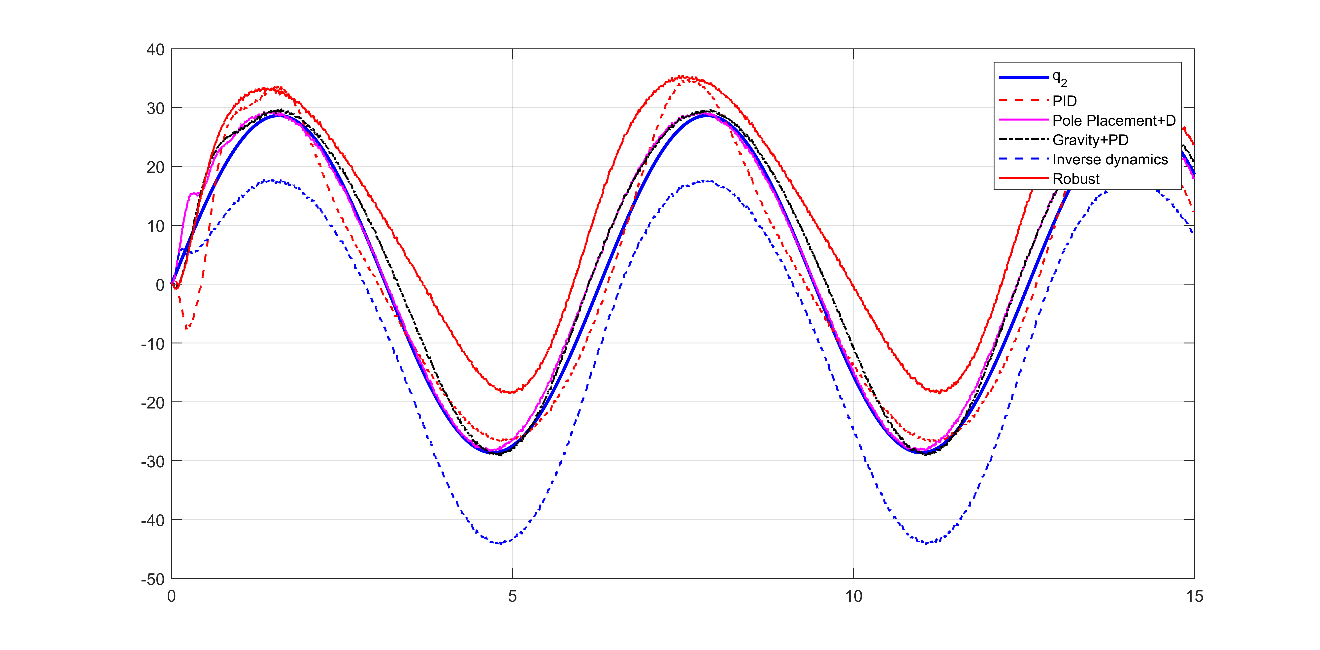
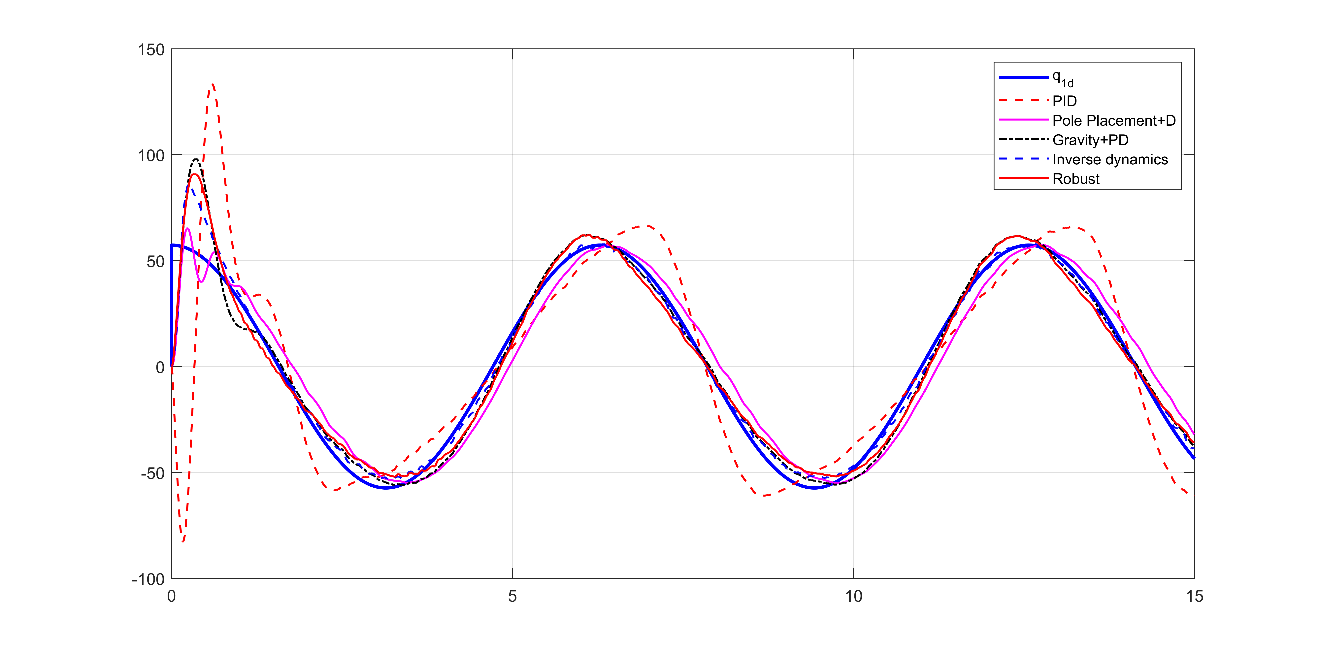


Figure 57: sine wave response comparison - joint position



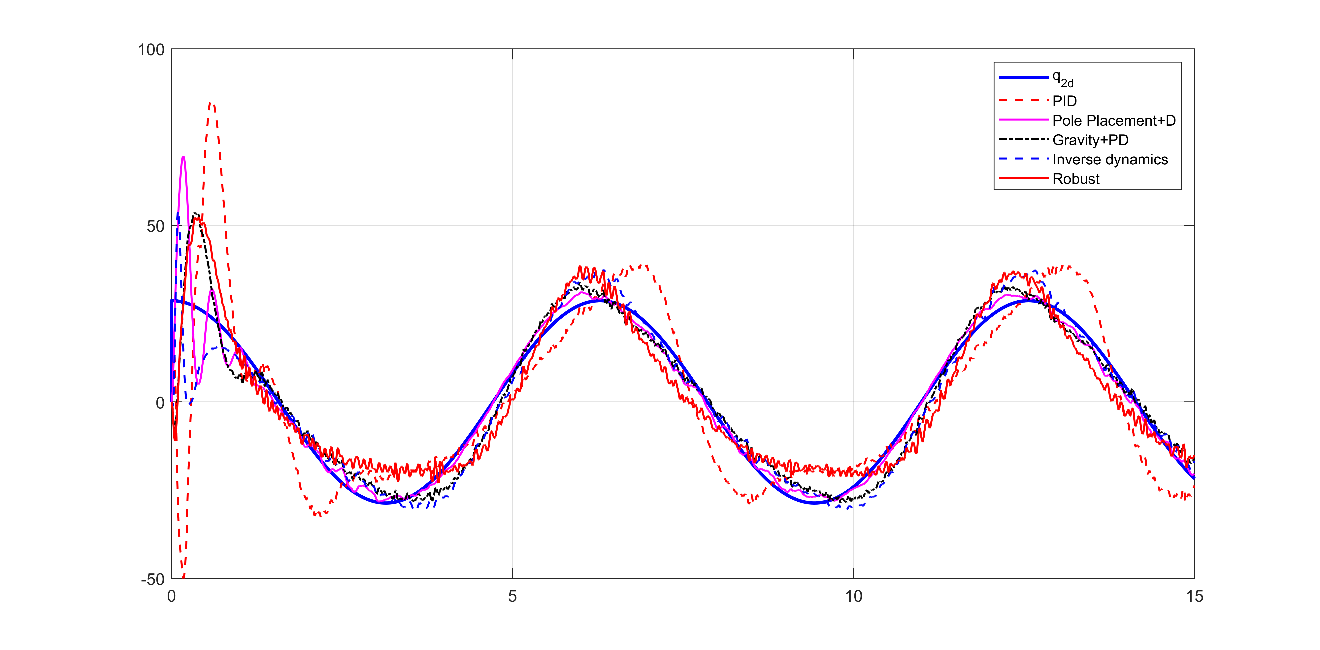


Figure 58: sine wave response comparison - joint velocity

### Observations

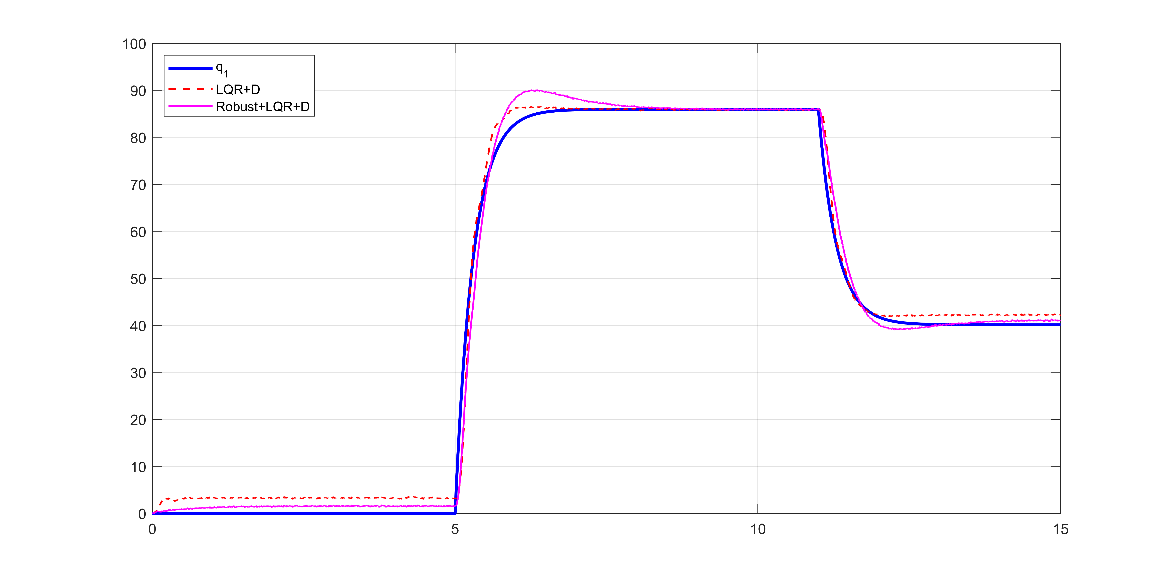
* No major performance changes in all controllers, there is some deterioration in performance though.
* All controllers still exhibit tracking and steady-state errors with varying degrees.
* Additional **I** term to the linear PD control part of the controllers (gravity+PD, inverse dynamics, robust) may reduce the error terms (tested in next section)

# Simulation results – Robust control

* This section focuses on the performance improvement when adding sliding-mode control.
* It continues with realistic case and dynamic parameter deviation feeding to the controllers
* The controllers that are considered are:
  + LQR+ Disturbance control
  + LQR+Disturbance + sliding mode term
* The controller parameters are

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Value | Parameter | Value |
|  | [100, 30, 1,1] |  | [10, 0.2] |
|  | [0.1, 0.2] |  | [1, 0.5] |
|  | [0.2, 1] |  |  |

### **Step response**



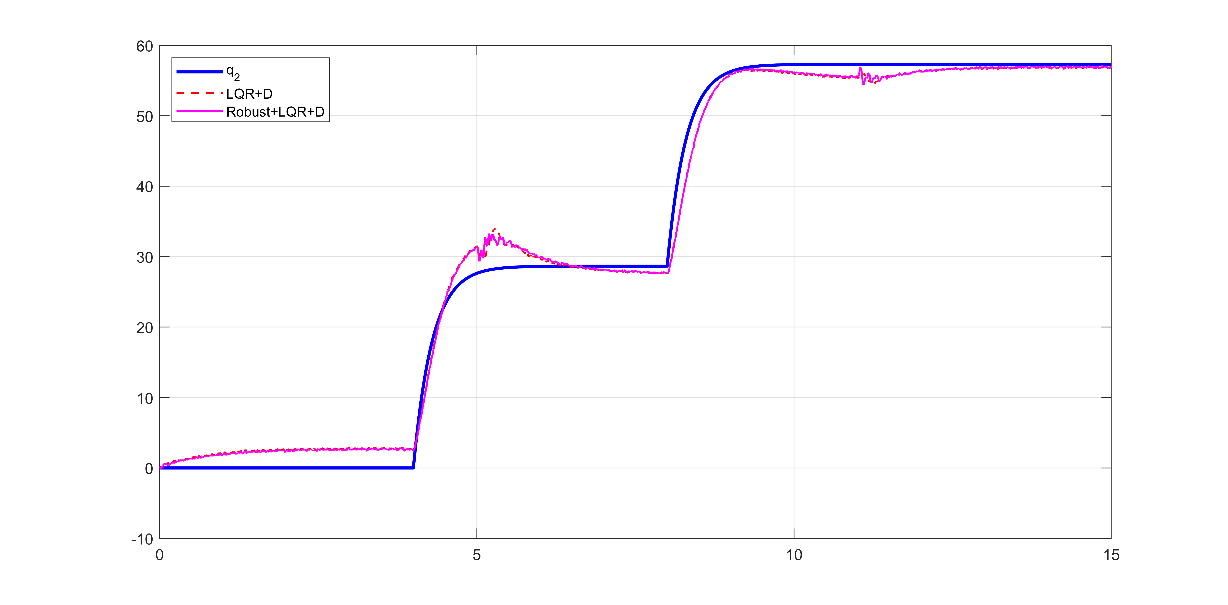
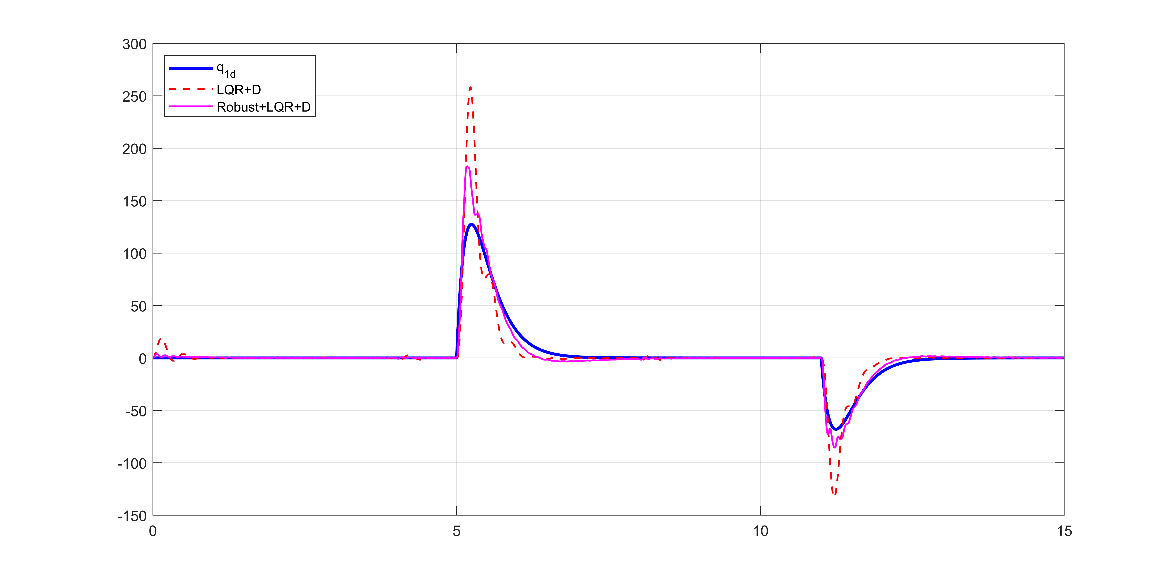


Figure 59: step response comparison - joint position



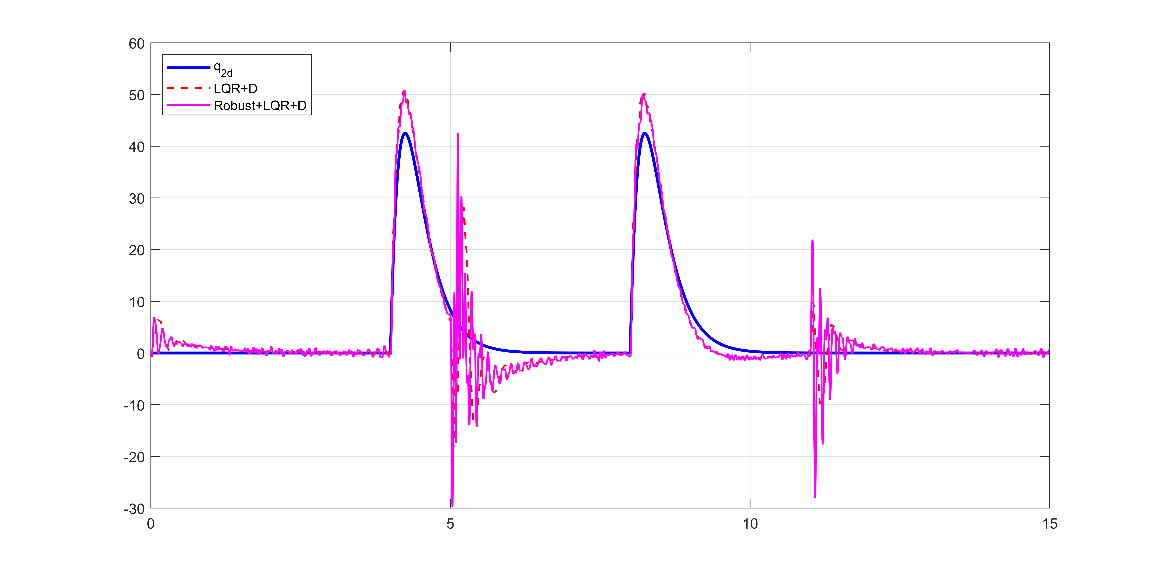
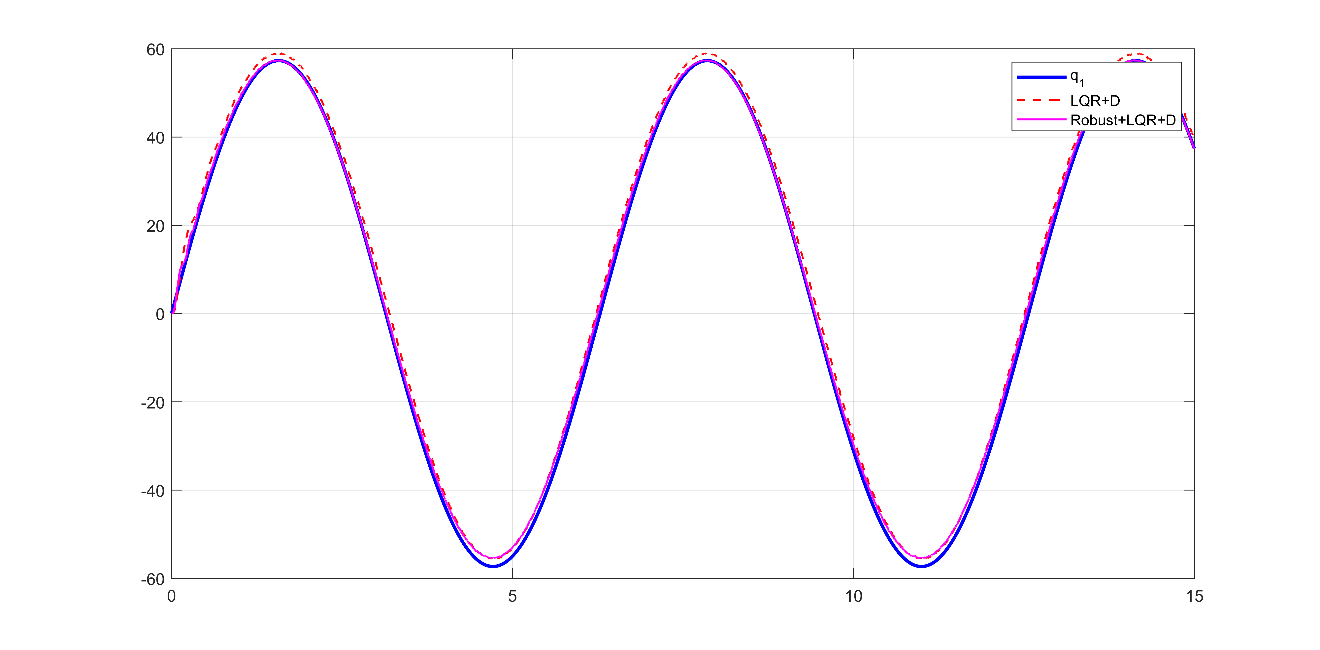


Figure 60: step response comparison - joint velocity

### **Sine wave response**



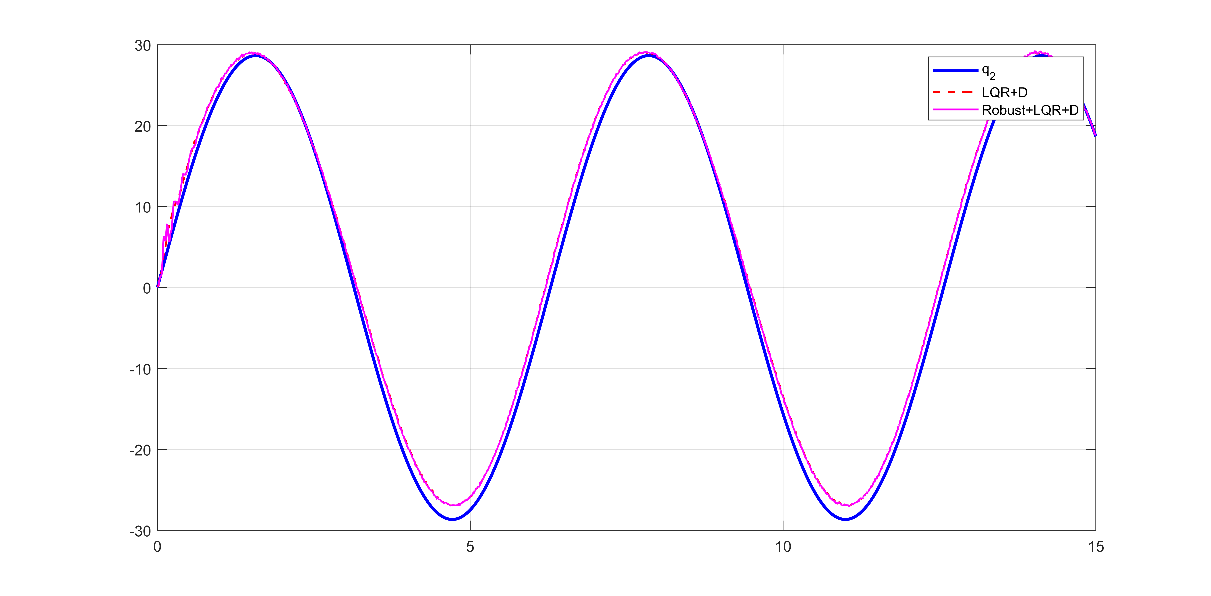


Figure 61: sine wave response comparison - joint position

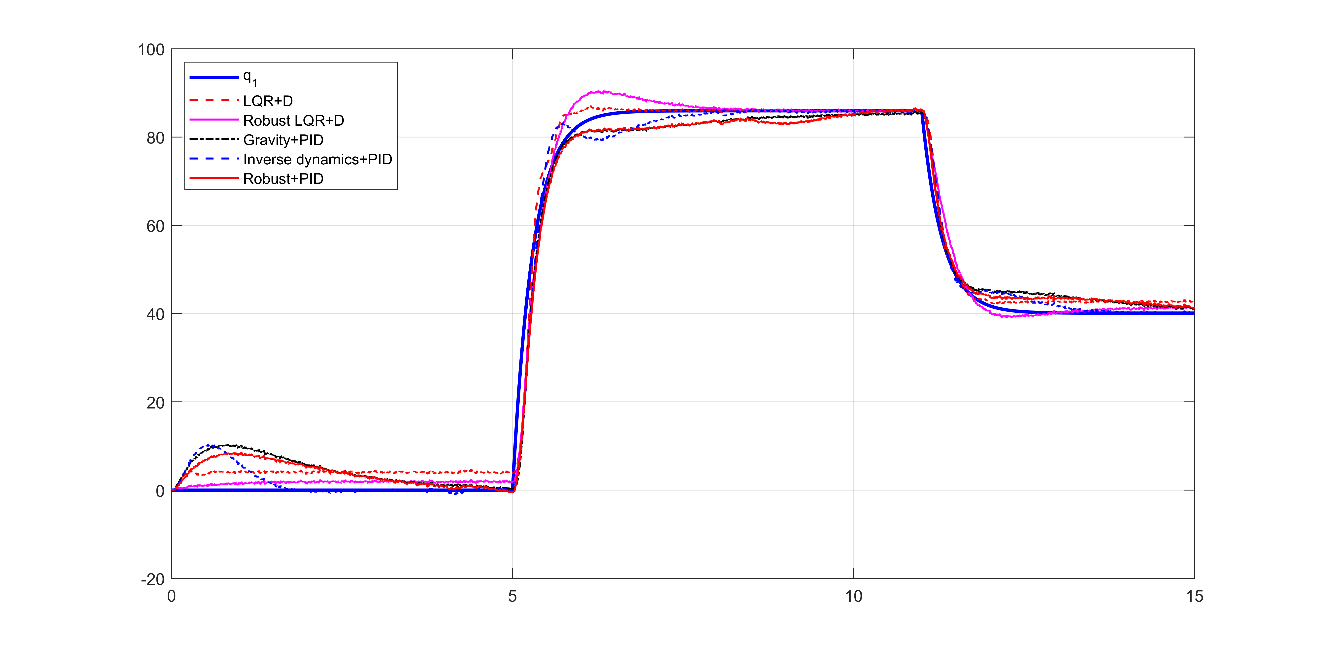
### Observations

* Steady-state and tracking error is improved when sliding-mode control term is added
* The improvement is clearer in joint 1 response due to the stronger sliding mode terms (higher gains). Increasing the gains for joint 2 produces bigger oscillations in the response.
* Sliding-mode term tends to increase overshoot and introduce some oscillations to the response.

# Simulation results – Improved controllers

* This section continues with realistic case and dynamic parameter deviation feeding to the controllers
* Additionally, the actual robot model is changed as follows:
  + Additional 25% mass to the links to simulate a load
  + Increase noise level by 100%.
* The controllers that are considered are:
  + Robust LQR + Disturbance control
  + Gravity+PID control: an I term is added and the controller is retuned
  + Inverse dynamics+PID control: an I term is added and the controller is retuned
  + Robust+PID control: an I term is added and the controller is retuned

### **Step response**



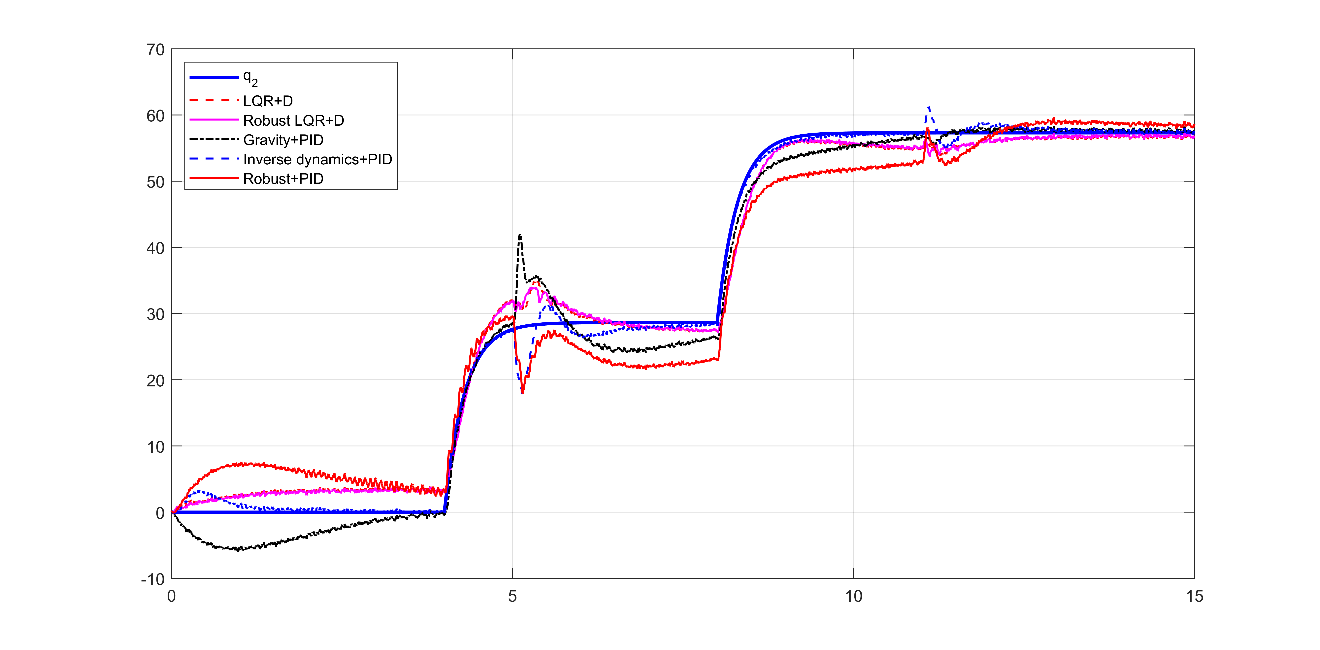
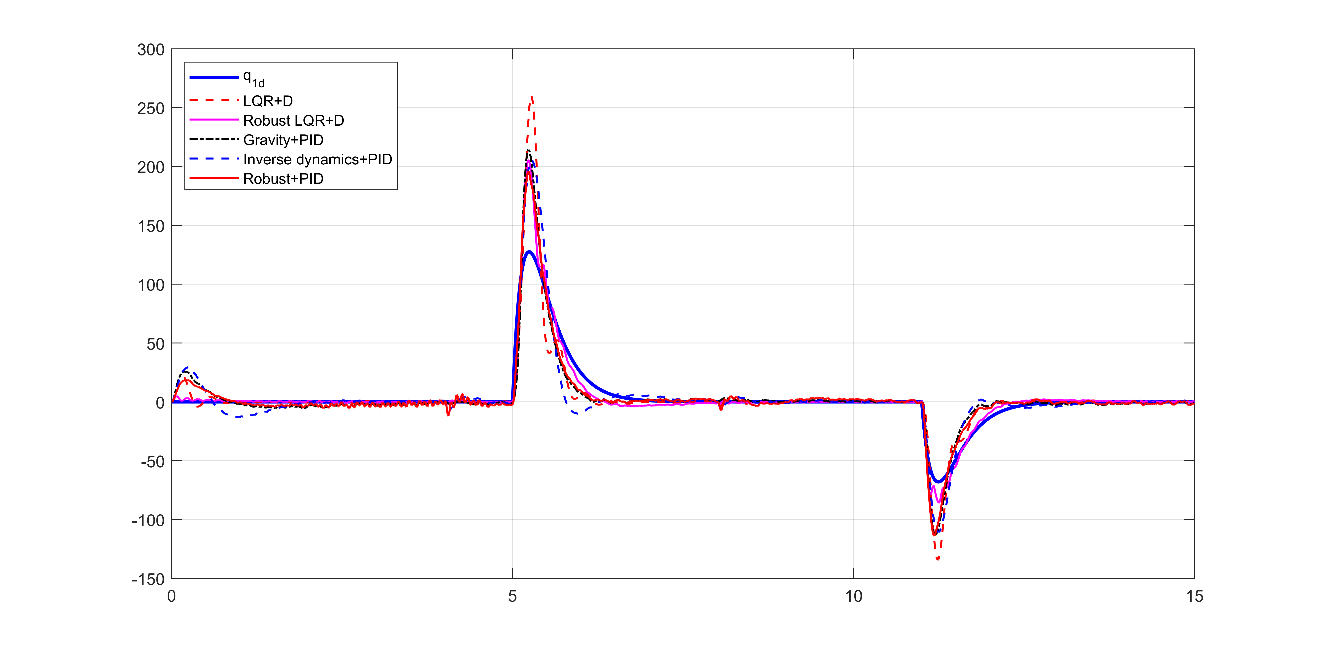


Figure 62: step response comparison - joint position



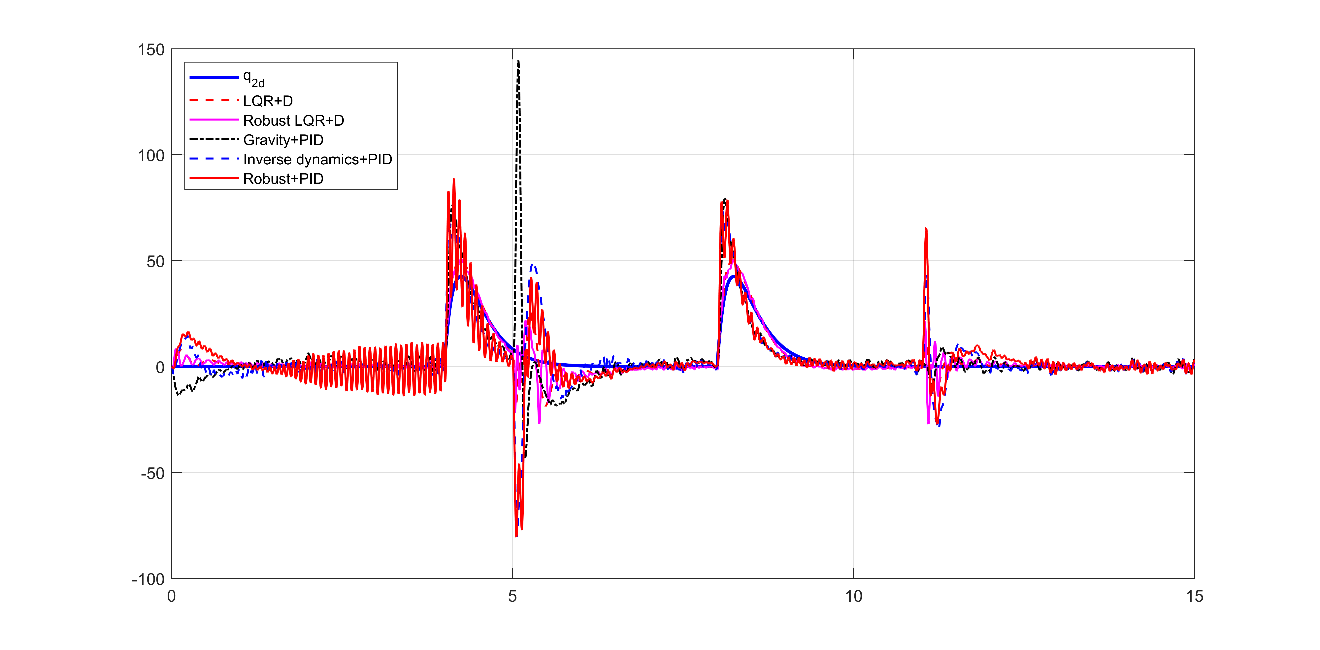
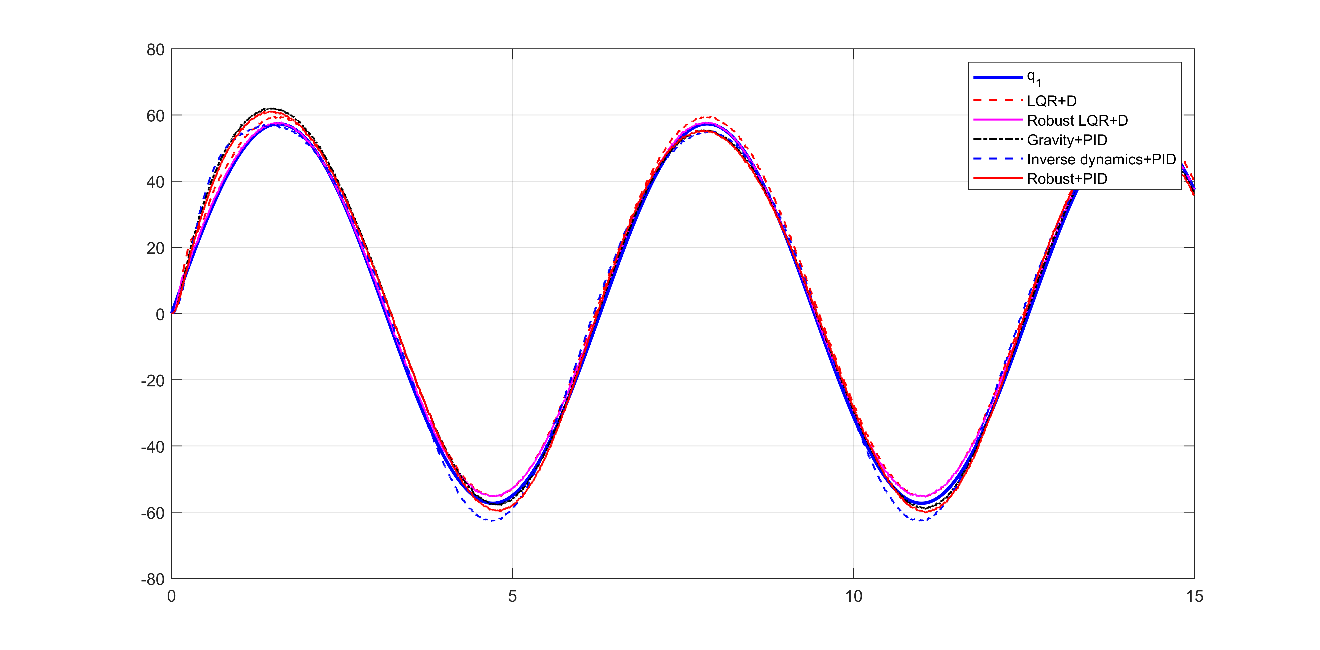
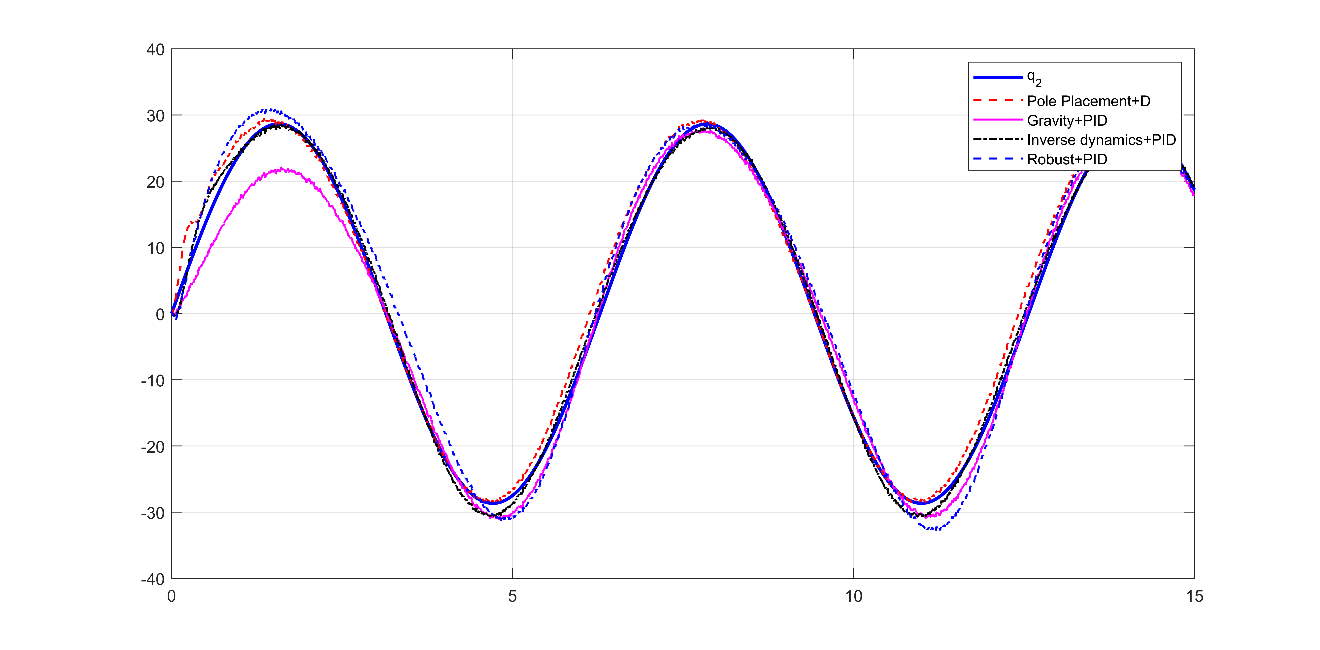


Figure 63: step response comparison - joint velocity

### **Sine wave response**





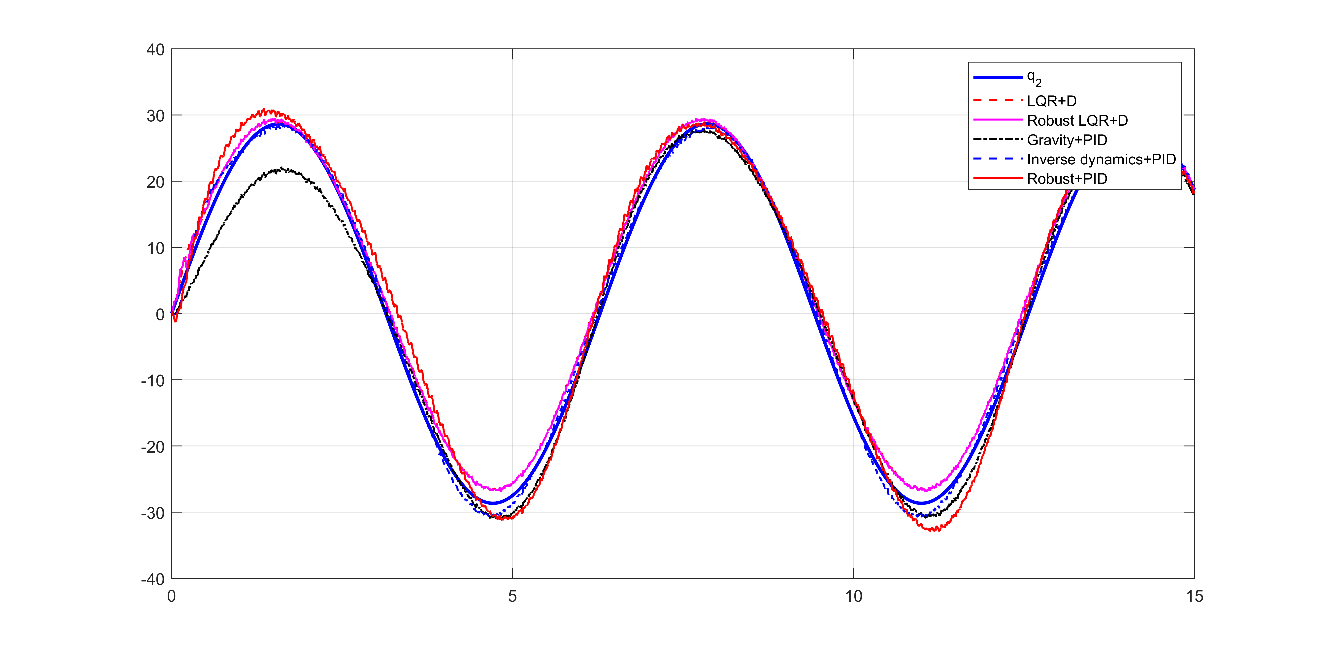
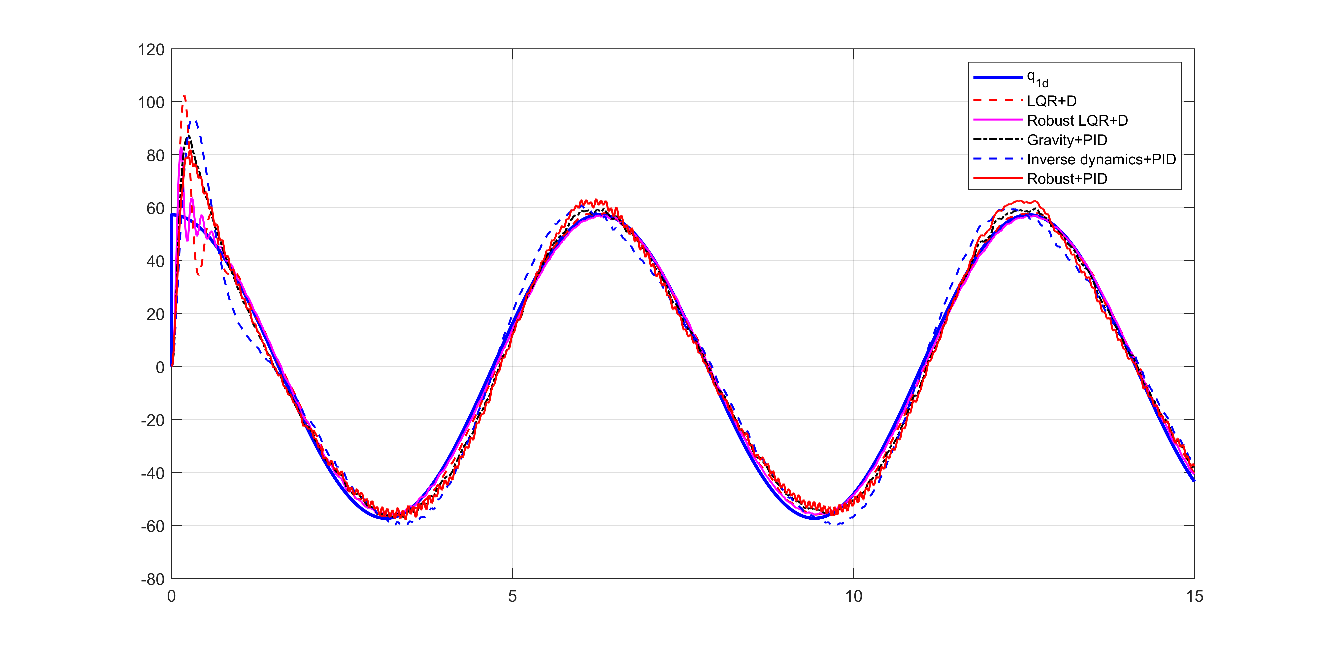


Figure 64: sine wave response comparison - joint position



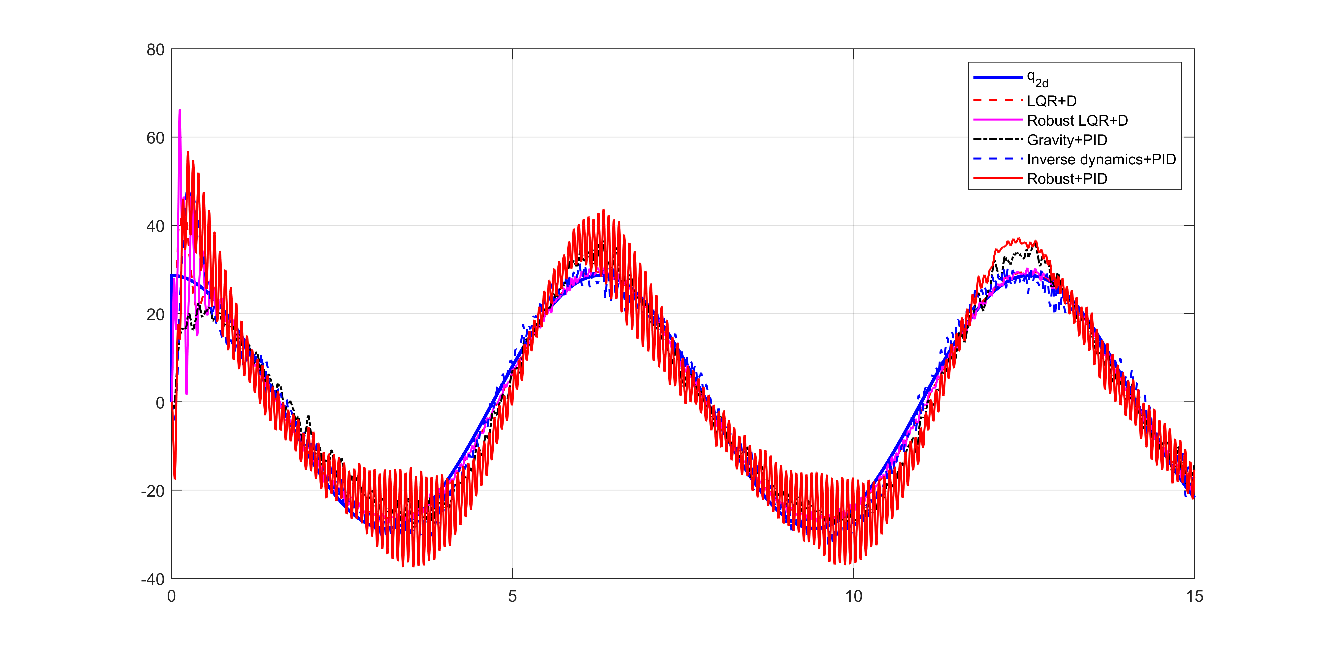


Figure 65: sine wave response comparison - joint velocity

### Observations

* Adding I term to the PID controller and retuning the gains eliminated the steady-state errors.
* Increasing the D gain of the PID amplifies the noise greatly and introduces oscillations in the response, however removing the D term will result in poor performance and large oscillations on the same PI gains.
* The sliding mode control implemented with LQR seems better than implemented with inverse dynamics (as per Siciliano’s book). The robust term implementation is quite different. Lhe one implemented with LQR (Sliding mode) is simpler, model-free and produce better results.

### Controller performance summary

Below is the performance summary from the simulations. The background color coding indicates general performance rating, the darker the green, the better the controller.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Controller** | **Steady-state error** | **Tracking error (position)** | **Overshoot and oscillation** | **Robustness** | **Velocity tracking** |
| **PD** | Large | Large | If gain is large or measurement is noisy | Robust to parameter changes (not dependent) | Very poor |
| **PID** | Zero if given enough time | Large (increases with trajectory frequency) | If gain is large or measurement is noisy.  Coupling effects from other joints | Robust to parameter changes (not dependent) | Poor |
| **Full state feedback** | Large | Large | Small | Robust to parameter changes (not dependent) | Poor |
| **Full state feedback + Disturbance** | Nonzero (dependent on DPEA\*) | Nonzero (dependent on DPEA\*) | Small | Not robust (depends on DPEA) | Good |
| **Inverse Dynamics** | - Dependent on DPEA\*  - I term will eliminate it | Good | Medium | Acceptable when I term is added | Good with PID control  (\*\*) |
| **Robust (Sliding mode)** | - Dependent on DPEA\*  - I term will eliminate it | Better than without robust term | Similar to controller without robust term | Good | Better than controller without robust term |
| **Gravity Compensated** | - Dependent on DPEA\*  - I term will eliminate it | Good | Large | Acceptable when I term is added | Good with PID control (\*\*) |
| \* DPEA stands for dynamic parameter estimation accuracy  \*\* Large velocity overshoot for step response | | | | | |

# References

* Trajectory Tracking Control via Decentralized Joint-Level Schemes <https://github.com/RickyMexx/ttc-decentralized>
* Robotics Modelling, Planning and Control by Bruno Siciliano, Lorenzo Sciavicco, Luigi Villani, and Giuseppe Oriolo