Electrical Drives and Electrical Vehicles

Viena Car Simulation

Bernardo Matos 84017
Alejandro Saez 95017
Jamilson Junior 87025
André Gomes 71166

Vienna Project with Speed Regulator

Table of Contents

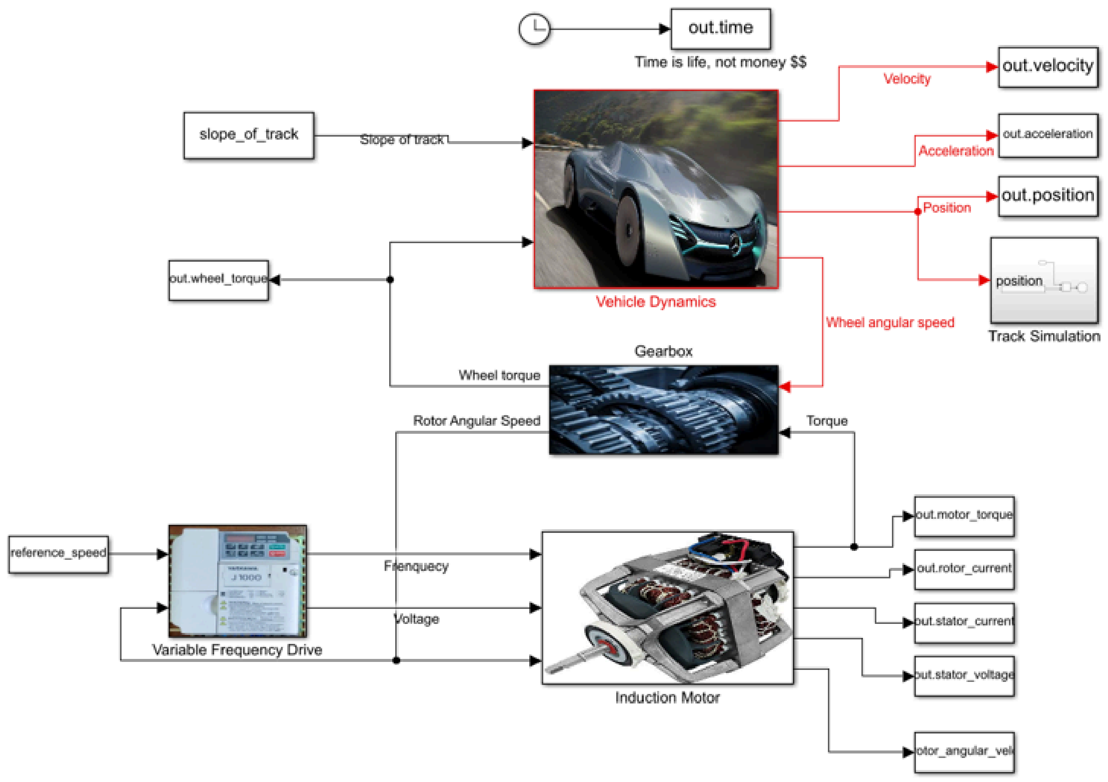
New Variable Speed Drive 1
Theretical approach of PID Control 2
Implementation 3
Proportional Control 3
Proportional Control Track1 3
Proportional Control, Reference Speed, Track1: 3
Propotional Control, Voltage and frequency, Track1: 5
Propotional Control, Stator Current, Track2 6
Propotional Control, Motor and Wheel Torque, Track1 7
Proportional Control track2 8
Propotional Control, Reference Angular Speed, Track2 8
Propotional Control, Voltage and Frequency, Track2 10
Propotional Control, Stator Current, Track2 11
Propotional Control, Motor and Wheel Torque, Track2 12
Proportinal Integral Control 13
Proportional Integral Control Track1 13
Proportional Integral Control, Reference Speed, Track1 14
Proportional Integral Control, Voltage and Frequency, Track1 15
Proportional Integral Control, Stator Current, Track1 16
Proportional Integral Control, Motor and Wheel Torque, Track1 17
Proportional Integral Control track2 18
Proportional Integral Control, Reference Speed, Track2 18
Proportional Integral Control, Voltage and Frequency, Track2 20
Proportional Integral Control, Stator Current, Track2 21
Proportional Integral Control, Motor and Wheel Torque, Track2 22
Conclusions 23

New Variable Speed Drive

In this part of the vienna car project, the objective is to build upon the previous variable speed drive.

```
%The block diagram of the whole system is:

open_system('vienna_car_part3');
```



The objective of the variable frequency drive is to provide the induction motor with a stator voltage and frequency in order for the car to run. The variable frequency drive has to follow a reference value, in this case, given by the rotational velocity of the wheels developed in the first part of the project. In order to follow the reference well, a PID control was added as shown in the following figure.

Theretical approach of PID Control

PID control stands for Proportional Integral Derivative control, in which the objective of having in the output a signal follows a reference given in the input. Each part of the PID has the following effect:

- Proportional- It is the biggest driver of the output to the reference, that is, the bigger the proportional constant, the faster the output will reach the reference. However, if the proportional component is too high, the output will overshoot the reference, representing the momentum of the system. trying to balance the previous effect, one might think that setting a proportional constant low will reduce overshoot and instability of the system, but if this constant is too low, the output will never reach the reference in real time.

- Differential - The differential component is responsible for measuring the rate of which the output is approaching the reference. The differential control then contributes with values that are contrary to the proportional control, mitigating the overshoot and allowing for higher response speeds. The downside is that the differential controller is sensitive to noise, and for an electric motor that has a very quick response, it may cause a lot of instability.

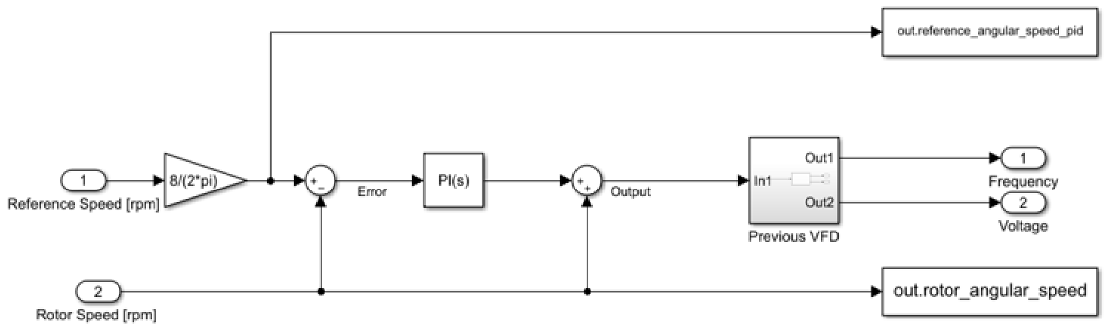
- Integral - The integral control has the effect of reducing the error between the output and the reference when the system is stabilising.

Implementation

The previous VFD is now complemented with the PID control:

1. It receives the reference speed (wheel speeds of the first part of the project), multiples it by 8 to create the desired motor speed.
2. It calculates the error between the reference speed and the actual speed of the motor.
3. The error enters in a ready made PID block from simulink.
4. The control signal of the PID block is then added with the same actual speed of the motors, resulting in the signal that will give the V/f values the exact same way as calculated in the second part of the project.

```
clc
clear
load('./sim_part1.mat');
close all
open_system('vienna_car_part3/Variable Frequency Drive/Variabale
 Frequency Drive Model');
```



Proportional Control

Now, it is considered only the proportional component of the controller. The proportional constant has a value of:

```
proportinal_constant = 10;
integral_constant = 0;
```

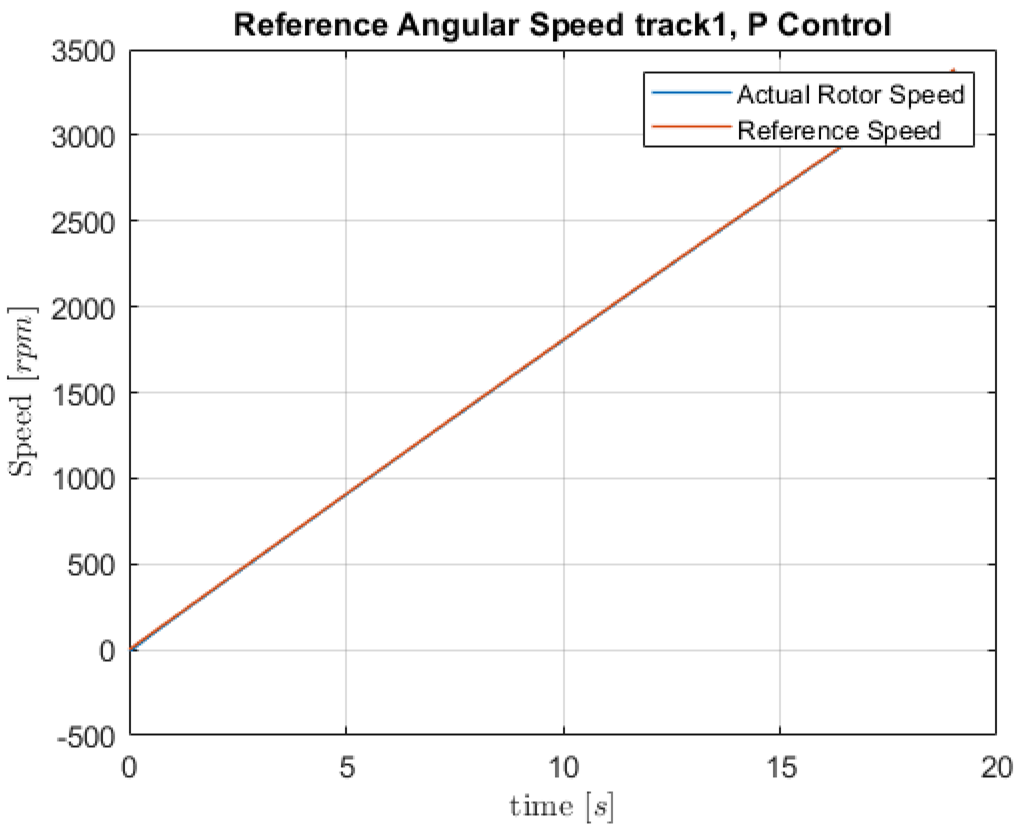
Proportional Control Track1

```
slope_of_track = 3;
reference_speed = reference_speed_track1;
simulation_length_track = distance_A_B;
track1_part3_p = sim('vienna_car_part3');
```

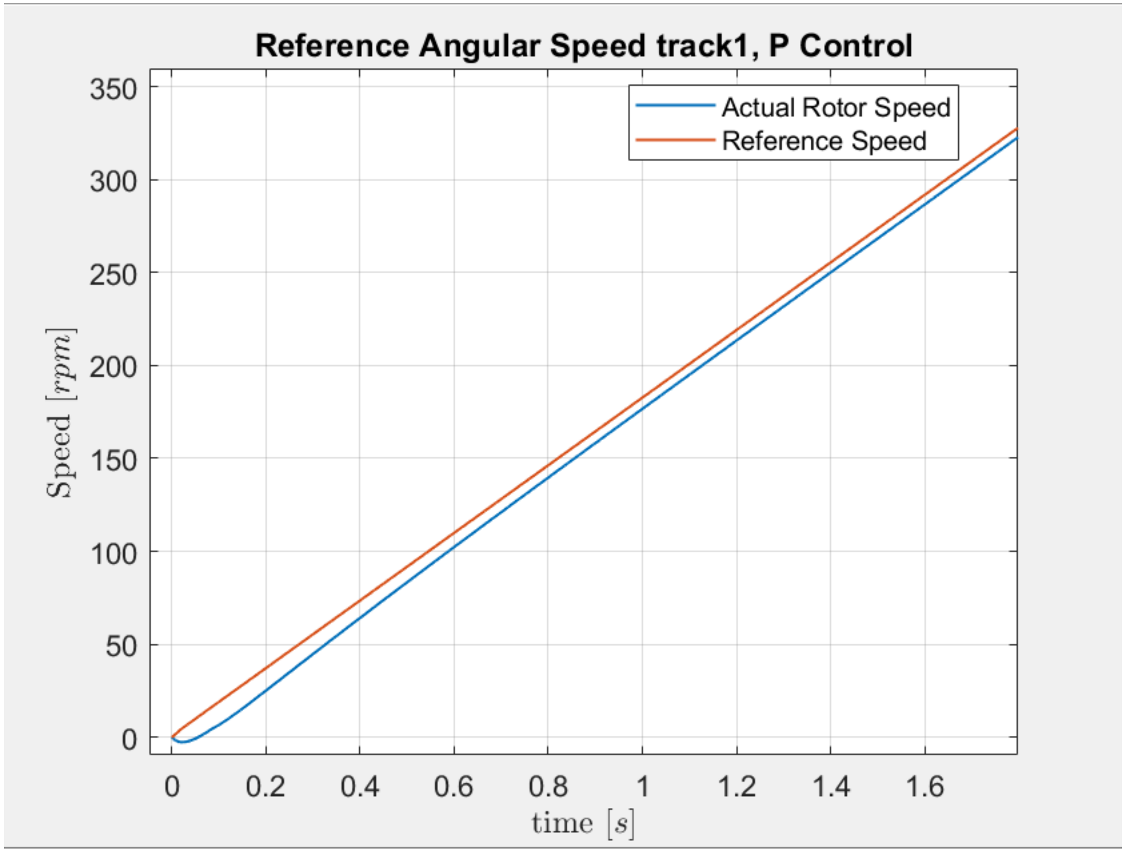
Proportional Control, Reference Speed, Track1:

The reference angular speed and speed and the output angular speed of the motor are presented below:

```
figure()
plot(track1_part3_p.time,
 track1_part3_p.rotor_angular_speed,'LineWidth', 1);
hold all
plot(track1_part3_p.time,
 track1_part3_p.reference_angular_speed_pid, 'LineWidth', 1);
set( gca, 'FontSize', 11);
grid on;
title('Reference Angular Speed track1, P Control');
xlabel('time $[s]$','Interpreter', 'latex');
ylabel('Speed $[rpm]$','Interpreter', 'latex');
legend('Actual Rotor Speed','Reference Speed');
```



One can see that the proportional controller follows the reference, being a good candidate for a controller of this vehicle. However, as we can see in the figure below, this controller has a steady state error:

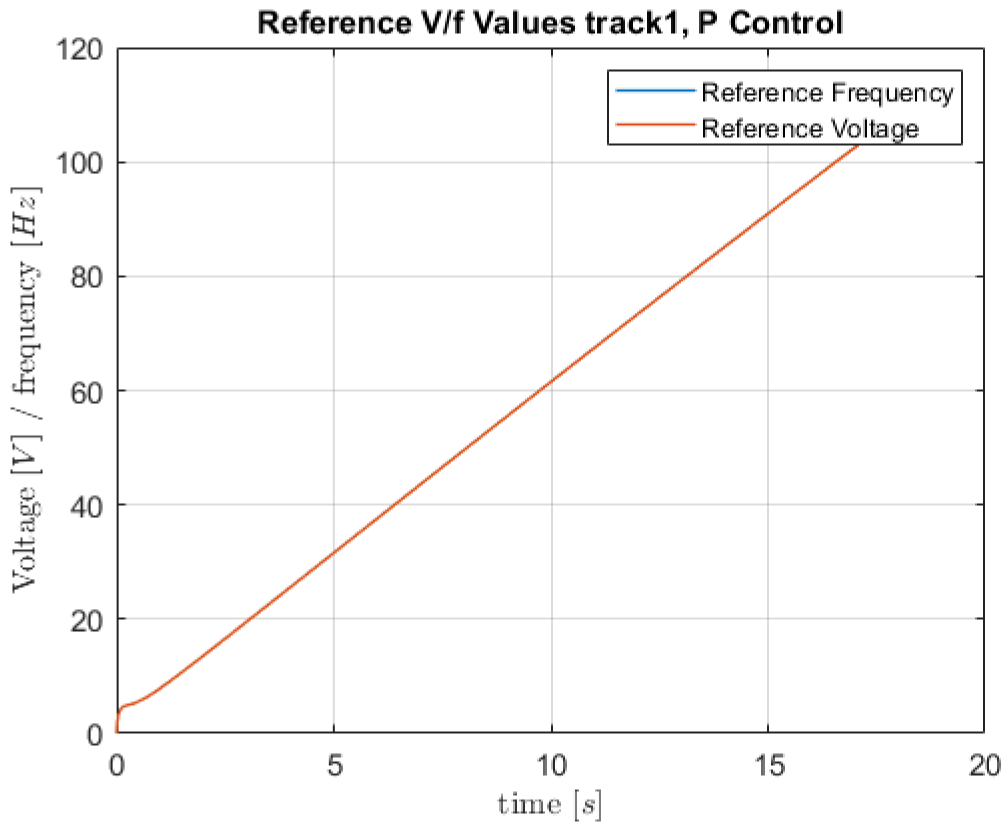


This steady state error exists because the proportional control is too small to make the output signal reach the reference in finite time.

Propotional Control, Voltage and frequency, Track1:

The output V/f values of the VFD block are:

```
figure()
plot(track1_part3_p.time,
 track1_part3_p.reference_frequency, 'LineWidth', 1);
hold all
plot(track1_part3_p.time,
 track1_part3_p.reference_voltage, 'LineWidth', 1);
set( gca, 'FontSize', 11);
grid on;
title('Reference V/f Values track1, P Control');
xlabel('time $[s]$','Interpreter', 'latex');
ylabel('Voltage $[V]$ / frequency $[Hz]$','Interpreter', 'latex');
legend('Reference Frequency', 'Reference Voltage');
```



The maximum V/f = 1 value is:

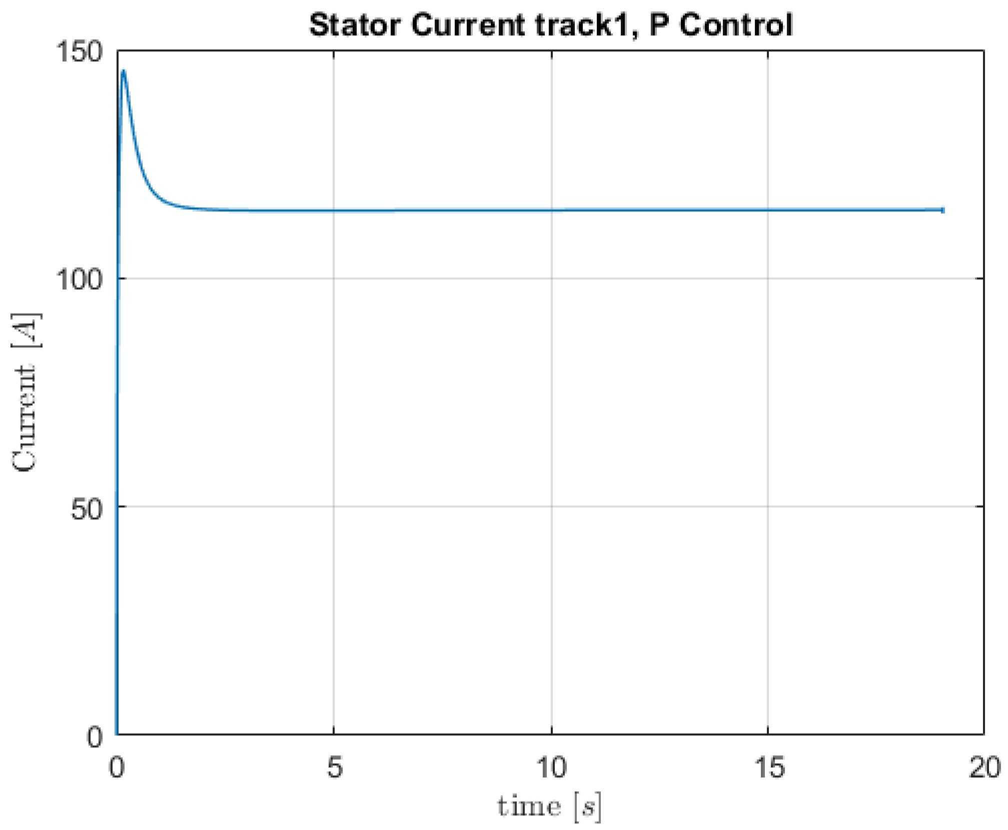
```
fprintf('%d [V]/[Hz]',max(track1_part3_p.reference_voltage));
```

```
1.141064e+02 [V]/[Hz]
```

Propotional Control, Stator Current, Track2

The previous V/f values result in the following stator current:

```
figure()
plot(track1_part3_p.time,
 abs(track1_part3_p.stator_current), 'LineWidth', 1);
set( gca, 'FontSize', 11);
grid on;
title('Stator Current track1, P Control');
xlabel('time $[s]$','Interpreter', 'latex');
ylabel('Current $[A]$','Interpreter', 'latex');
```



For a proportional gain of 10, the current stabilizes in around 115 [A]. The maximum current value is:

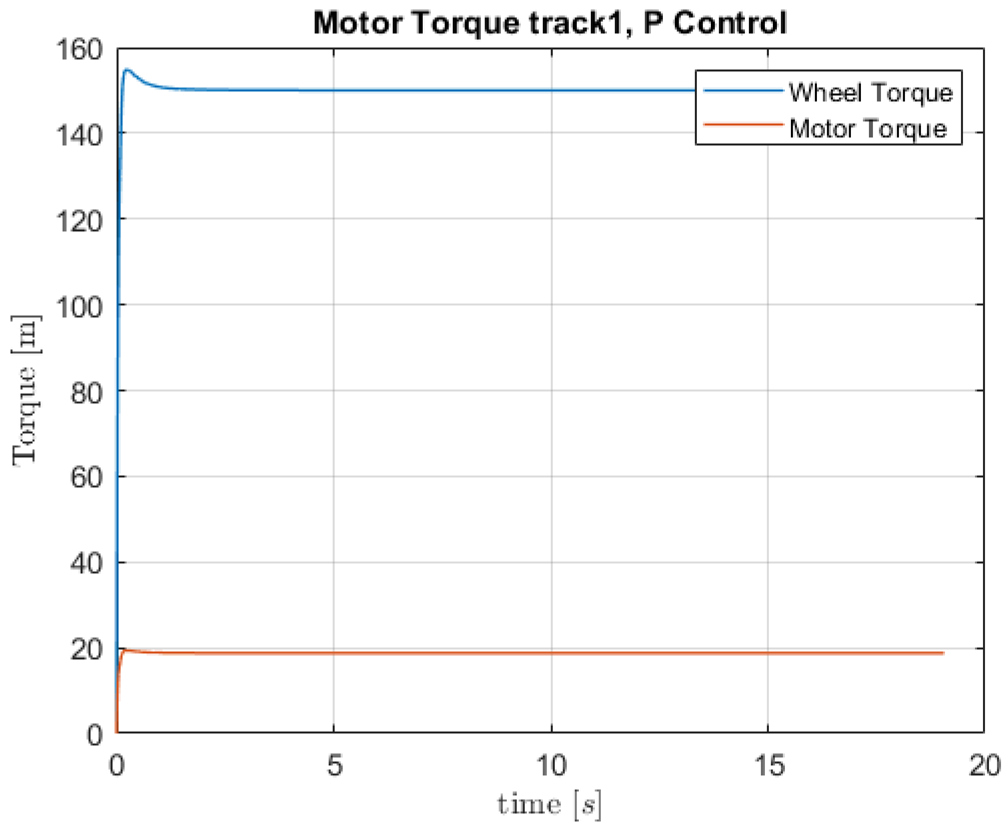
```
fprintf('%d [A]',max(abs(track1_part3_p.stator_current)));
```

```
1.454748e+02 [A]
```

Propotional Control, Motor and Wheel Torque, Track1

The resulting motor and wheel torque are given by:

```
figure()
plot(track1_part3_p.time, track1_part3_p.wheel_torque, 'LineWidth',
 1);
hold all;
plot(track1_part3_p.time, track1_part3_p.motor_torque, 'LineWidth',
 1);
set( gca, 'FontSize', 11);
grid on;
title('Motor Torque track1, P Control');
xlabel('time $[s]$','Interpreter', 'latex');
ylabel('Torque [m]','Interpreter', 'latex');
legend('Wheel Torque', 'Motor Torque');
```



One can see that, as expected, the value of the mortar torque stabilizes in 150 [Nm]. The maximum initial torque is:

```
fprintf('%d [A]',max(track1_part3_p.wheel_torque));
```

```
1.548644e+02 [A]
```

Proportional Control track2

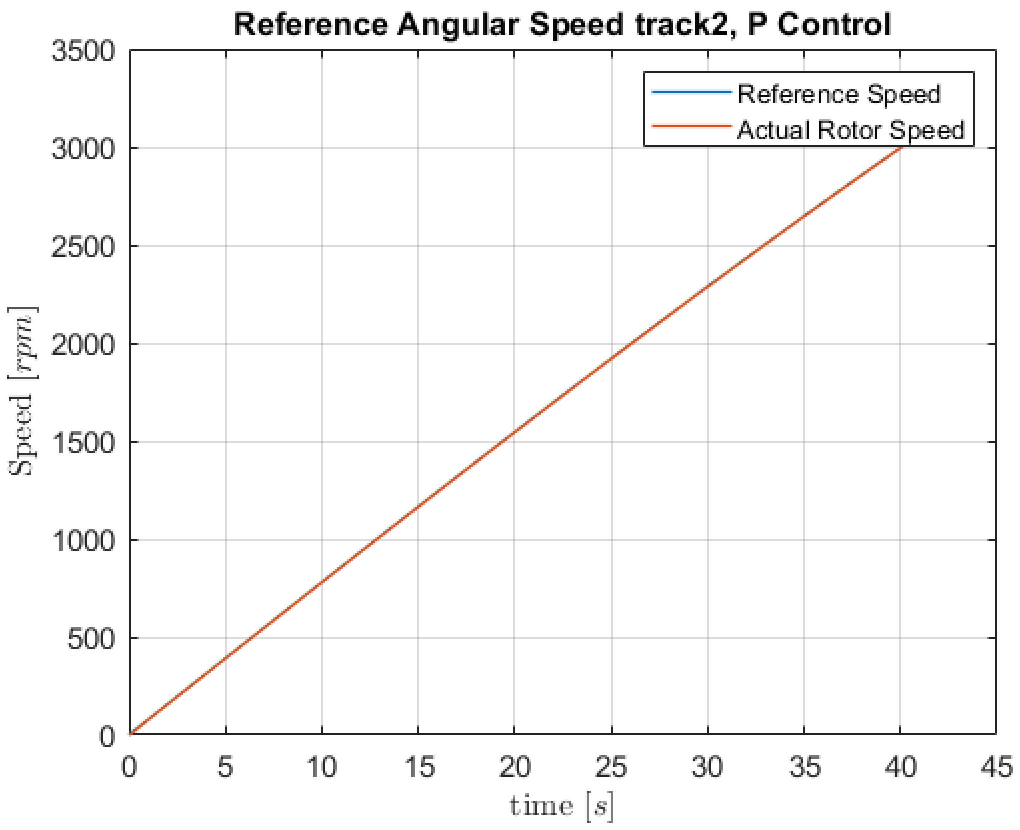
```
slope_of_track = 0;
reference_speed = reference_speed_track2;
simulation_length_track = track2_length;
track2_part3_p = sim('vienna_car_part3');
```

Propotional Control, Reference Angular Speed, Track2

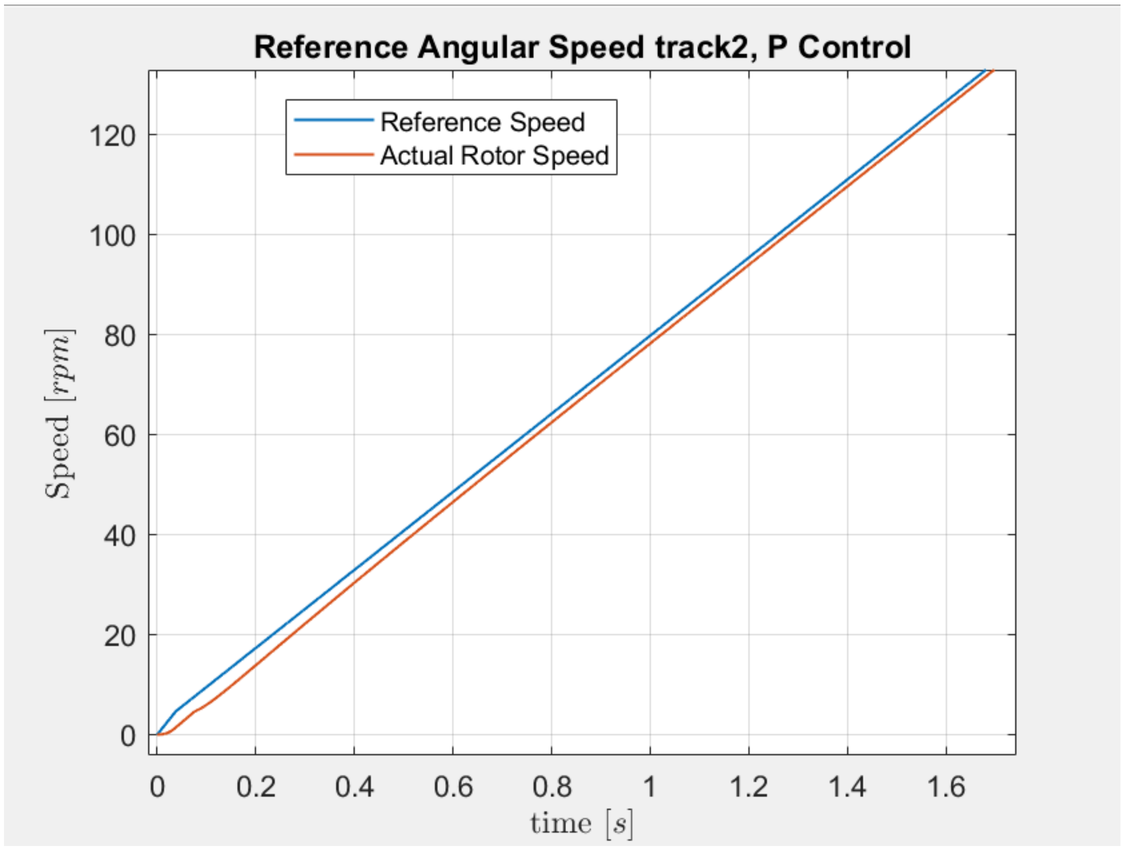
```
The reference angular speed and speed and the output angular speed of the motor ar
```

```
figure()
plot(track2_part3_p.time,
 track2_part3_p.reference_angular_speed_pid, 'LineWidth', 1);
hold all
plot(track2_part3_p.time,
 track2_part3_p.rotor_angular_speed, 'LineWidth', 1);
```

```
set( gca, 'FontSize', 11);
grid on;
title('Reference Angular Speed track2, P Control');
xlabel('time $[s]$','Interpreter', 'latex');
ylabel('Speed $[rpm]$','Interpreter', 'latex');
legend('Reference Speed', 'Actual Rotor Speed');
```



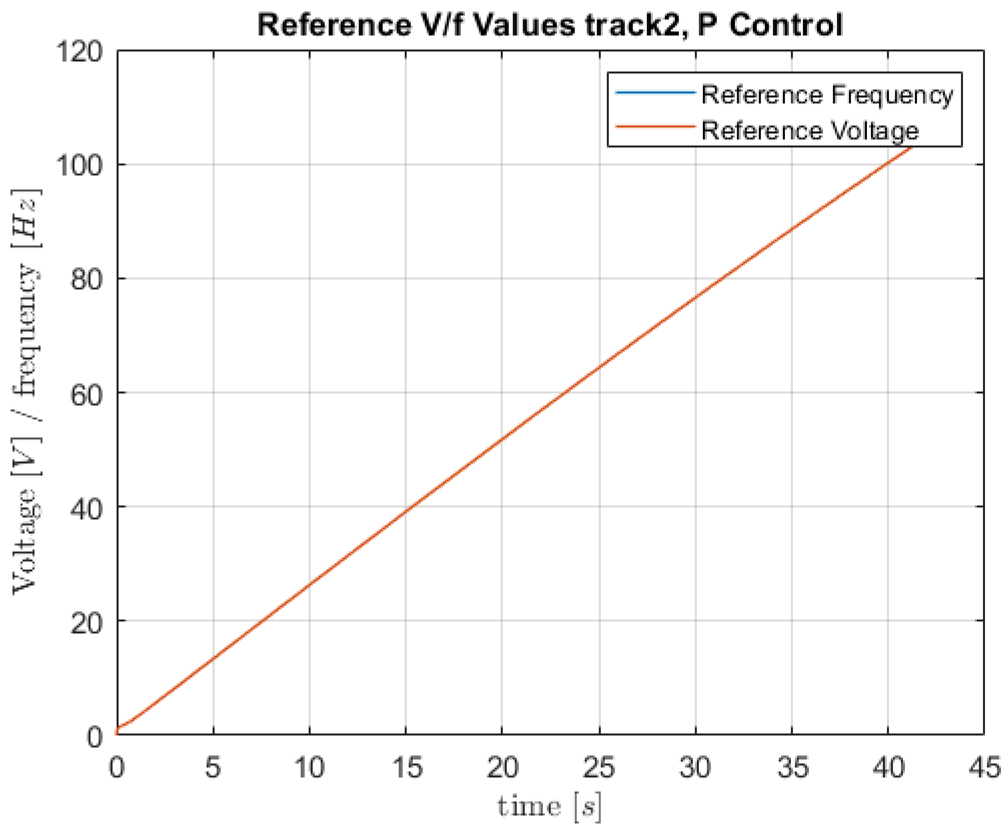
Once again the, reference angular speed and the angular speed of the motor are acceptable, but one can see that a steady state error still exists:



Propotional Control, Voltage and Frequency, Track2

The output V/f values of the VFD block are:

```
figure()
plot(track2_part3_p.time,
 track2_part3_p.reference_frequency, 'LineWidth', 1);
hold all
plot(track2_part3_p.time,
 track2_part3_p.reference_voltage, 'LineWidth', 1);
set( gca, 'FontSize', 11);
grid on;
title('Reference V/f Values track2, P Control');
xlabel('time $[s]$','Interpreter', 'latex');
ylabel('Voltage $[V]$ / frequency $[Hz]$','Interpreter', 'latex');
legend('Reference Frequency', 'Reference Voltage');
```



The maximum V/f = 1 value is:

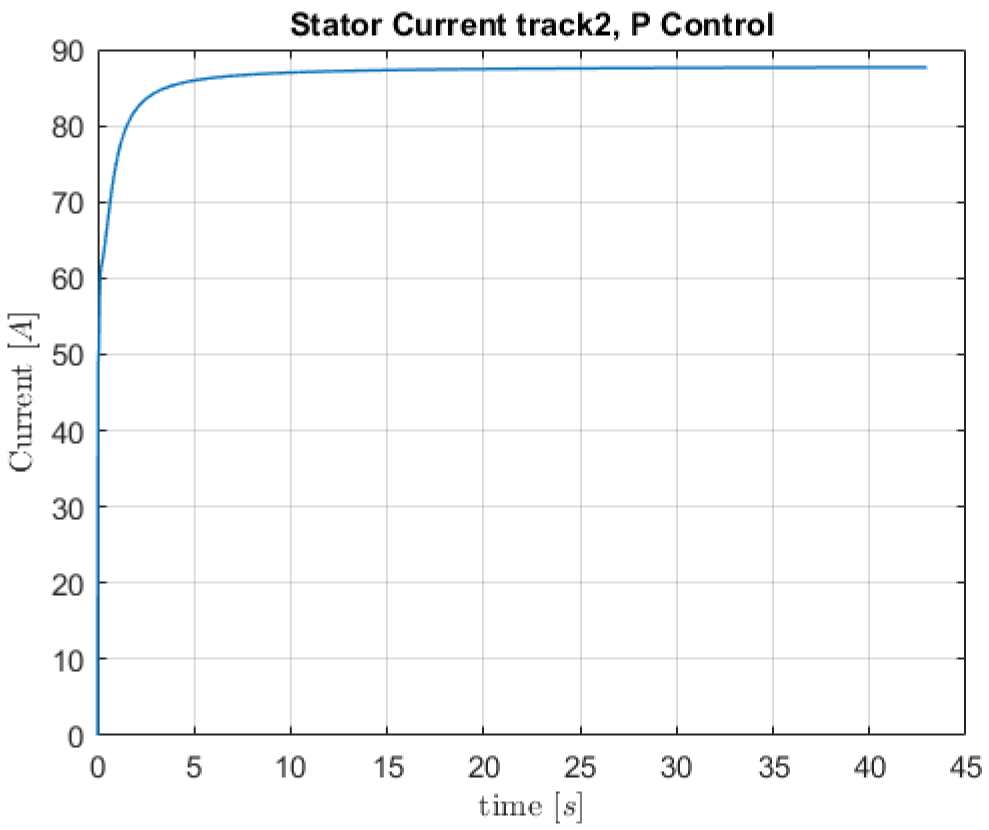
```
fprintf('%d [V]/[Hz]',max(track2_part3_p.reference_voltage));
```

```
1.069298e+02 [V]/[Hz]
```

Propotional Control, Stator Current, Track2

The previous V/f values result in the following stator current:

```
figure()
plot(track2_part3_p.time,
 abs(track2_part3_p.stator_current), 'LineWidth', 1);
set( gca, 'FontSize', 11);
grid on;
title('Stator Current track2, P Control');
xlabel('time $[s]$','Interpreter', 'latex');
ylabel('Current $[A]$','Interpreter', 'latex');
```



For a proportional gain of 10, the current stabilizes in around 87 [A]. The maximum current value is:

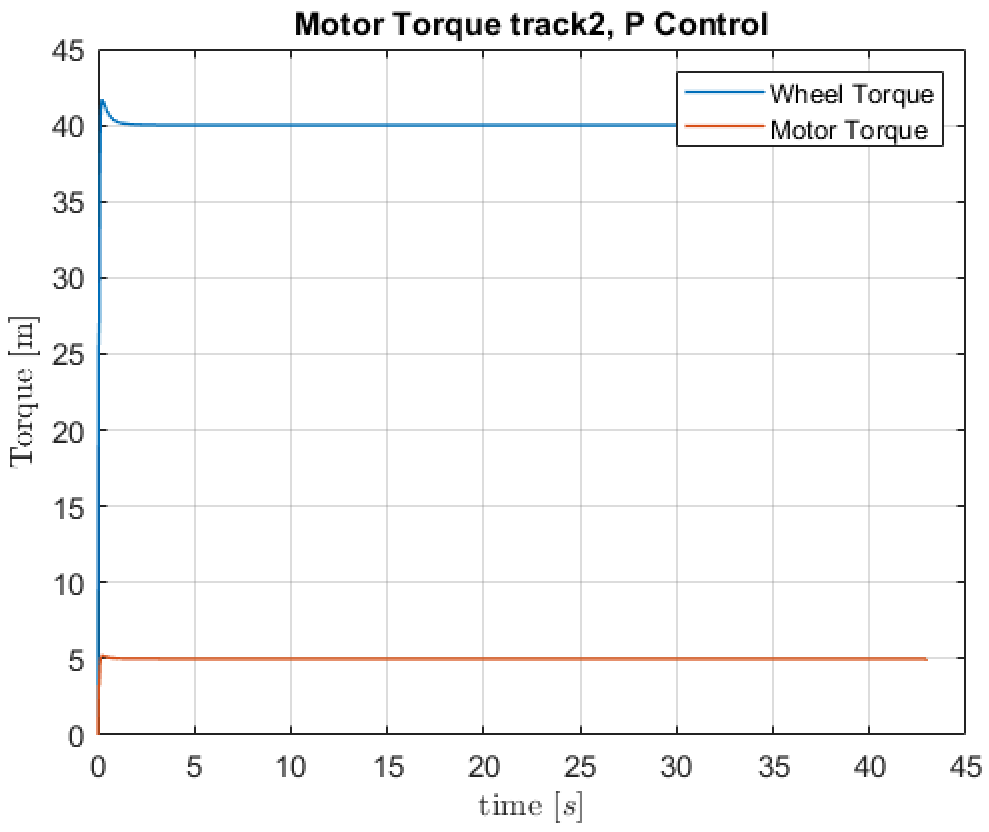
```
fprintf('%d [A]',max(abs(track2_part3_p.stator_current)));
```

```
8.766743e+01 [A]
```

Propotional Control, Motor and Wheel Torque, Track2

The resulting motor and wheel torque are given by:

```
figure()
plot(track2_part3_p.time, track2_part3_p.wheel_torque, 'LineWidth',
 1);
hold all;
plot(track2_part3_p.time, track2_part3_p.motor_torque, 'LineWidth',
 1);
set( gca, 'FontSize', 11);
grid on;
title('Motor Torque track2, P Control');
xlabel('time $[s]$','Interpreter', 'latex');
ylabel('Torque [m]','Interpreter', 'latex');
legend('Wheel Torque', 'Motor Torque');
```



One can see that, as expected, the value of the motor torque stabilizes in 40 [Nm]. The maximum initial torque is:

```
fprintf('%d [A]',max(track2_part3_p.wheel_torque));
```

```
4.167757e+01 [A]
```

Proportinal Integral Control

Now, a non zero integral controller constant is added. This allows the angular velocity of the motor to reach the reference in finite time. Alse, the proportional constant can be increased in order to acheive a faster response of the system.

Proportional Integral Control Track1

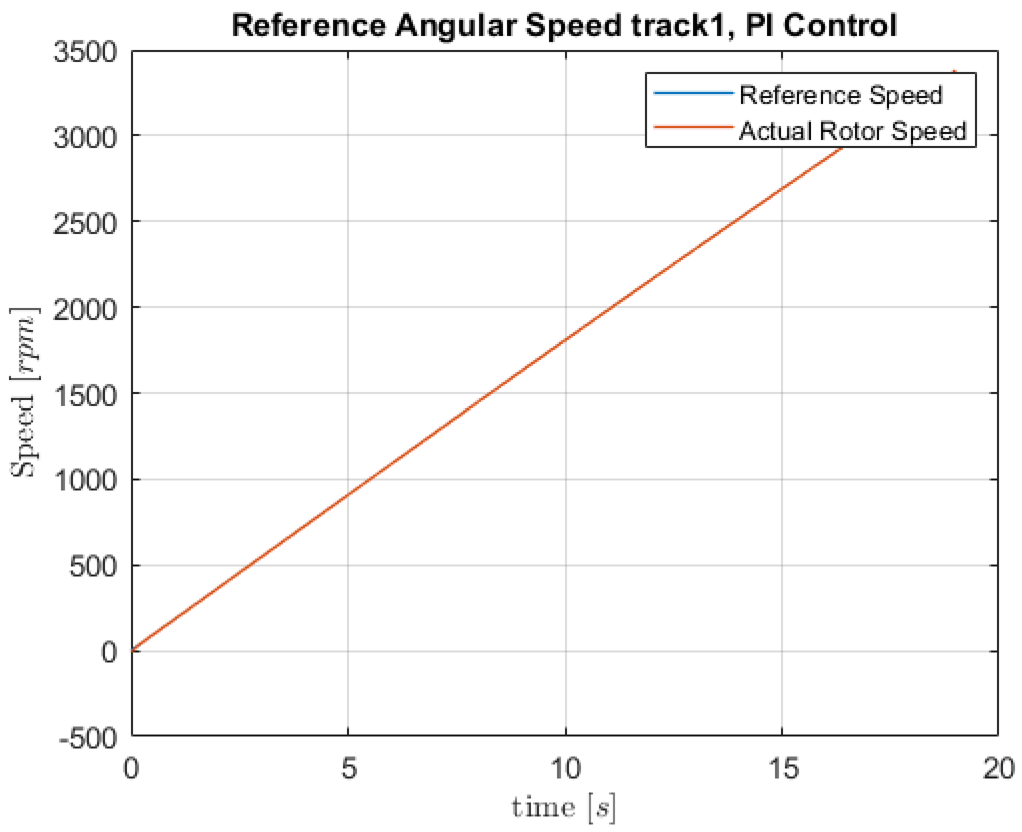
```
proportinal_constat = 20;
integral_constant = 200;

slope_of_track = 3;
reference_speed = reference_speed_track1;
simulation_length_track = distance_A_B;
track1_part3_pi = sim('vienna_car_part3');
```

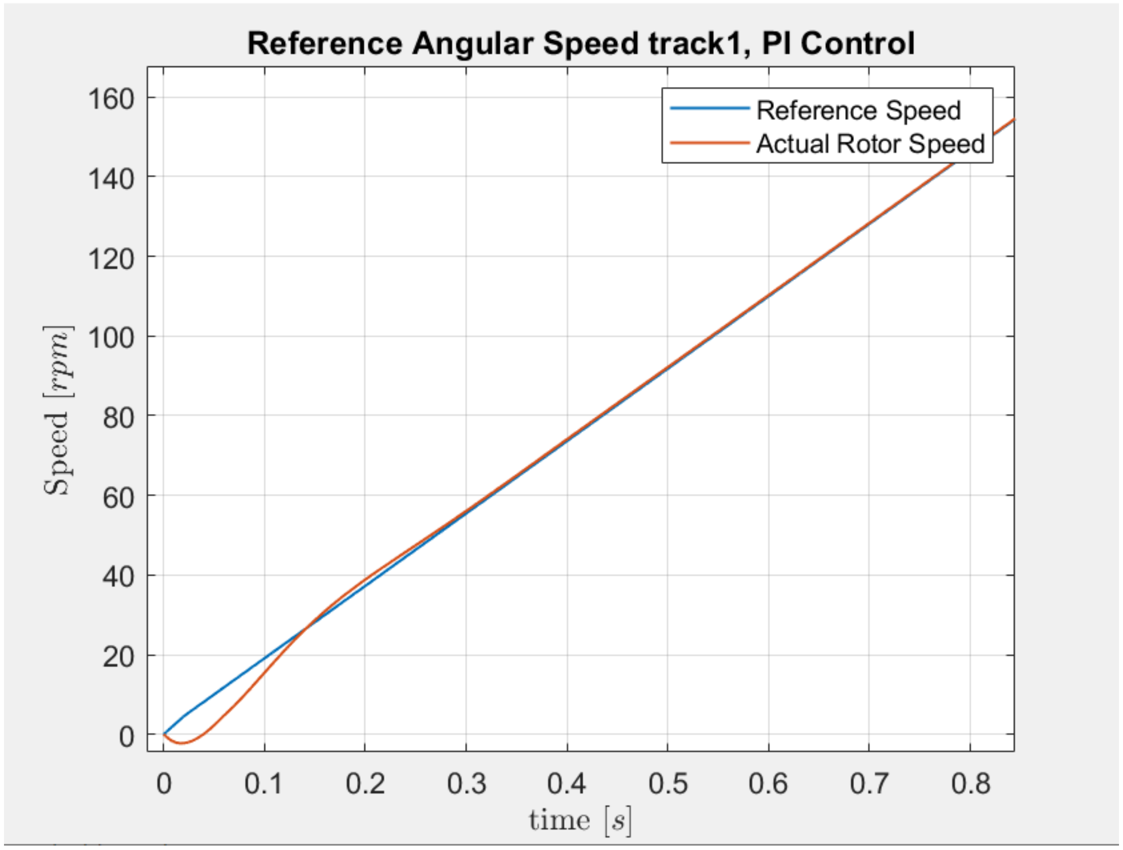
Proportional Integral Control, Reference Speed, Track1

```
The reference angular speed and speed and the output angular speed of the
```

```
figure()
plot(track1_part3_pi.time,
 track1_part3_pi.reference_angular_speed_pid, 'LineWidth', 1);
hold all
plot(track1_part3_pi.time,
 track1_part3_pi.rotor_angular_speed, 'LineWidth', 1);
set( gca, 'FontSize', 11);
grid on;
title('Reference Angular Speed track1, PI Control');
xlabel('time $[s]$','Interpreter', 'latex');
ylabel('Speed $[rpm]$','Interpreter', 'latex');
legend('Reference Speed', 'Actual Rotor Speed');
```



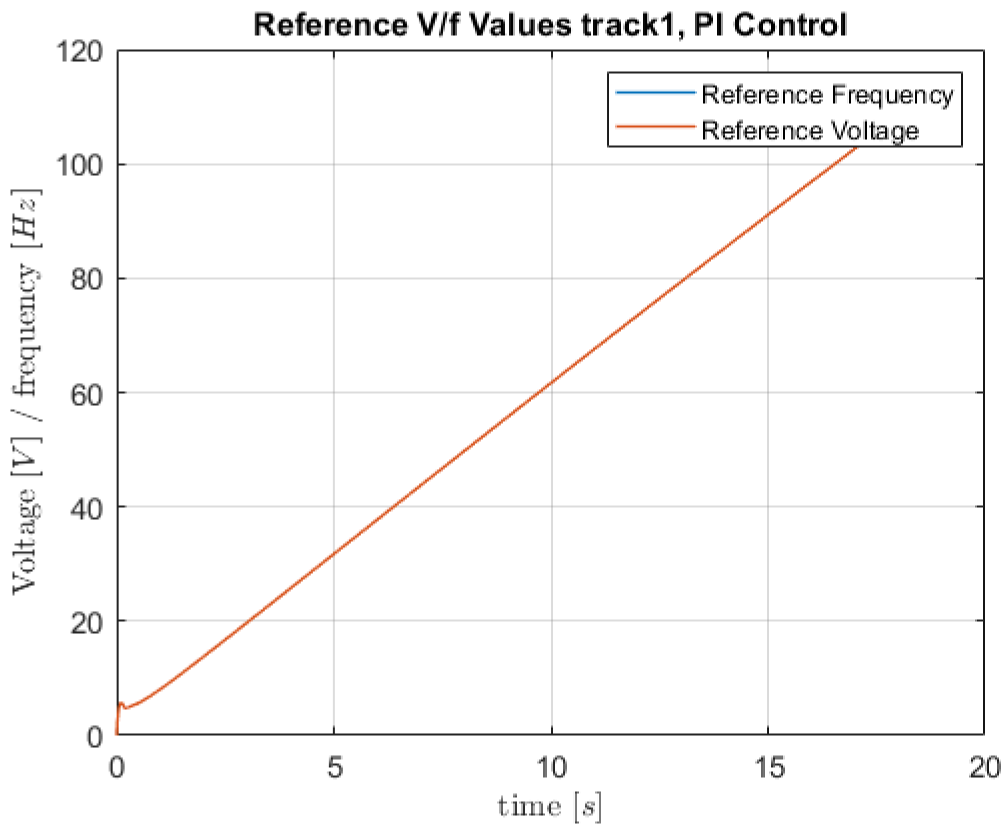
As expected, now the motor speed follows the reference, with a steady state error equal to zero. On top of that, the proportional constant is higher, resulting in a faster response time.



Proportional Integral Control, Voltage and Frequency, Track1

The output V/f values of the VFD block are:

```
figure()
plot(track1_part3_pi.time,
 track1_part3_pi.reference_frequency, 'LineWidth', 1);
hold all
plot(track1_part3_pi.time,
 track1_part3_pi.reference_voltage, 'LineWidth', 1);
set( gca, 'FontSize', 11);
grid on;
title('Reference V/f Values track1, PI Control');
xlabel('time $[s]$','Interpreter', 'latex');
ylabel('Voltage $[V]$ / frequency $[Hz]$','Interpreter', 'latex');
legend('Reference Frequency', 'Reference Voltage');
```



The maximum V/f = 1 value is:

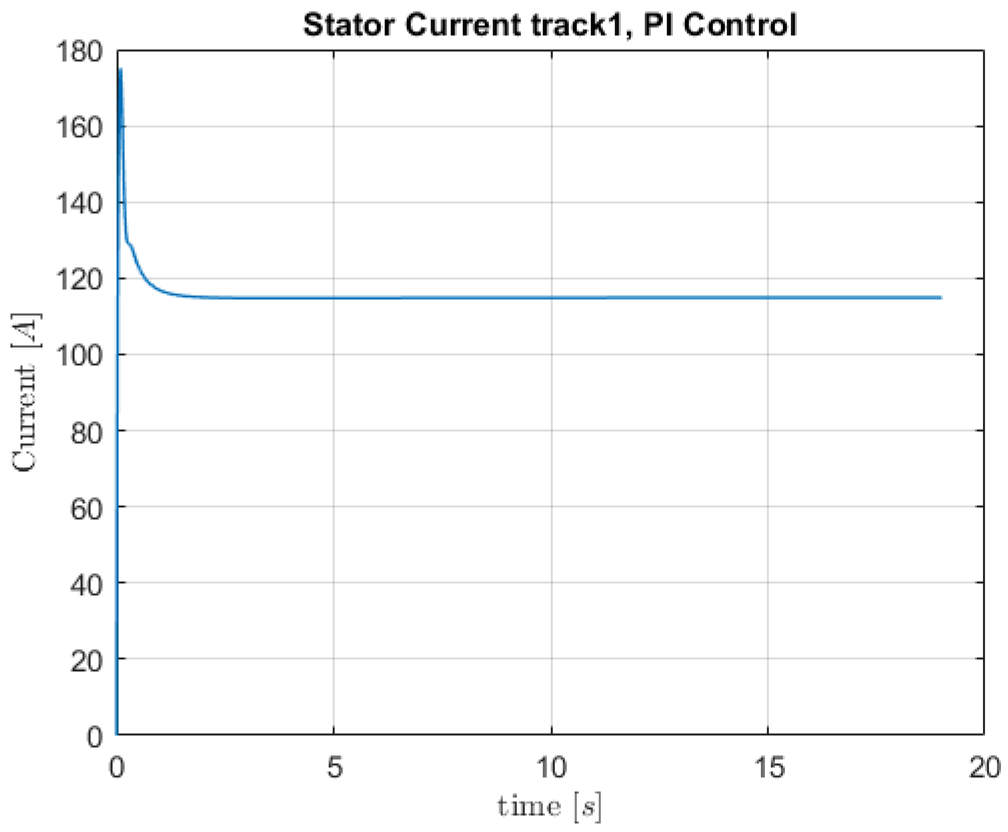
```
fprintf('%d [V]/[Hz]',max(track1_part3_pi.reference_frequency));
```

```
1.140815e+02 [V]/[Hz]
```

Proportional Integral Control, Stator Current, Track1

The previous V/f values result in the following stator current:

```
figure()
plot(track1_part3_pi.time,
 abs(track1_part3_pi.stator_current), 'LineWidth', 1);
set( gca, 'FontSize', 11);
grid on;
title('Stator Current track1, PI Control');
xlabel('time $[s]$','Interpreter', 'latex');
ylabel('Current $[A]$','Interpreter', 'latex');
```



For a proportional gain of 10, the current stabilizes in around 115 [A]. The maximum current value is:

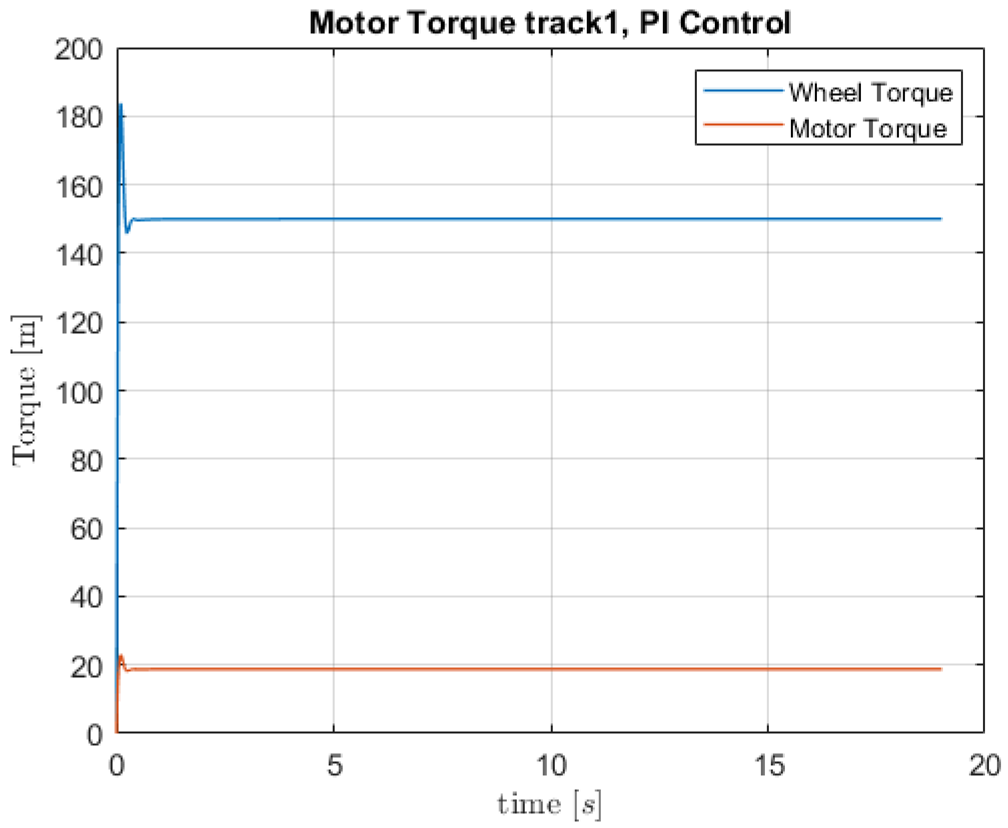
```
fprintf('%d [A]',max(abs(track1_part3_pi.stator_current)));
```

```
1.750539e+02 [A]
```

Proportional Integral Control, Motor and Wheel Torque, Track1

The resulting motor and wheel torque is given by:

```
figure()
plot(track1_part3_pi.time, track1_part3_pi.wheel_torque, 'LineWidth',
 1);
hold all;
plot(track1_part3_pi.time, track1_part3_pi.motor_torque, 'LineWidth',
 1);
set( gca, 'FontSize', 11);
grid on;
title('Motor Torque track1, PI Control');
xlabel('time $[s]$','Interpreter', 'latex');
ylabel('Torque [m]','Interpreter', 'latex');
legend('Wheel Torque', 'Motor Torque');
```



One can see that, as expected, the value of the mortar torque stabilizes in 150 [Nm]. The maximum initial torque is:

```
fprintf('%d [A]',max(track1_part3_pi.wheel_torque));
```

```
1.836883e+02 [A]
```

Proportional Integral Control track2

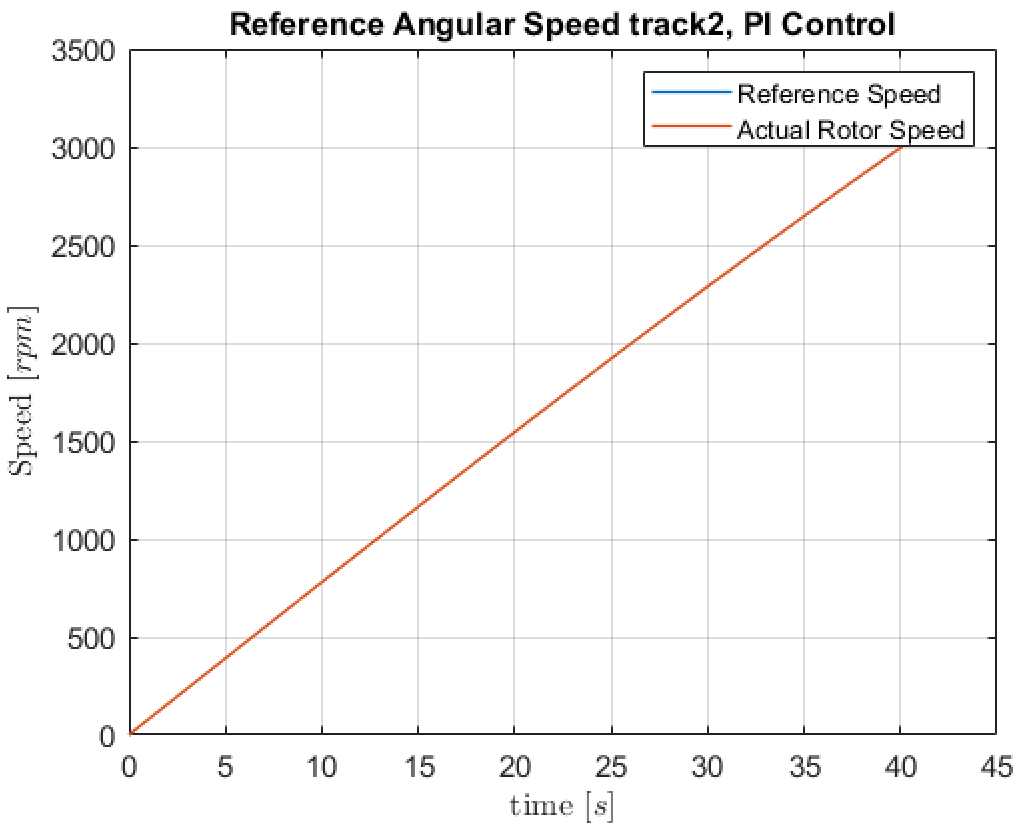
```
slope_of_track = 0;
reference_speed = reference_speed_track2;
simulation_length_track = track2_length;
track2_part3_pi = sim('vienna_car_part3');
```

Proportional Integral Control, Reference Speed, Track2

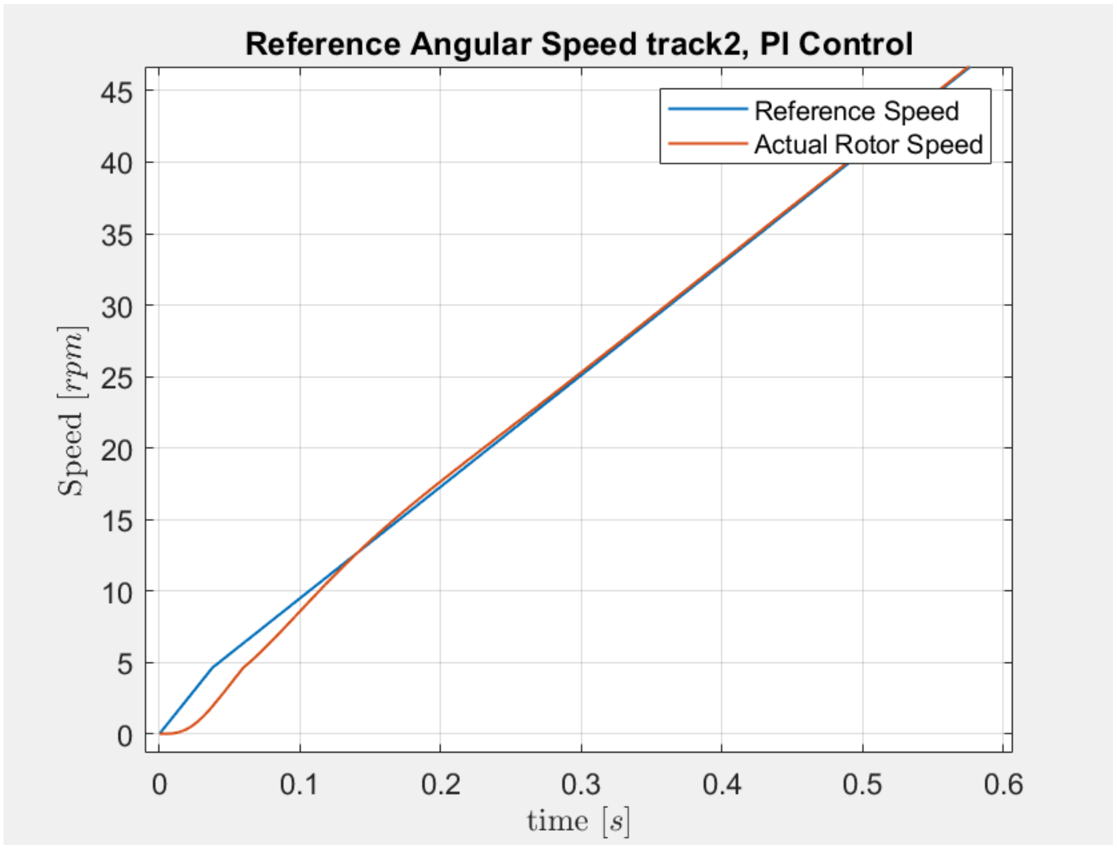
```
The reference angular speed and speed and the output angular speed from
the controller are presented below:
```

```
figure()
plot(track2_part3_pi.time,
 track2_part3_pi.reference_angular_speed_pid, 'LineWidth', 1);
hold all
```

```
plot(track2_part3_pi.time,
 track2_part3_pi.rotor_angular_speed, 'LineWidth', 1);
set( gca, 'FontSize', 11);
grid on;
title('Reference Angular Speed track2, PI Control');
xlabel('time $[s]$','Interpreter', 'latex');
ylabel('Speed $[rpm]$','Interpreter', 'latex');
legend('Reference Speed', 'Actual Rotor Speed');
```



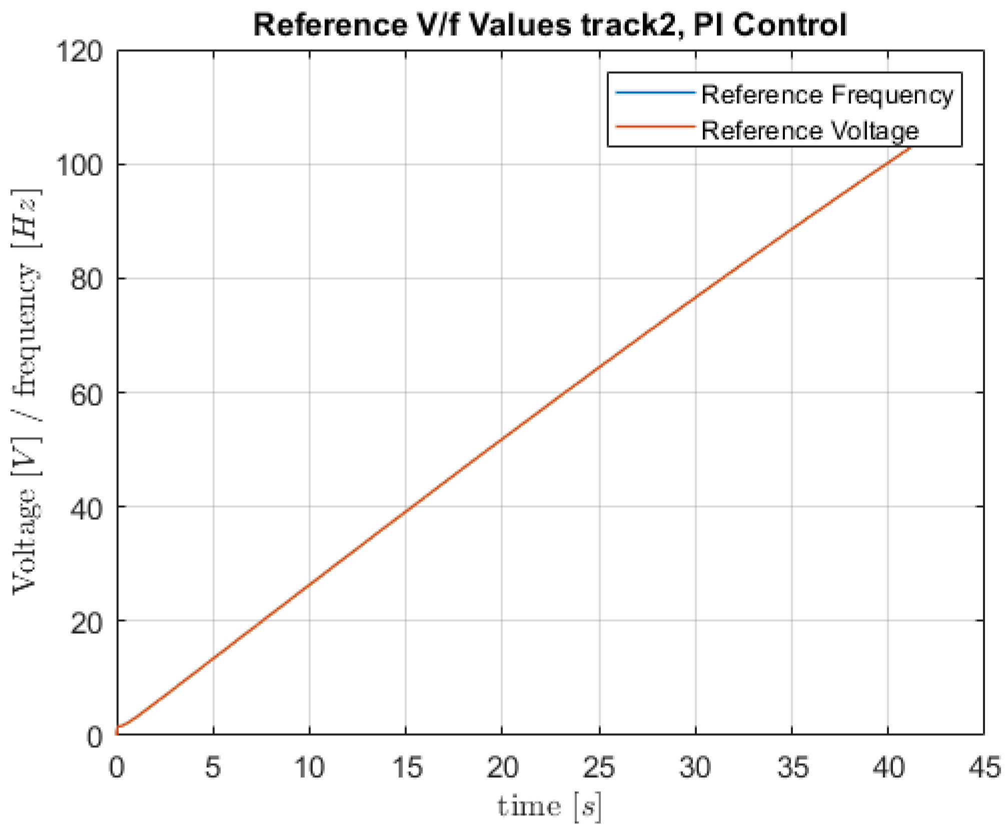
Once again, the motor speed follows the reference, with a steady state error equal to zero.



Proportional Integral Control, Voltage and Frequency, Track2

The output V/f values of the VFD block are:

```
figure()
plot(track2_part3_pi.time,
 track2_part3_pi.reference_frequency, 'LineWidth', 1);
hold all
plot(track2_part3_pi.time,
 track2_part3_pi.reference_voltage, 'LineWidth', 1);
set( gca, 'FontSize', 11);
grid on;
title('Reference V/f Values track2, PI Control');
xlabel('time $[s]$','Interpreter', 'latex');
ylabel('Voltage $[V]$ / frequency $[Hz]$','Interpreter', 'latex');
legend('Reference Frequency', 'Reference Voltage');
```



The maximum V/f = 1 value is:

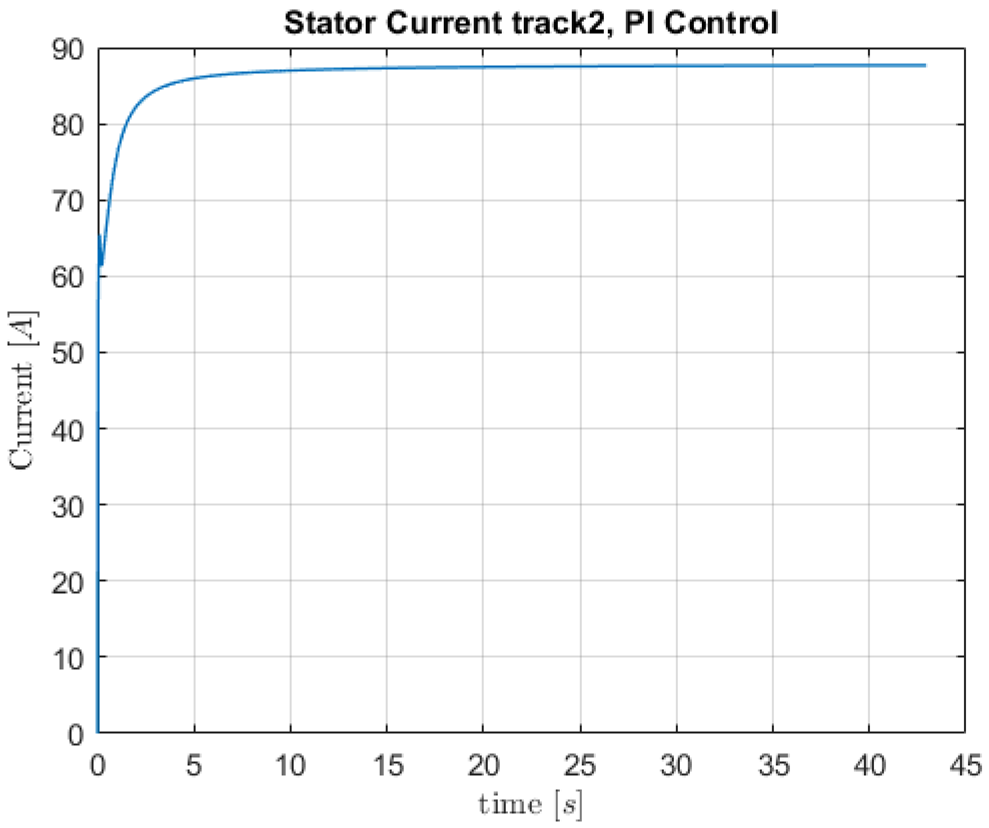
```
fprintf('%d [V]/[Hz]',max(track2_part3_pi.reference_frequency));
```

```
1.069324e+02 [V]/[Hz]
```

Proportional Integral Control, Stator Current, Track2

The previous V/f values result in the following stator current:

```
figure()
plot(track2_part3_pi.time,
 abs(track2_part3_pi.stator_current), 'LineWidth', 1);
set( gca, 'FontSize', 11);
grid on;
title('Stator Current track2, PI Control');
xlabel('time $[s]$','Interpreter', 'latex');
ylabel('Current $[A]$','Interpreter', 'latex');
```



For a proportional gain of 10, the current stabilizes in around 87 [A]. The maximum current value is:

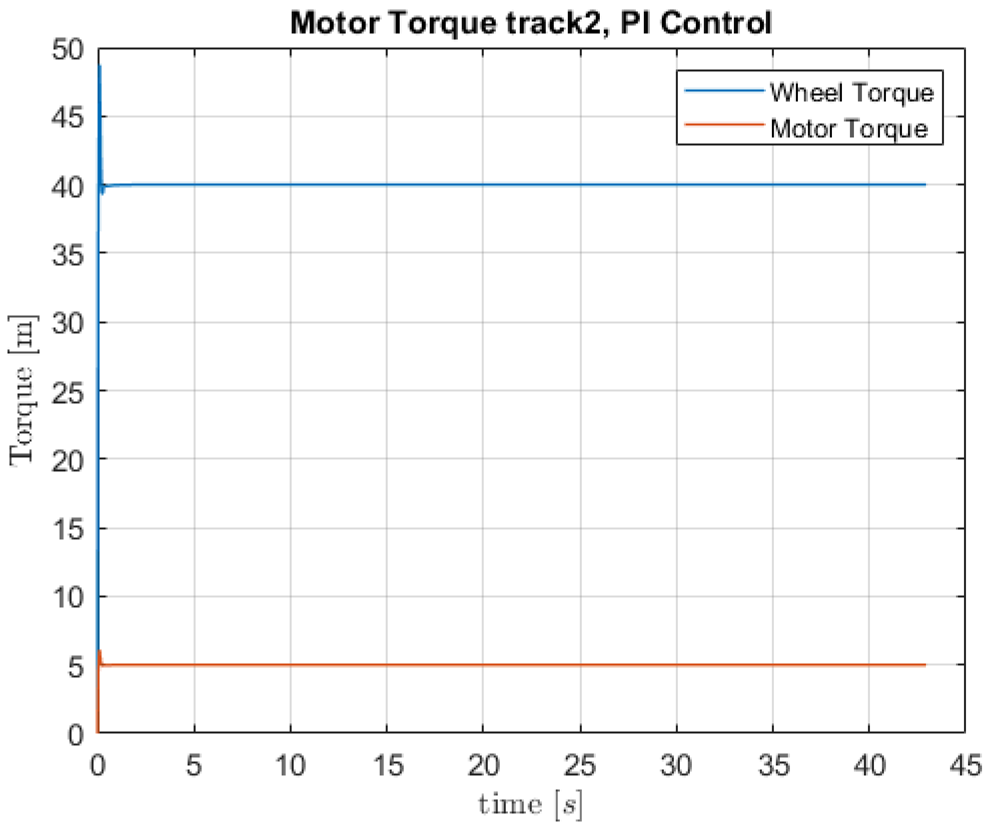
```
fprintf('%d [A]',max(abs(track2_part3_pi.stator_current)));
```

```
8.766777e+01 [A]
```

Proportional Integral Control, Motor and Wheel Torque, Track2

The resulting motor and wheel torque is given by:

```
figure()
plot(track2_part3_pi.time, track2_part3_pi.wheel_torque, 'LineWidth',
 1);
hold all;
plot(track2_part3_pi.time, track2_part3_pi.motor_torque, 'LineWidth',
 1);
set( gca, 'FontSize', 11);
grid on;
title('Motor Torque track2, PI Control');
xlabel('time $[s]$','Interpreter', 'latex');
ylabel('Torque [m]','Interpreter', 'latex');
legend('Wheel Torque', 'Motor Torque');
```



One can see that, as expected, the value of the motor torque stabilizes in 40 [Nm]. The maximum initial torque is:

```
fprintf('%d [A]',max(track2_part3_pi.wheel_torque));
```

```
4.876117e+01 [A]
```

```
close
```

Conclusions

The main differences between the P control and the PI control are:

- **Motor Angular Speed:** As expected, for both tracks, the proportional control had a small steady state error, while the PI control quickly reached the reference value.
- **V/f values:** For both tracks and controllers, the V/f values were in the expected values.
- **Stator current values:** The current behaved the same way, for each controller. In track1, the current stabilized in 115 [A], and in track2, the current stabilized in around 87 [A]. **However**, in the PI controller, due to a higher P constant value, the starting current was bigger. For P = 10, the starting current was around 145 [A], while for P = 20, the starting current was around 157 [A].
- **Motor torque:** In both controls, the torque stabilized in 150 [Nm] and 40 [Nm], for track1 and track2 respectively. The biggest difference is noticed in the starting torque, where for a=P = 10, the starting torque is 154 [Nm] and 41.6 [Nm], for track1 and track2 respectively, while for P = 20, the starting

torque is 183 [Nm] and 48 [Nm], for track1 and track2 respectively. This is a consequence of the fact that bigger proportional constant results in a higher starting current.

Published with MATLAB® R2019b