# Feedback controller tuning on a humanoid robot


Leon van Breugel

DC 2011.009


**Bachelor end project**

Begeleider: dr. MSc. Dragan Kostic


Eindhoven University of Technology

Faculty of Mechanical Engineering

Dynamics and Control


Eindhoven, February 2011

# Table of contents

# Abstract

The subject of this project is Tulip, the humanoid robot at the Eindhoven University of Technology. This robot is developed to take part in the adult size league of the RoboCup competition. Tulip has twelve degrees of freedom, which enable it to walk without any form of external support. The original goal of this project is to improve the motion tracking behavior of each of these twelve degrees of freedom. The first step is identifying the system dynamics by doing frequency response function measurements. With this information improved feedback controllers can be designed. These new controllers will then be tested and revised, until the desired performance is reached. However, these goals are not entirely met. System identification was done on eight degrees of freedom, and actual improved performance has been measured on three degrees of freedom.

# Introduction

Developing a machine that has the same capabilities as a human has always interested many scientists around the world. In the year 50 AD Hero of Alexandria, a Greek mathematician, described a machine that could automatically pour wine for party guests. Also Leonardo da Vinci contributed several ideas. The first actual humanoid robots however were only developed in the late 1980's, and this development is continuing into the 21$^{st}$ century, with the HRP-4, developed by the Japanese National Institute of Advanced Industrial Science and Technology, as the most recent example. In the future, these robots should substitute people in a variety of tasks in industry, household, services and care [1,2]. Eindhoven University is also contributing to this development with Tulip. Tulip is being developed to take part in the adult size humanoid soccer league. To perform well in this competition, TUlip has 12 degrees of freedom, six on each leg. These degrees of freedom are:

- Hip rotation around x, y, and z axis.
- Knee rotation around y axis.
- Ankle rotation around x and y axis.

See figure 1.1 for clarification.

Every degree of freedom is separately actuated and controlled. At the start of this project, each actuator was controlled by a PD-controller. These PD-controllers are tuned using a trial-and-error method, so their performance left room for improvement. The goal of this project is to develop new controllers in a scientifically correct manner. The first step in this process is to identify the system. This is done by measuring frequency response functions. Using the data from these FRF measurements, an improved controller is proposed. This controller is then tested and revised until the performance reaches the desired level.

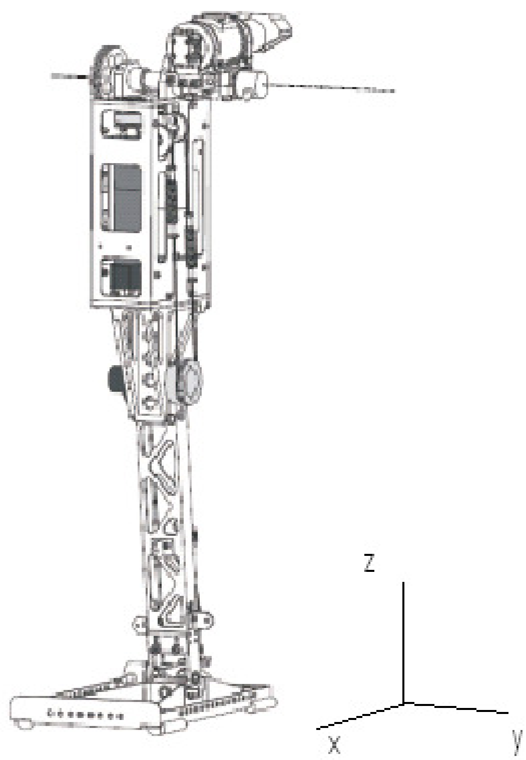



*Figure 1.1: Tulip's degrees of freedom.*

## FRF measurements

A frequency response function is a measure of any system's output spectrum in response to an input signal. The frequency response is typically characterized in a Bode plot by the magnitude of the system's response, measured in decibels (dB), and the phase, measured in radians, versus frequency. From these measurements several things can be learned about the specifications of the system, for example the stiffness of the system, and the presence of any (anti)resonances in the system.

First, some theoretical background on FRF measuring is discussed.

Suppose we have a input-output relation:

$$y = H(s)x \tag{2.1}$$

Then the autocorrelation (2.2) and the crosscorrelation (2.3) are defined as

$$\varphi_{xx}(\tau) = E[x(t)x(t-\tau)] \tag{2.2}$$

$$\varphi_{xy}(\tau) = E[x(t)y(t-\tau)] \tag{2.3}$$

If we take the Fourier transform of these equations, we obtain the auto power spectrum (2.4) and the cross power spectrum (2.5).

$$\varphi_{xx}(\tau) \overset{F}{\rightarrow} \Phi_{xx}(\omega) \tag{2.4}$$

$$\varphi_{xy}(\tau) \overset{F}{\rightarrow} \Phi_{xy}(\omega) \tag{2.5}$$

If we then take the quotient of these power spectrums, we obtain the transfer function H, and the autocorrelation $\Phi_{yy}$.

$$H(j\omega) = \frac{\Phi_{xy}(\omega)}{\Phi_{xx}(\omega)} \tag{2.6}$$

$$\Phi_{yy}(\omega) = H(j\omega)H(-j\omega)\Phi_{xx}(\omega) \tag{2.7}$$

If we then look at the input-output relation with noise included:

$$y = H(s)x + n \tag{2.8}$$

Second, we assume that *n* and *x* are not correlated, than the auto power spectrum $\Phi_{yy}$, and the quotient $\Phi_{yy}\Phi_{xx}$ become:

$$\Phi_{yy} = H^2\Phi_{xx} + \Phi_{nn} = \frac{\Phi_{xy}^2}{\Phi_{xx}} + \Phi_{nn} \tag{2.9}$$

$$\Phi_{yy}\Phi_{xx} = \Phi_{xy}^2 + \Phi_{nn}\Phi_{xx} \tag{2.10}$$

We can now define coherence $\Gamma$ as:

$$\Gamma = \frac{\Phi_{xy}^2}{\Phi_{yy}\Phi_{xx}} \tag{2.11}$$

This coherence function shows what part of the measured output results from the input, instead of the noise. This makes the coherence a handy tool in assessing the reliability of the measured frequency response.

There are several ways to measure the frequency response of a system. This will be explained using figure 2.1, a schematic representation of the closed-loop control scheme.

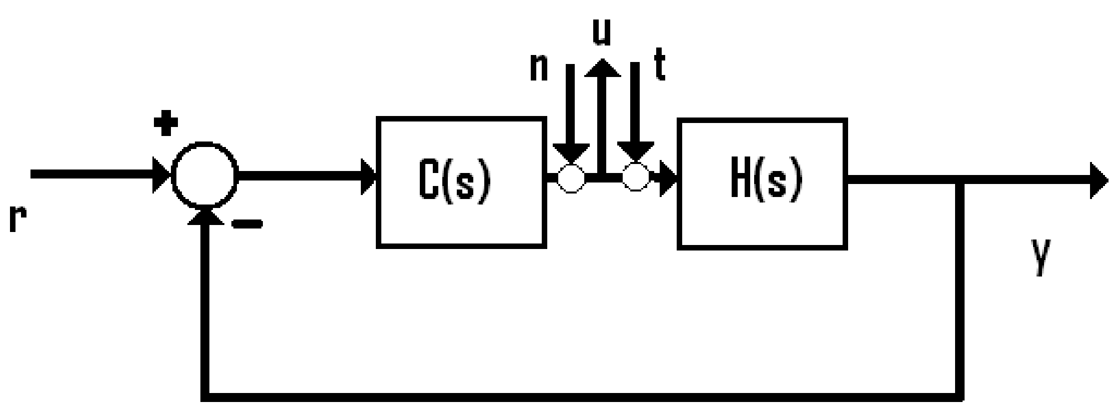



*Figure 2.1: Schematic representation closed-loop control scheme.*

In this figure, r is the reference, n the measurable noise, u a measurable output, t the non-measurable system noise, and y another measurable output. For this particular scheme, we have the following equations for y and u:

$$y = H(n + t - Cy) \rightarrow y(1 + HC) = H(n + t) \tag{2.12}$$

$$u = n - C(Ht + Hu) \rightarrow u(1 + HC) = n - HCt \tag{2.13}$$

The quotient of these two equations then becomes:

$$\frac{y}{u} = \frac{Hn + Ht}{n - HCt} \tag{2.14}$$

One method to measure H, is to make sure that the input signal is dominant over the system noise *t*. In this way, H can be measured directly. Another method is to apply a measurable noise *n* to the system, which is dominant over the system noise *t*. These two methods are applied to measure the frequency response function of Tulip.

The first method involves generating a sine-like signal containing various frequencies. As mentioned above, this input signal should be much larger than the system noise. The signal can be generated using a Matlab file [3], which allows the user to specify several characteristics of the measurement signal, such as the frequency range, the amplitude, and the number of signal periods. This Matlab file can be seen in Appendix E. An example of a measurement signal can be seen in figure 2.2. Because the input signal itself is low-frequent, this method gives quite reliable results in the low frequency range, but not so much in the higher frequency range.

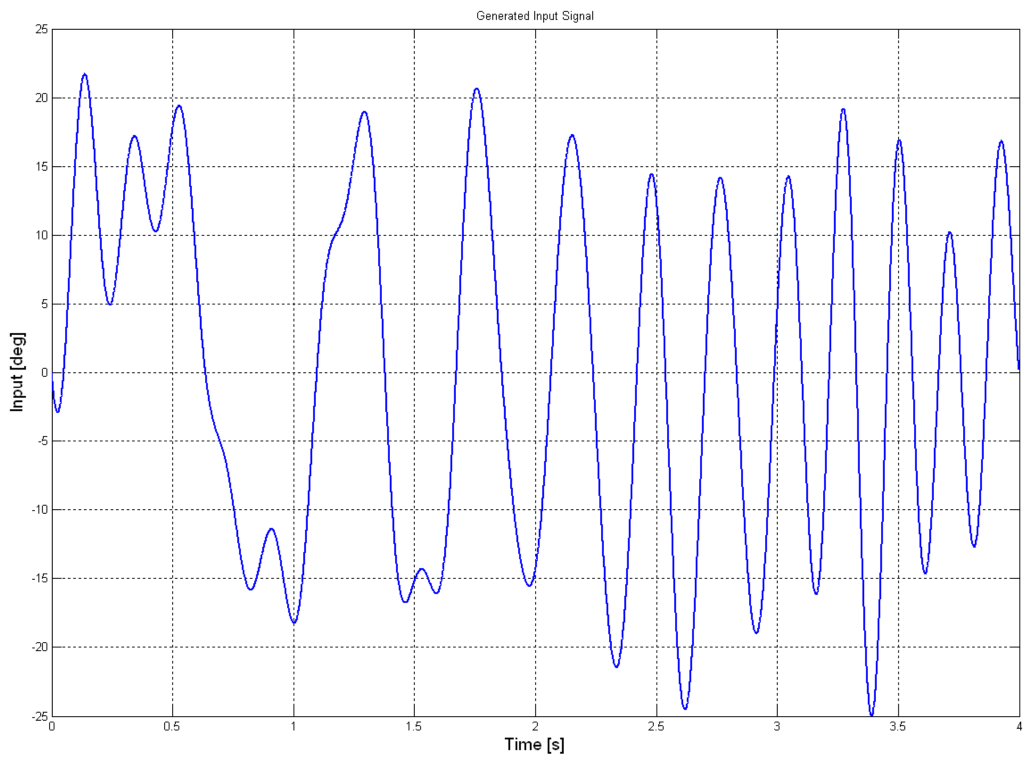



*Figure 2.2: A system identification signal containing various frequencies*

Another way to measure an FRF is by applying a white noise to the system. This white noise is a random signal with a flat power spectral density. In other words, this signal contains practically every frequency, as mentioned above, this white noise should have a large enough amplitude to rule out influences from non-measurable system noise. This enables measurement of the complete FRF over a large bandwidth in a very short period. In addition to this white noise, the degree of freedom is simultaneously subject to a cosine signal. This is necessary to prevent the stick-slip phenomenon to occur. Stick-slip is caused by the difference in static and kinetic friction, causing a system to alternately move (slip) and stand still (stick). This can cause unreliable results during FRF measurements. By applying a constant movement to the system, this is largely prevented. A picture of the system identification signal is shown in figure 2.3. In comparison with the first method, this gives more reliable results in the higher frequency range, but the results are less reliable in the lower frequency range. This is because the input signal itself is also high-frequent. The results from the two measurements have to be compared to obtain the best result.

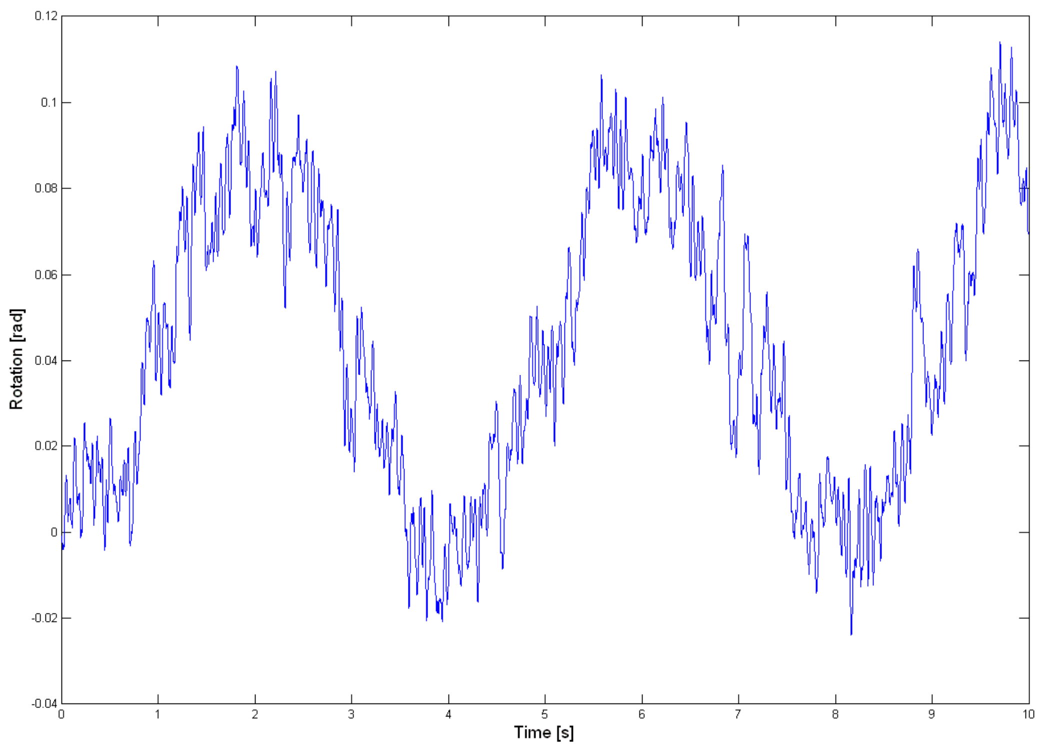



*Figure 2.3: Signal for measuring FRF's*

It should be noted that during an experiment, when a particular degree of freedom is subjected to the signal mentioned above, data is sampled using a sampling frequency of 1000Hz. Therefore, the FRF measurements are limited to 500Hz. This follows from the Nyquist-Shannon sampling theorem [4].

Then the measurements have to be processed. All data is logged in a text-file. This text file is then read into Matlab, and using the command ‘tfestimate’ the frequency response function of the particular degree of freedom is estimated, which can then be visualized. In figure 2.4 the bode plot of the frequency response function of the right hip x-rotation is shown.

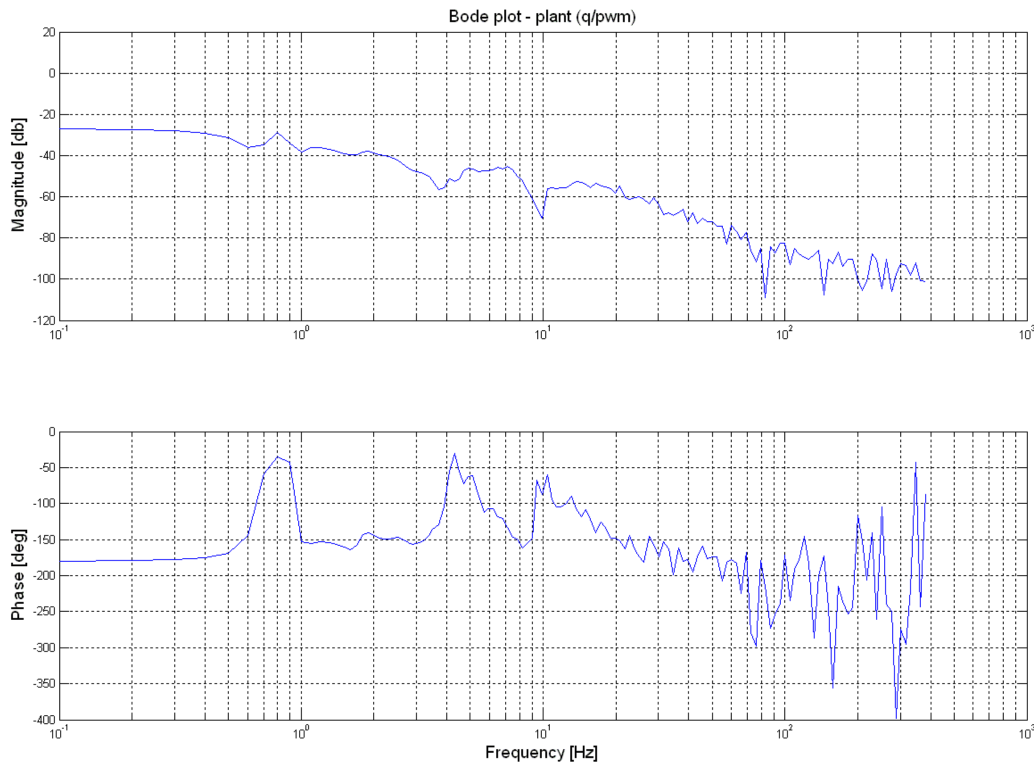



*Figure 2.4: Bode plot of the FRF of right hip x rotation.*

Some important things can be noted from this graph. First, a large anti-resonance is visible at 10 Hz, and a somewhat smaller one at 4 Hz. Furthermore, the graph is nearly flat for frequencies lower than 1 Hz. This seems to be incorrect, because a slope of -2 (40 dB/decade) would be expected, since Tulip is an inertia-dominated system. It can also be seen that at higher frequencies, noise starts to influence the measurement more and more.

The measurement is now visualized, but to be able to use this data for controller design, a parametric function has to be fitted through the measured FRF data. This can be done using the Matlab function ‘frfit’ [5]. This only gives a rough fit, so it has to be manually altered afterwards to achieve a good fit, that takes all resonances and such into account. The result can be seen in figure 2.5.

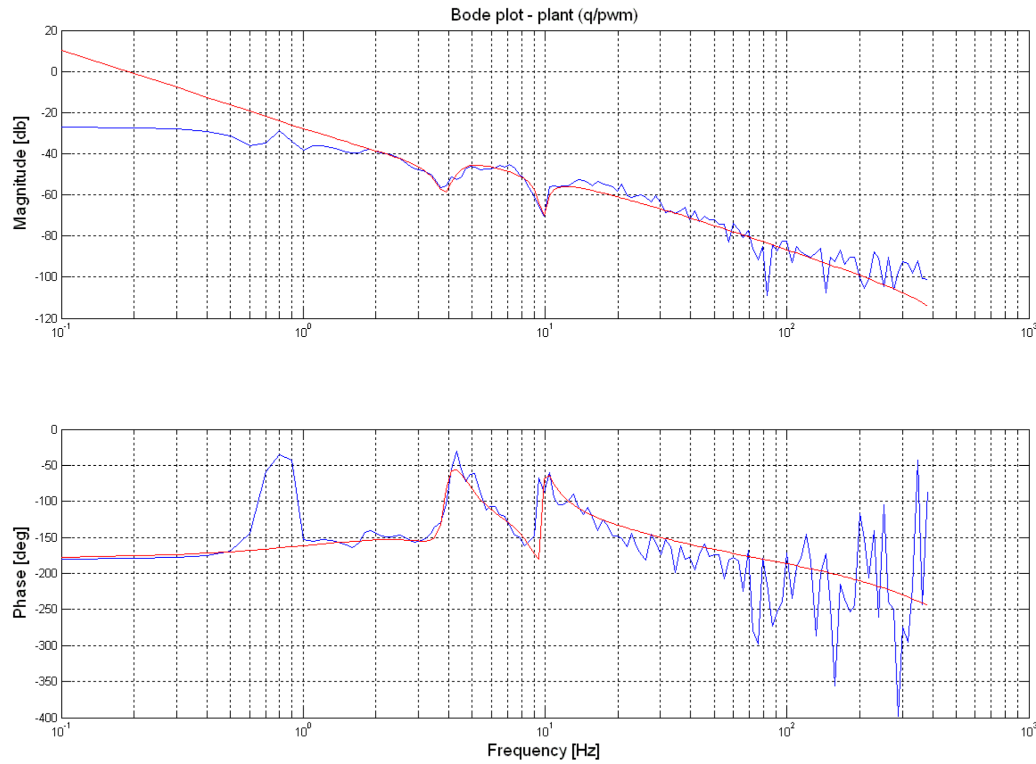



*Figure 2.5: The FRF measurement with the parametric fit in red.*

The fit differs on several areas from the measurement data. At frequencies below 2 Hz, a straight line with slope -2 is used, because at these low frequencies, measurement data tends to be unreliable. Furthermore, at high frequencies the fit shows a bit of phase lag, where the measurement data seems to suggest some phase advance. This however is not correct, since phase advance at an increasing frequency is very unlikely. The remaining fits can be found in Appendix A. For the knee Y and ankle X degrees of freedom, no measurements were done.

## Controller design

The control scheme of Tulip is shown in the figure 3.1. It is a feedback control loop.

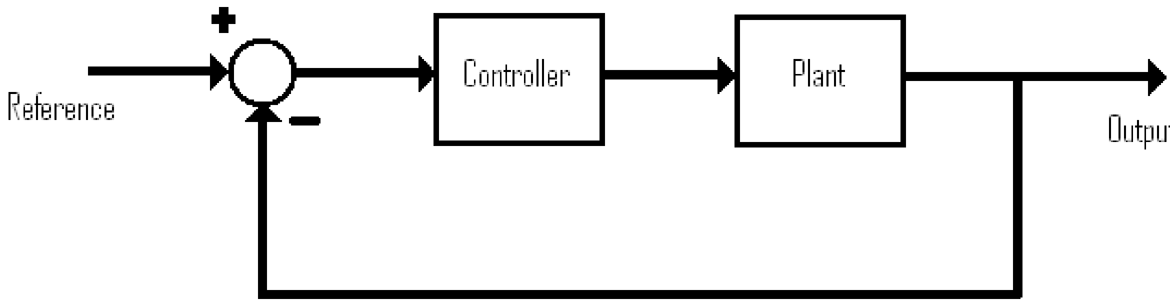



*Figure 3.1: Control scheme of Tulip.*

The characteristics of the system are known from the FRF measurements, and with this information a controller can be designed. The goal is to design a stable controller with better tracking behavior than the current controllers. To achieve this better tracking behavior, a high bandwidth is desirable. Furthermore, a low-pass filter is added. This is necessary to suppress high frequent vibrations in the system.

To guarantee a stable system the following requirements should be met:

1) The poles of the discrete-time open-loop transfer function should all have a magnitude smaller than 1. If these poles have a magnitude larger than 1, the Nyquist criterion will become too complicated for a bachelor end project. Therefore, in the scope of this project, these poles should all have a magnitude smaller than 1.
2) The sensitivity is the degree in which measurement errors influence the output. Increasing the performance of a system usually has a negative effect on the sensitivity, so some kind of balance should be found between the sensitivity and the performance. For this project, the maximum allowable sensitivity is chosen to be 5 dB, which corresponds to a factor 1.78.
3) The phase margin is the amount of phase lag a system can have at the 0-dB frequency before reaching a 180 degrees phase lag. This should be between 30 and 60 degrees, so the phase lag at the 0-dB frequency should be between 120 and 150 degrees. If the phase lag becomes lower than 120 degrees, settling time will become larger. If phase lag becomes higher than 150 degrees, the overshoot will become larger.
4) The gain margin is the margin between 0-dB gain and the gain at the frequency where the phase lag reaches 180 degrees. In practice, this is the factor with which the gain can be multiplied before instability occurs. For this project, the gain margin is chosen to be a factor 2, or 6 dB.

5) In addition to all the above criterions, the Nyquist criterion is applied. This is done by making a polar plot of the open loop transfer function. There should be no encirclements of the point (-1,0). If there are any of these encirclements, then the system is unstable.

These conditions were not met for every degree of freedom. For left hip y rotation and left ankle y rotation I did not manage to design a controller which met the stability criteria. The simulation test results for these controllers are shown in Appendix C. For left hip y rotation, the simulation shows instable behavior. The simulation for left ankle y shows stable behavior, but this does not guarantee stability when the controller is actually applied to the robot.

The resulting controller for the right hip x rotation is discussed below, information about other controllers can be found in Appendix B.

- Proportional gain of 900
- Differential gain of 14
- Low-pass filter time constant of 0.002

This has the following results:

- Phase margin of 60 degrees
- Gain margin of 9.81 dB
- Maximum sensitivity of 4.83 dB
- All open loop poles have a magnitude smaller than 1
- 0 dB frequency of 59.2 rad/sec

The Nyquist plot, which confirms the stability of the system, is shown in figure 3.2.

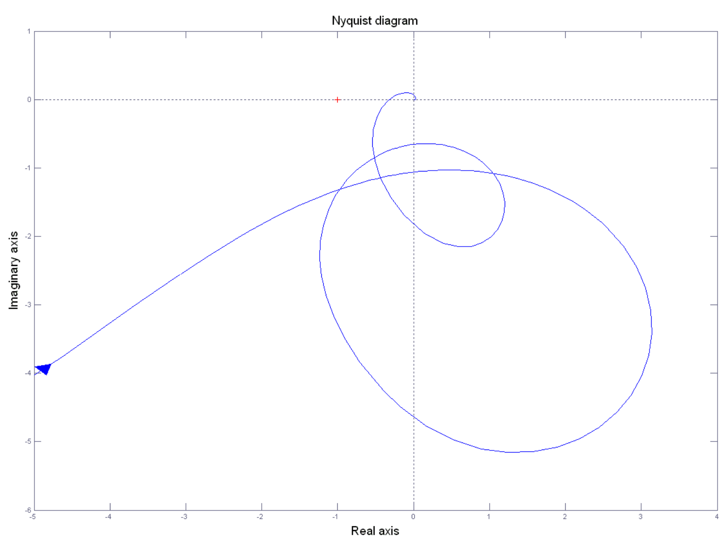



*Figure 3.2: Nyquist plot for right hip x rotation.*

## Results

For safety reasons, the influence of a new controller on the system is first simulated using Simulink. The used Simulink program can be seen in figure 4.1.

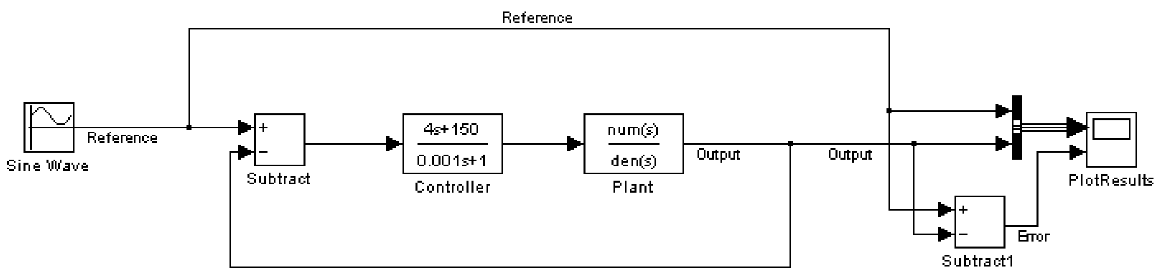



*Figure 4.1: Simulink program*

The dynamic behavior of the system with the new controller can be tested using this program. The response to a sine signal is tested. The results of the Simulink model and the actual test results are shown in figures 4.2 and 4.3. Both show a test on the right hip Y rotation, with the old controller.

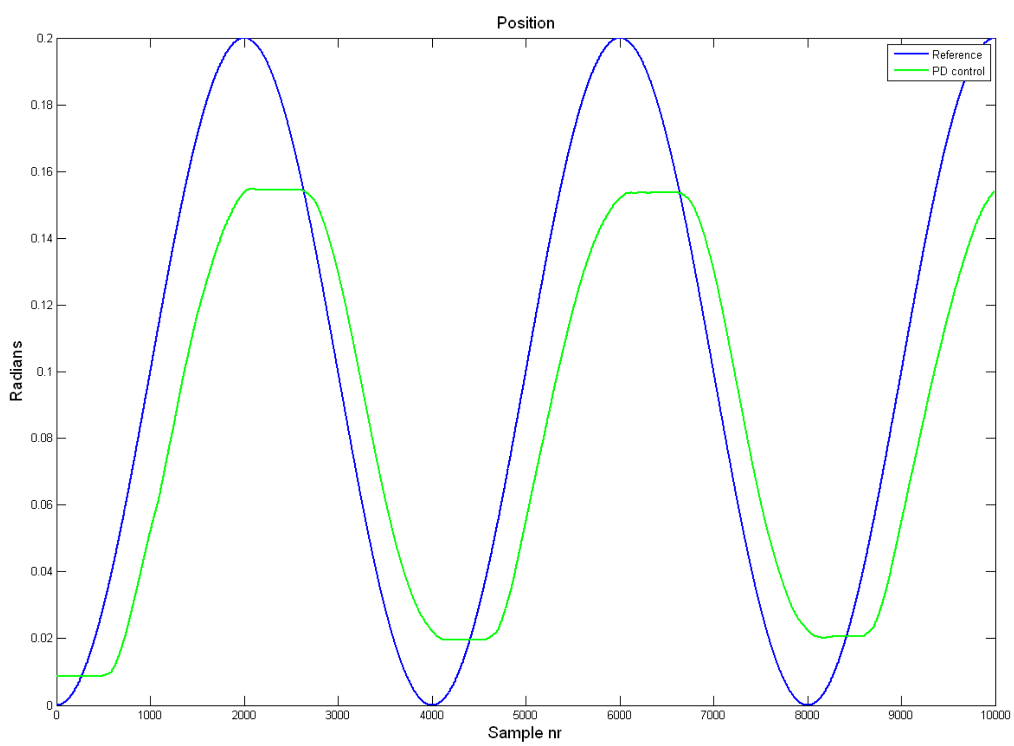



*Figure 4.2: Measurement results right hip y rotation*

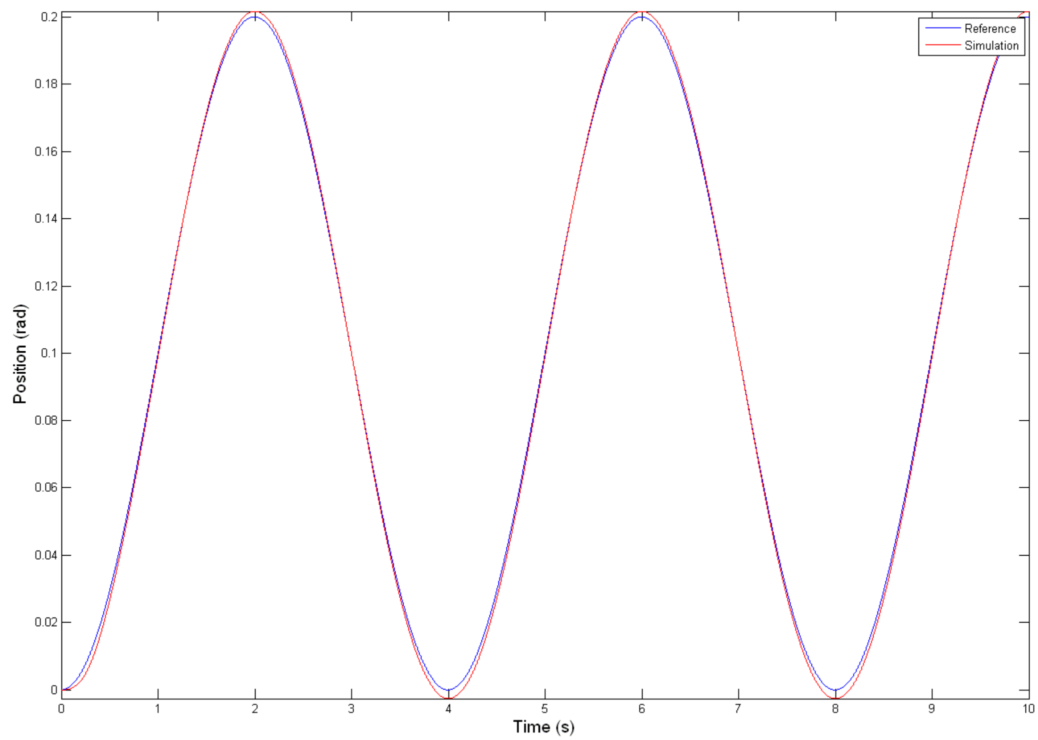



*Figure 4.3: Simulation results right hip y rotation*

It can be seen that the simulation shows much better tracking behavior than the actual controller. This difference is caused by the fact that the Simulink model only includes a linear transfer function, and neglects the non-linear effects [8] that are present in the robot. One of the differences between the simulation and the measurement result is the poor tracking behavior, the 'flattening' of the graph, near the minima and maxima of the sine signal. This phenomenon is caused by friction. The friction in a degree of freedom in the robot  is depicted in figure 4.4.

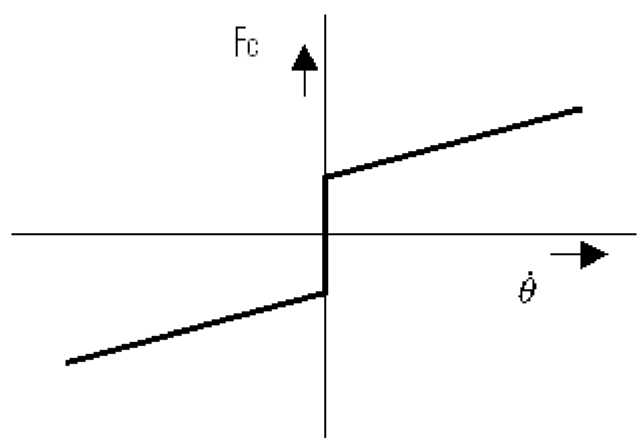



*Figure 4.4: Friction force as function of rotation speed.*

The friction characteristics can be described by the following formula (1.1):

$$F_c = b \cdot \dot{\theta} + c \cdot sgn\dot{\theta} \qquad (4.1)$$

In Tulip the Coulomb friction, the second term, is the most dominant. This explains the 'flattened' pieces in the graph.

Another large difference is the delay in the tracking behavior. This is caused by the fact that torque applied by the motor is not directly transferred to the corresponding degree of freedom. There is a gearbox and several wires between the motor and the degree of freedom, which cause the delay in the response.

For several degrees of freedom no actual tests were carried out on the robot. This has several explanations. For left hip y rotation and left ankle y rotation, no adequate control was designed, so this obviously was not tested. For right hip z rotation and left hip z rotation there are no test results either, because the robot did not respond to the input signal during the test. This is probably caused by a defect encoder. Also for right hip x rotation there is no result, this is because the test on left hip x rotation did not go well, and the testing on both hip x rotations was cancelled. See Appendix D for more information regarding the test on left hip x rotation.

For these degrees of freedom, only the Simulink results are available. These results may not be able to predict the actual behavior of the robot, but they can be used to compare different controllers. If a controller performs better than another one in Simulink, it will most likely also perform better when it is actually implemented. In figure 4.5, the simulated tracking error is displayed for the old and the new controller on the right hip y rotation. It can be seen that the tracking error is in general smaller using the new controller. The plot of the tracking behavior is not shown, because the difference is then nearly invisible. For the other simulation results, see Appendix C.

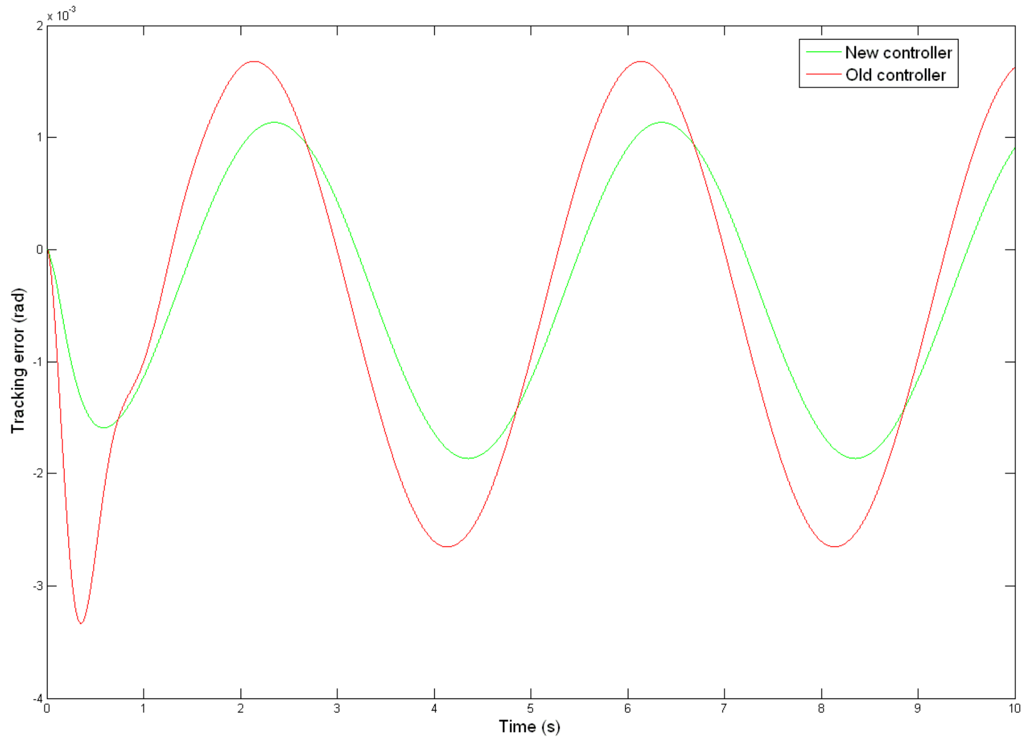



*Figure 4.5: Simulated tracking error right hip y rotation.*

According to the simulation, the new controller will give better performance. The only way to find out if this is true is testing it on the robot. For safety reasons, the new controller is first tested at only 10% gain. If this does not cause any problems, the gain is increased. This process continues until problems arise, or until 100% gain is reached. During the testing of the new controller for the right hip y rotation, the robot starts shaking more and more as the gain is increased. At 70% gain, the vibrations become to large to continue to test at higher gains. In figures 4.6 and 4.7, the tracking behaviour and tracking error with this new controller at 70% gain can be seen. It is clearly visible that the performance is better with the old controller. The vibrations can also be seen, especially near the maxima and minima of the sine signal. Furthermore, it should be noted that the sampling frequency is 1000Hz, and thus 1000 measurement samples correspond to 1 second measurement time.

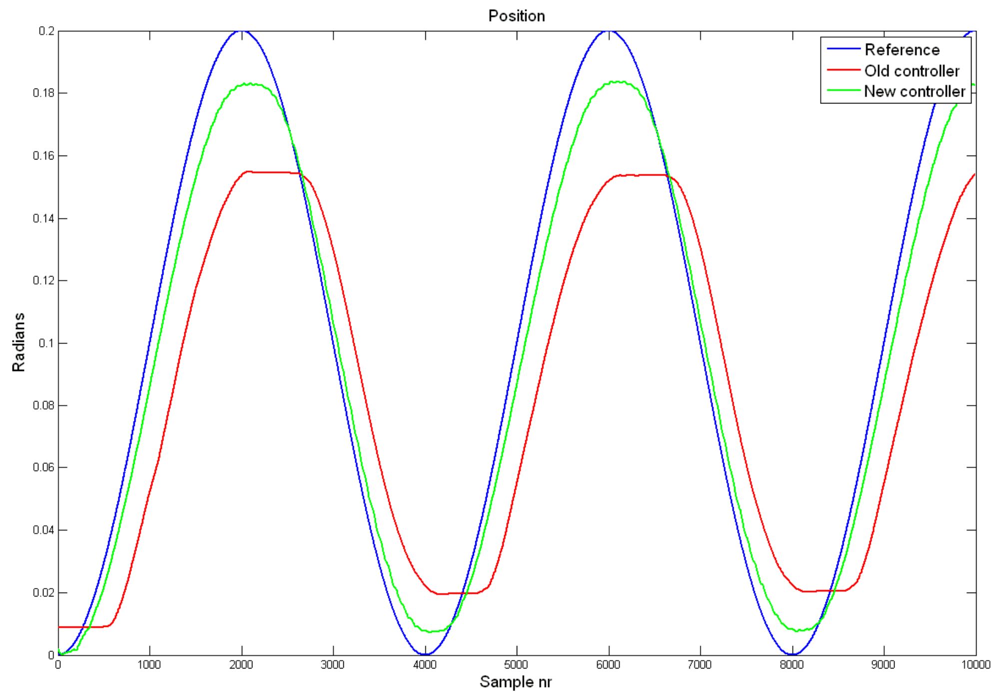



*Figure 4.6: Tracking behavior of the new and old right hip y controller*

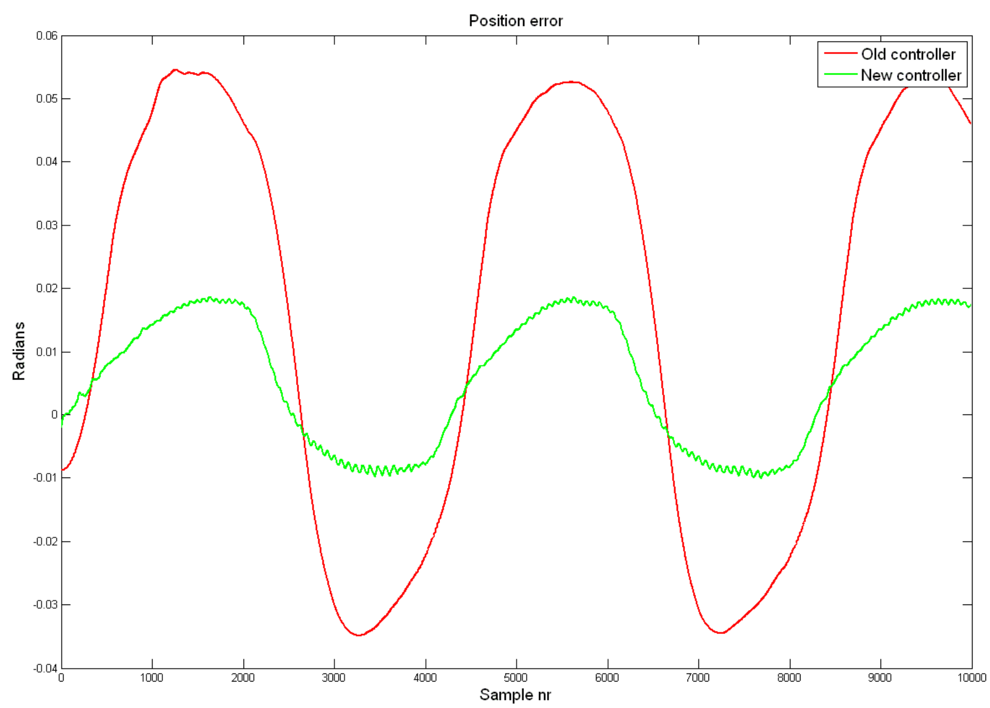



*Figure 4.7: Tracking error of the new and old right hip y controller*

The vibration can be suppressed by implementing a notch filter. To effectively do this, the frequency of the vibration should be known. To determine this frequency, a PSD plot is made. This plot is shown in figure 4.8. The vibration can be seen just above 10 Hz.

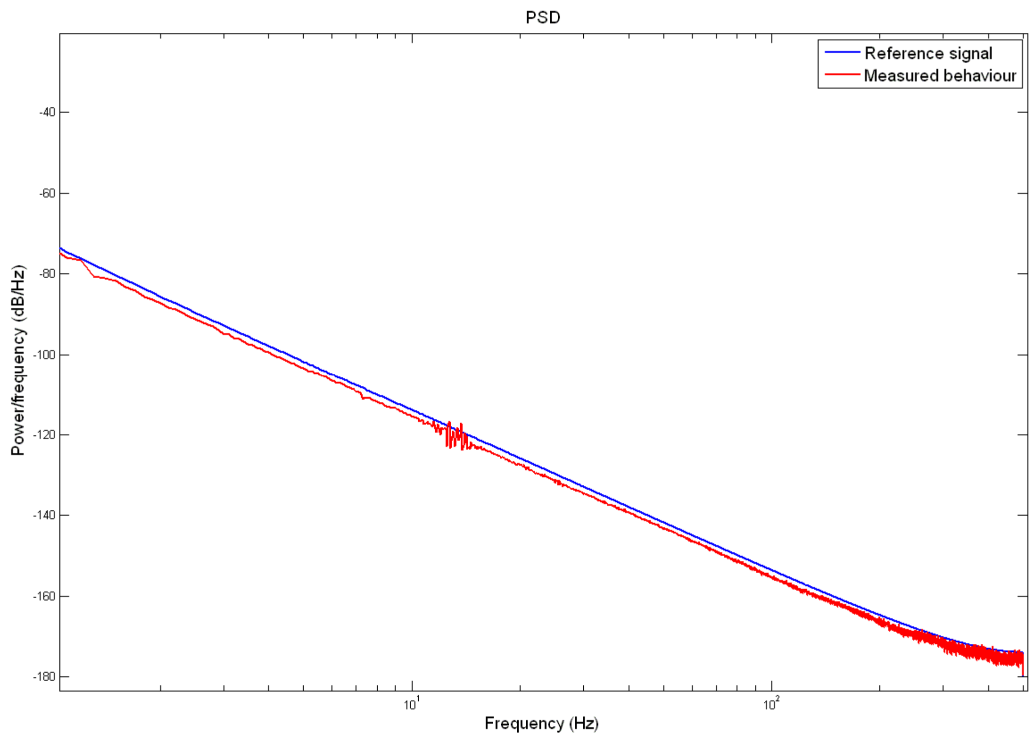



*Figure 4.8: Power spectrum density plot of the reference signal and the measured behavior.*

With this data, a notch filter can be designed to suppress this frequency. When this is tested on the robot however, it shows instable behavior, where the model predicts a stable system. This instability is thus caused by errors in the model. As mentioned earlier, the model does not include non-linear effects such as friction. In order to successfully implement a notch filter in this controller, further measurements and research are required. However, due to a lack of available measurement time and the complexity of the matter, this is not included is this bachelor end project.

The results from the other tested controllers can be found in Appendix D.

## Conclusions

The goal of this project was to design a new feedback controller for all twelve degrees of freedom of TUlip. Not all goals have been reached due to several technological reasons, as well as a lack of time. FRF measurements have been done on eight degrees of freedom. This supplied valuable information about the dynamic behavior of the robot and can be used in further research and development. For six of these eight degrees of freedom, a stable controller was designed. In theory, these controllers should improve the tracking behavior. In practice however, only two of these controllers were successfully tested and a decreased tracking error was measured. The testing of the four other controllers was cancelled because of technological problems or unexpected behavior. The two controllers that did work properly can still be improved, since accurate controller tuning is a process that requires a large amount of time in testing and revising.

## Recommendations

There are several points that should be investigated in following projects:

Knee y and ankle x rotation on both legs are yet to be investigated.  It would be very interesting to do FRF measurements on these degrees of freedom, and develop adequate controllers. Also the controller design process should be completed for all other degrees of freedom. This includes designing adequate controllers for left hip y rotation and left ankle y rotation, and testing and revising the controllers for the remaining degrees of freedom. It is likely that for several degrees of freedom a notch filter has to be implemented to design a decent controller, so research should be done on the cause of the instable behavior when a controller with notch filter is implemented.

The hip z rotation controllers are working quite well at this moment, but a better accuracy is possible.

Another valuable tool in future research could be a Simulink model that includes nonlinear effects such as friction. Once this is developed, simulations can give much more accurate information about the behavior of the robot.

The possibility of implementing a feedforward should be investigated. A feedforward could help improve tracking behavior.

# Reference list

[1] Humanoid robots, B. Choi, In-tech, Rijeka, 2009

[2] Handbook of robotics, B. Siciliano and O. Khatib, Springer, Berlin, 2008

[3] generate_ident_signal.m, which uses msinclip.m (I. Kollar and Vrije Universiteit Brussel, ELEC, 1991-2004). Further reading: http://dali.feld.cvut.cz/ucebna/matlab/toolbox/fdident/msinclip.html

[4] Lecture notes ‘Signal analysis’, P.W.M. Nuij, 2007

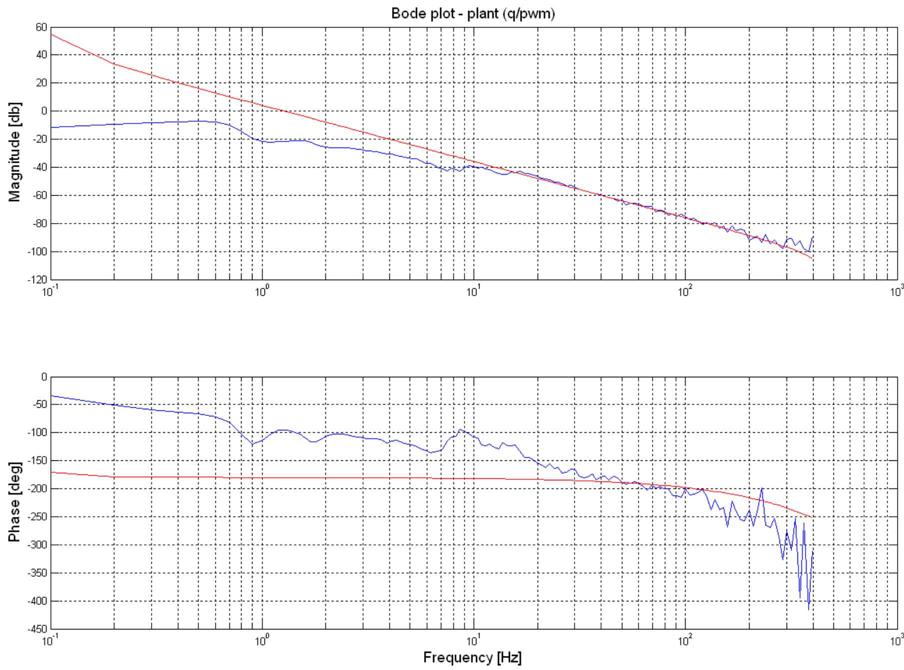
[5] frfit.m (revision 1.6), P. Gahinet and P. Apkarian, Copyright 1995-2002 The MathWorks, Inc.

[6] Control system tuning, M. Steinbuch and M. Vervoordeldonk, 1996, Philips research centre for technical training.

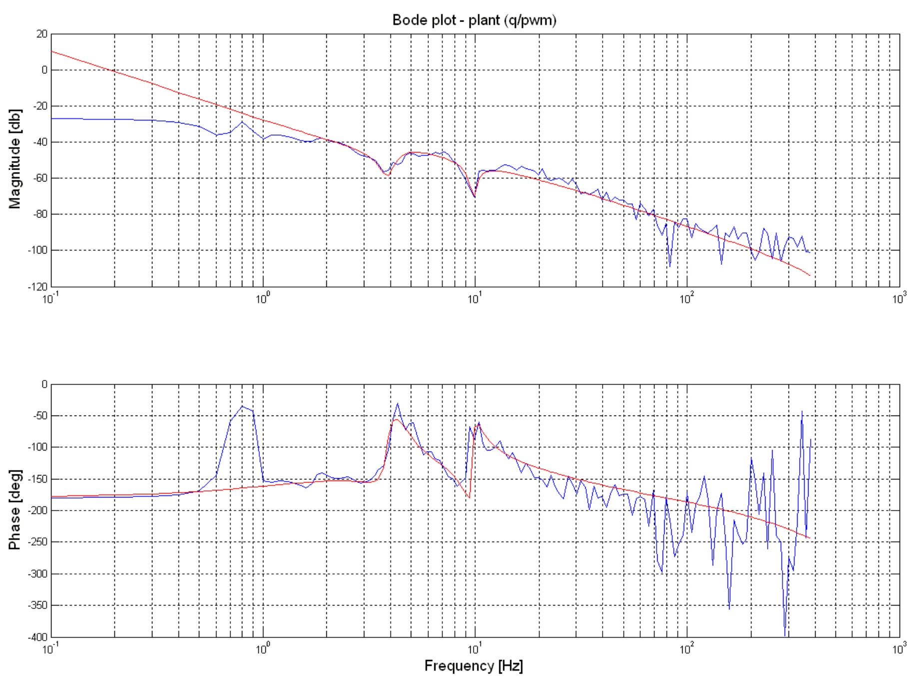
[7] Lecture notes ‘Regeltechniek’, M. Steinbuch and J. Kok, 2007

[8] Nonlinear systems , H. Khalil, 2002

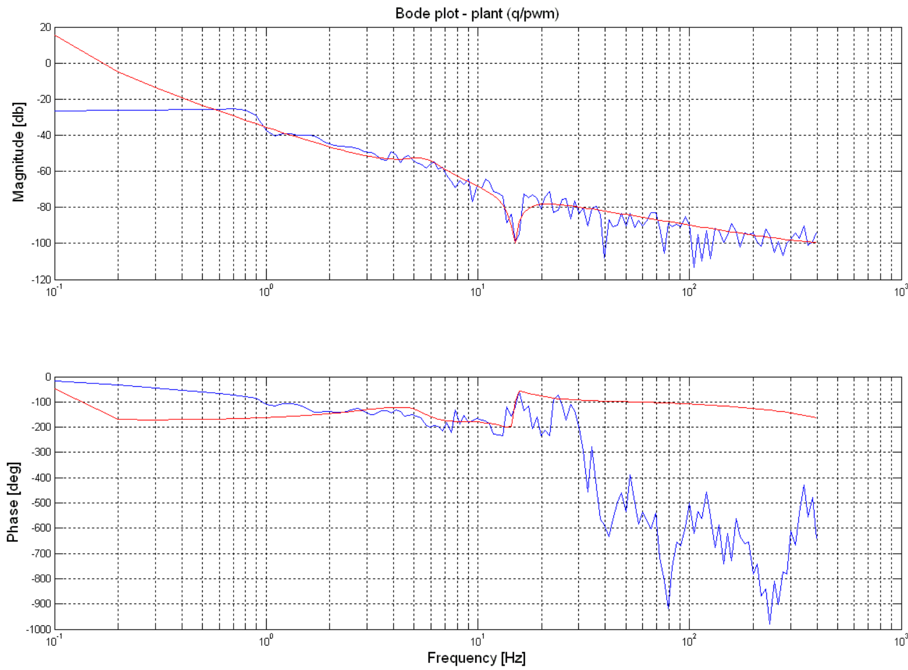
# Appendix A – FRF data fits



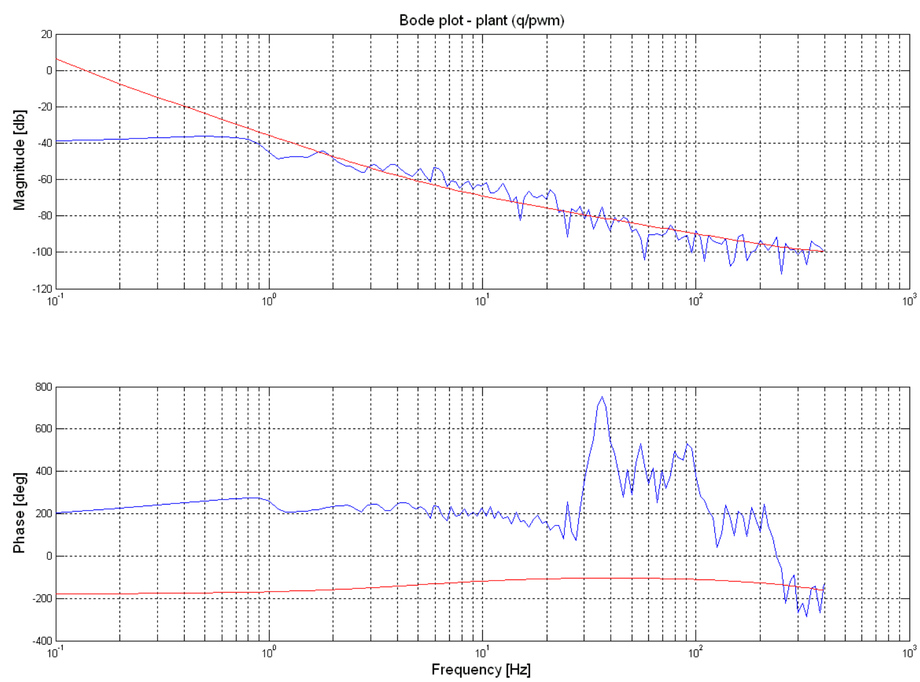

*Right hip z rotation*



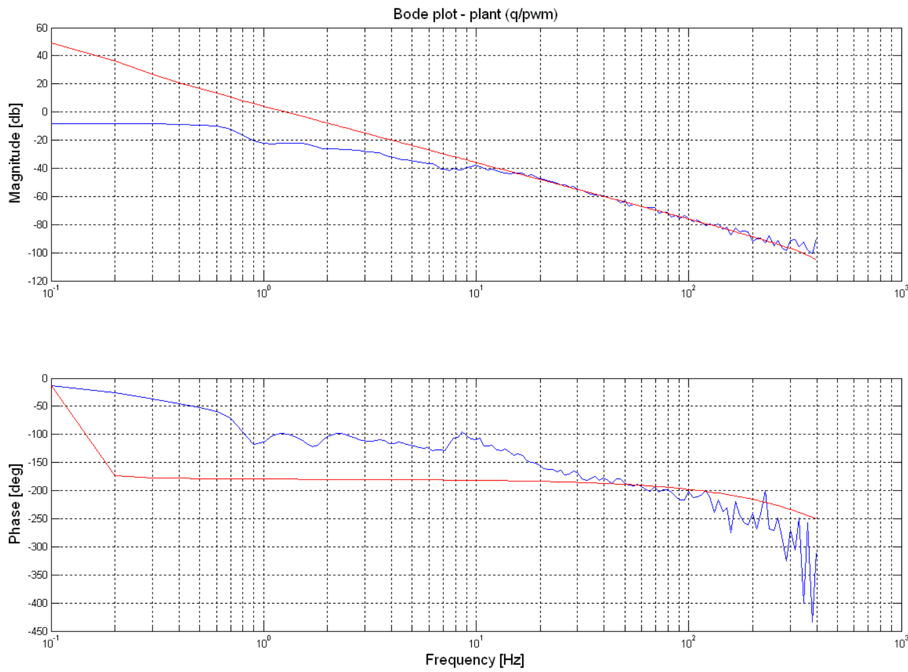

*Right hip x rotation*



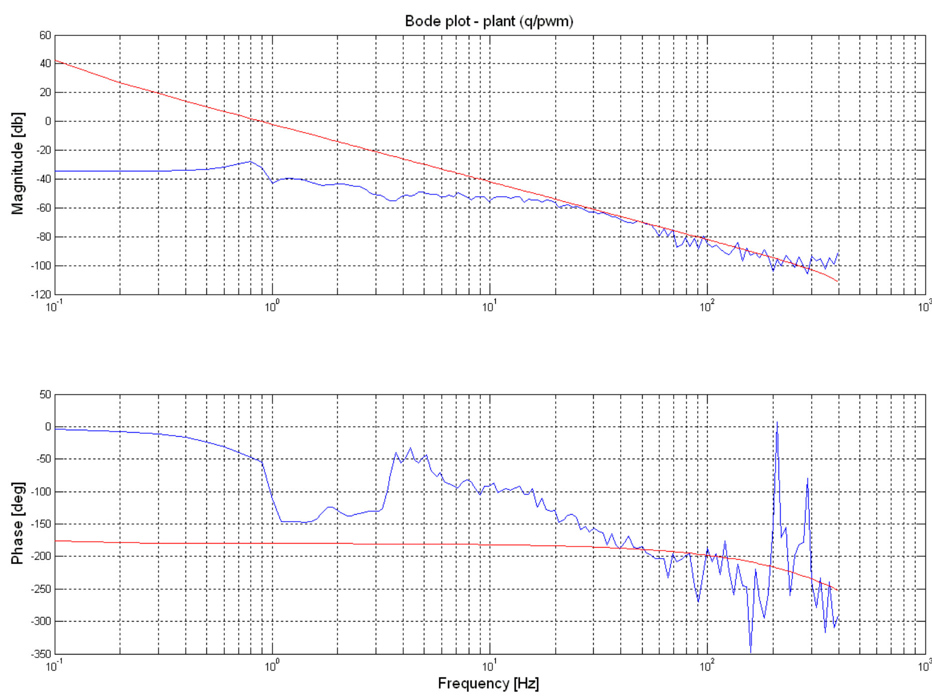

*Right hip y rotation*



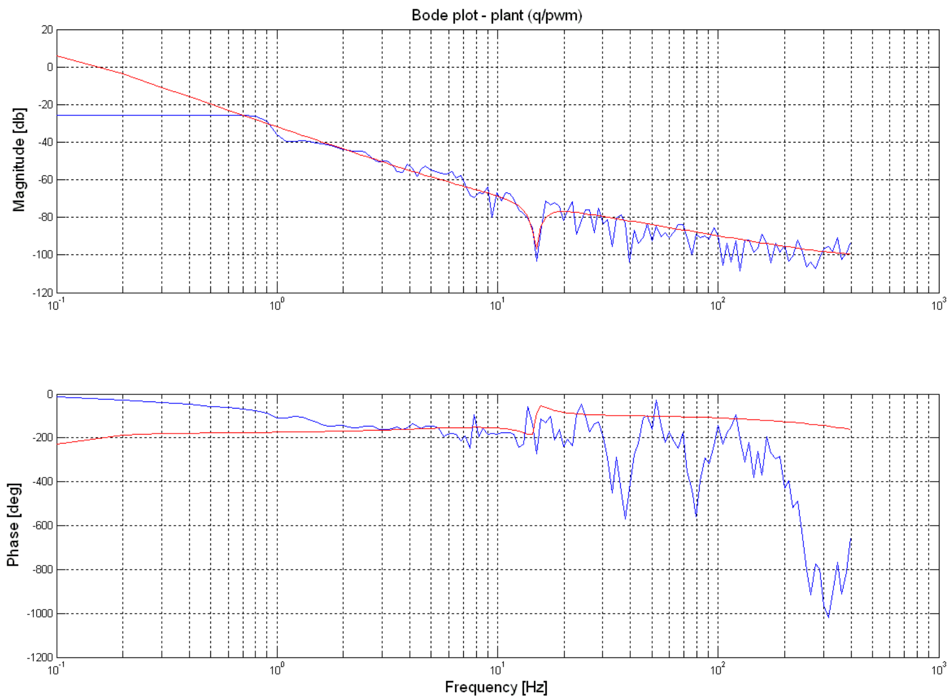

*Right ankle y rotation*



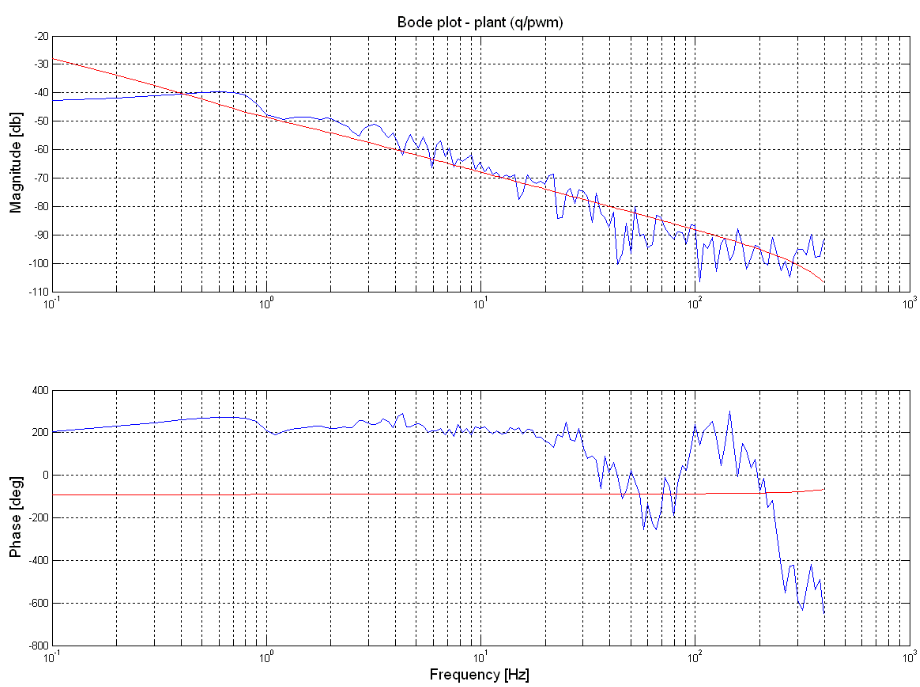

*Left hip z rotation*




*Left hip x rotation*




*Left hip y rotation*




*Left ankle y rotation*

## Appendix B – Controllers

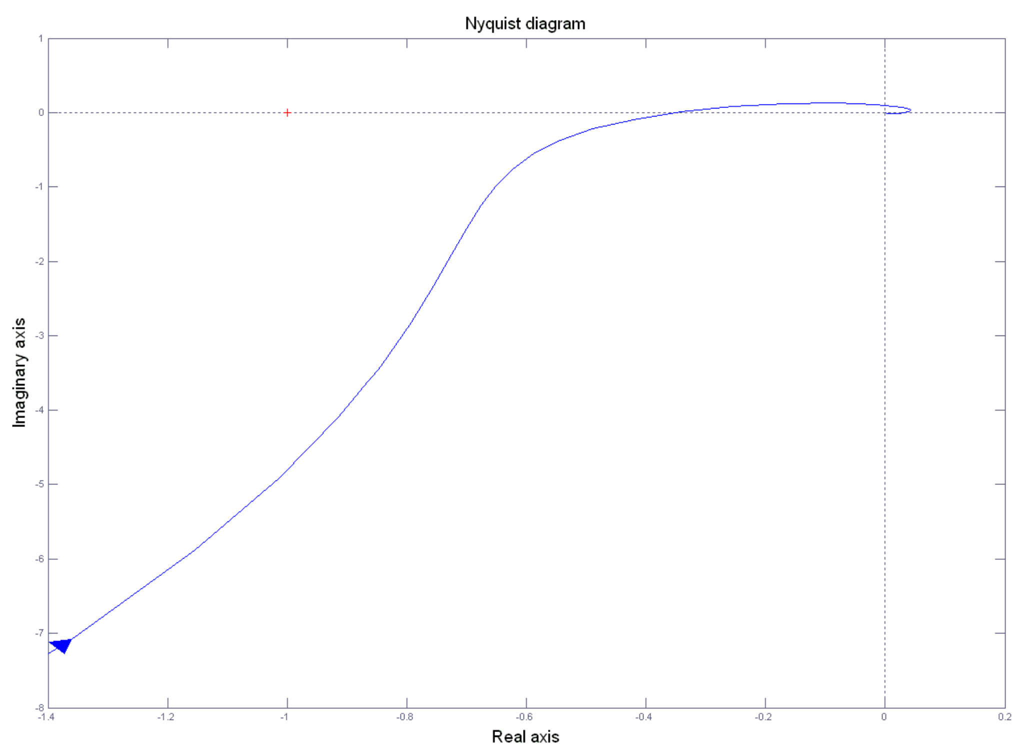
Right hip z rotation

- Proportional gain of 15
- Differential gain of 4.25
- Low-pass filter time constant of 0.001

This has the following results:

- Phase margin of 61 degrees
- Gain margin of 9.04 dB
- Maximum sensitivity of 5.10 dB
- All open loop poles have a magnitude smaller than 1
- 0 dB frequency of 262 rad/sec

Nyquist plot



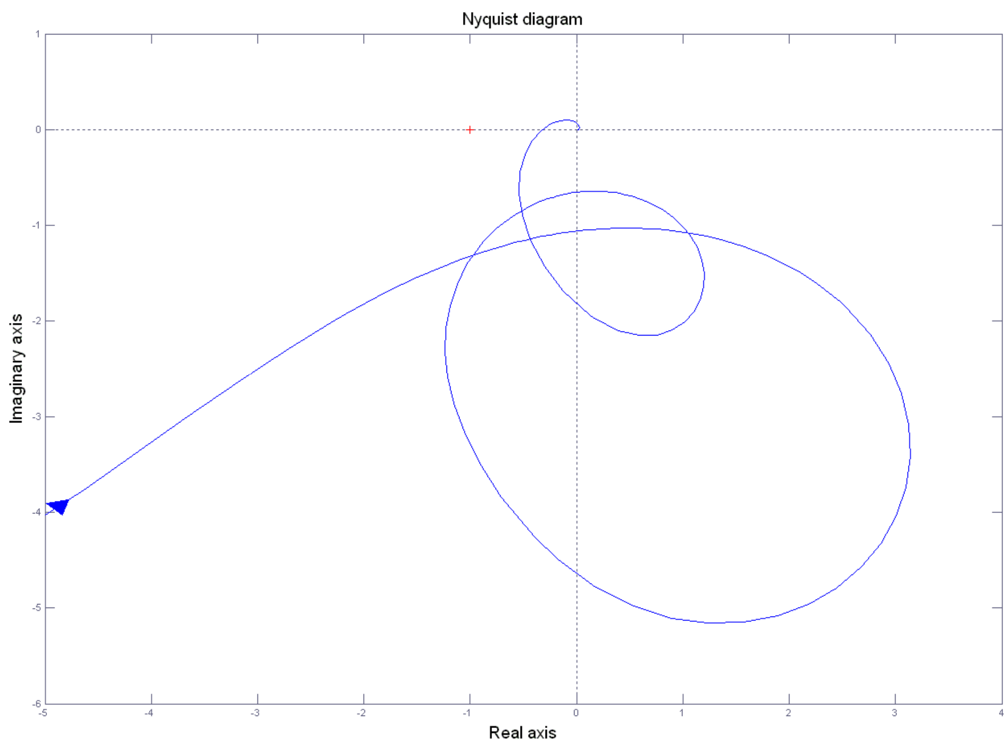
Right hip x

- Proportional gain of 900
- Differential gain of 14
- Low-pass filter time constant of 0.002

This has the following results:

- Phase margin of 60 degrees
- Gain margin of 9.81 dB
- Maximum sensitivity of 4.83 dB
- All open loop poles have a magnitude smaller than 1
- 0 dB frequency of 59.2 rad/sec

Nyquist plot



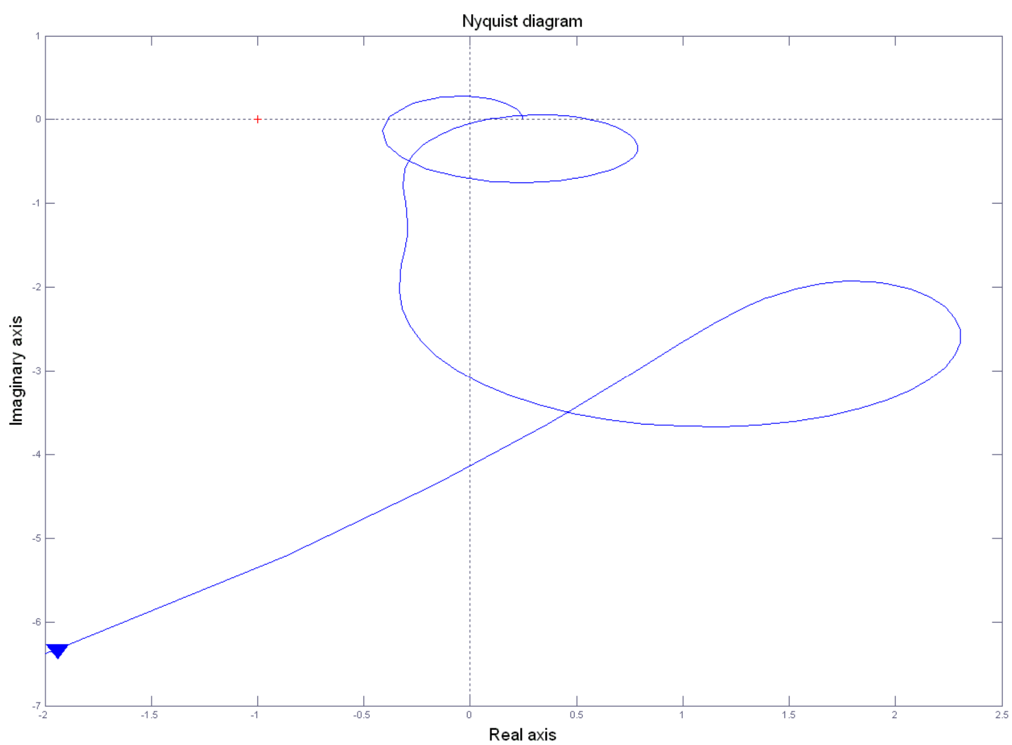
Right hip y rotation

- Proportional gain of 200
- Differential gain of 50
- Low-pass filter time constant of 0.002

This has the following results:

- Phase margin of 60 degrees
- Gain margin of 8.13 dB
- Maximum sensitivity of 4.51 dB
- All open loop poles have a magnitude smaller than 1
- 0 dB frequency of 68 rad/sec

Nyquist plot



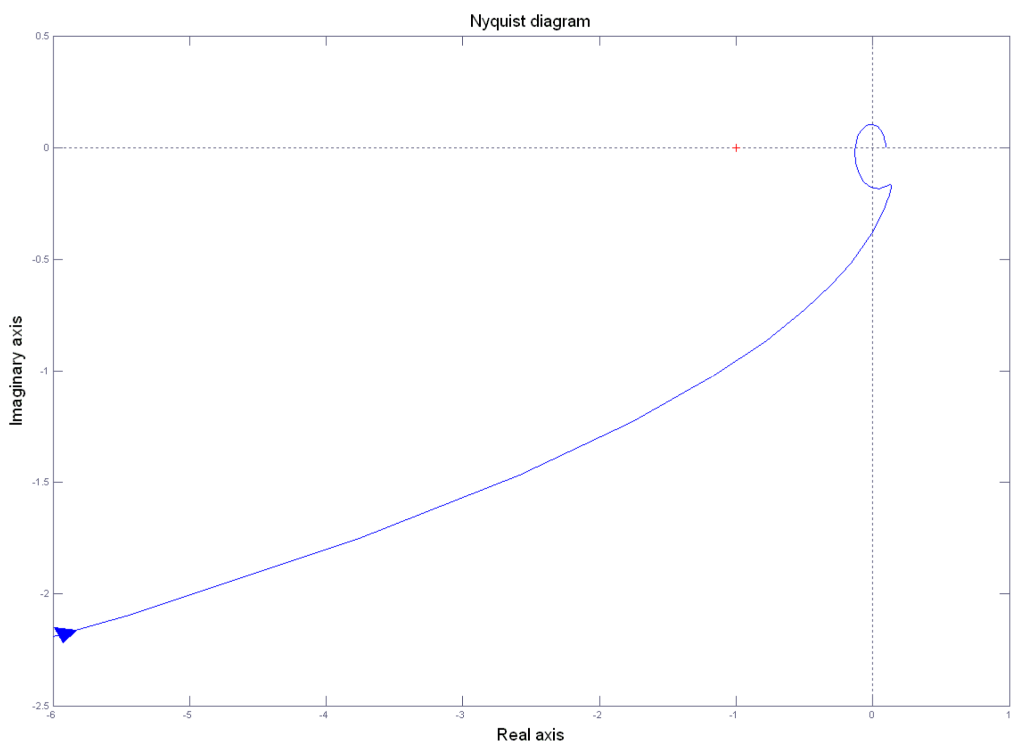
Right ankle y rotation

- Proportional gain of 500
- Differential gain of 10
- Low-pass filter time constant of 0.001

This has the following results:

- Phase margin of 52 degrees
- Gain margin of 17.7 dB
- Maximum sensitivity of 1.22 dB
- All open loop poles have a magnitude smaller than 1
- 0 dB frequency of 20.3 rad/sec

Nyquist plot



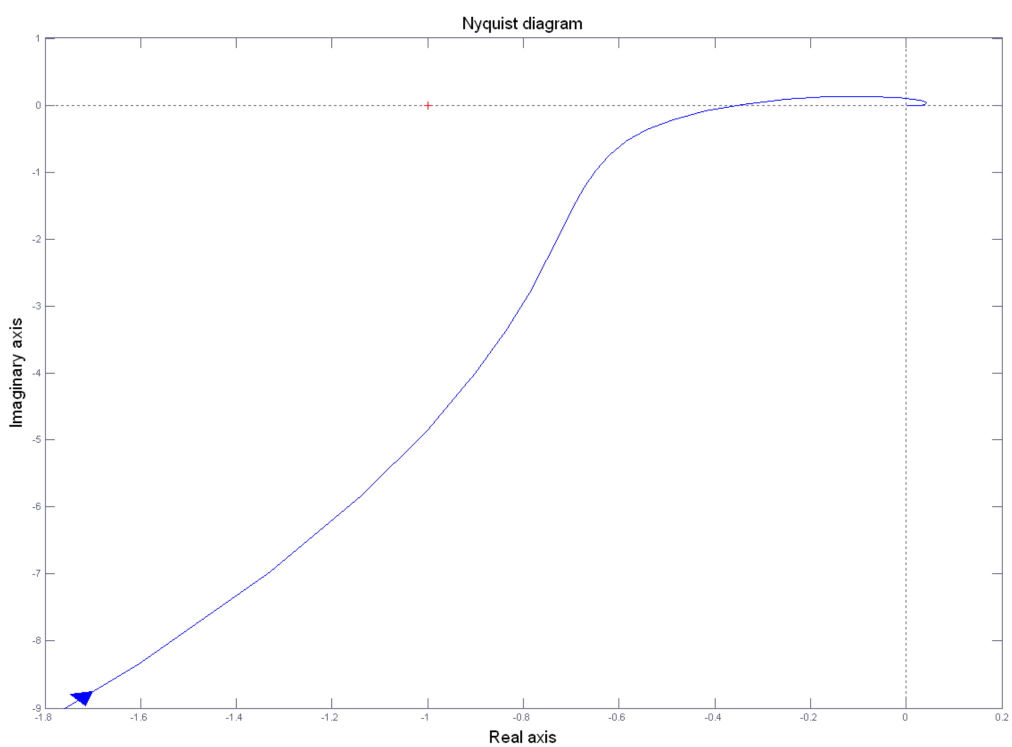
Left hip z rotation

- Proportional gain of 15
- Differential gain of 4.25
- Low-pass filter time constant of 0.001

This has the following results:

- Phase margin of 51 degrees
- Gain margin of 9.04 dB
- Maximum sensitivity of 5.08 dB
- All open loop poles have a magnitude smaller than 1
- 0 dB frequency of 259 rad/sec

Nyquist plot



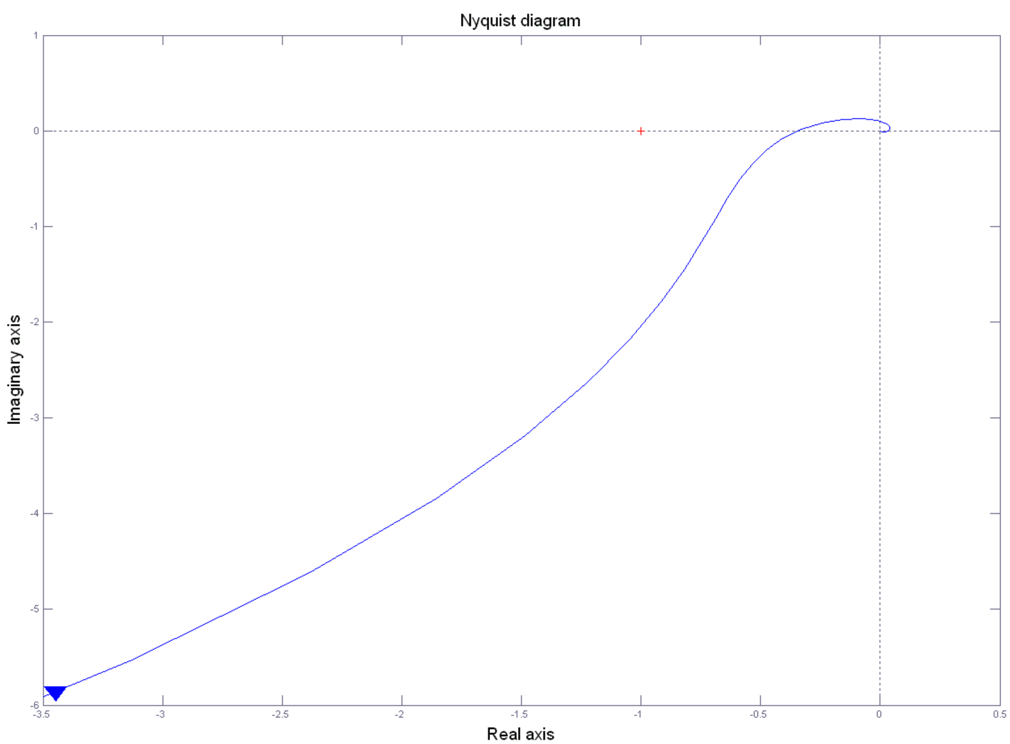
Left hip x rotation

- Proportional gain of 150
- Differential gain of 8
- Low-pass filter time constant of 0.001

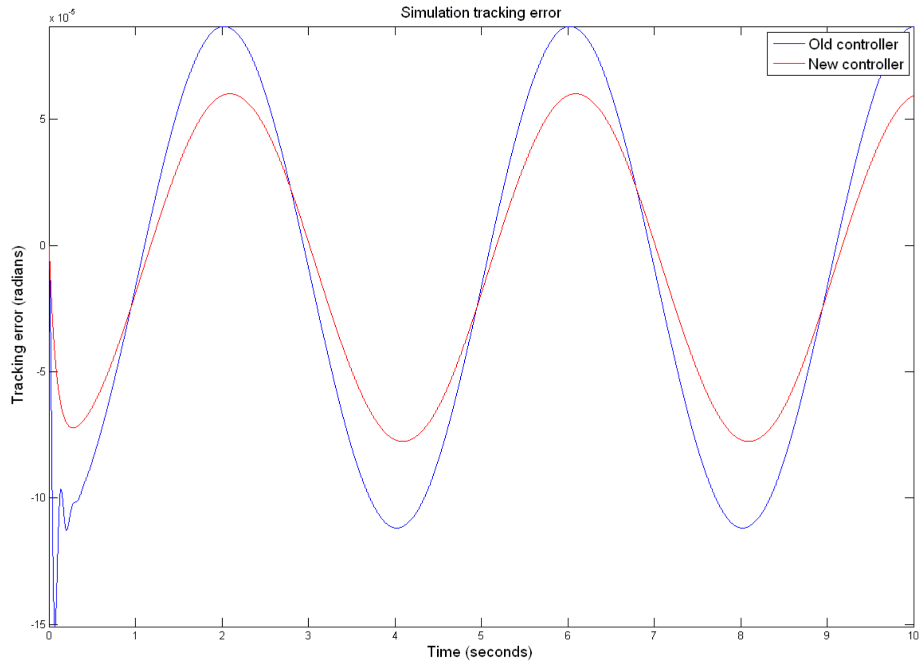
This has the following results:

- Phase margin of 49 degrees
- Gain margin of 9.29 dB
- Maximum sensitivity of 5.03 dB
- All open loop poles have a magnitude smaller than 1
- 0 dB frequency of 248 rad/sec

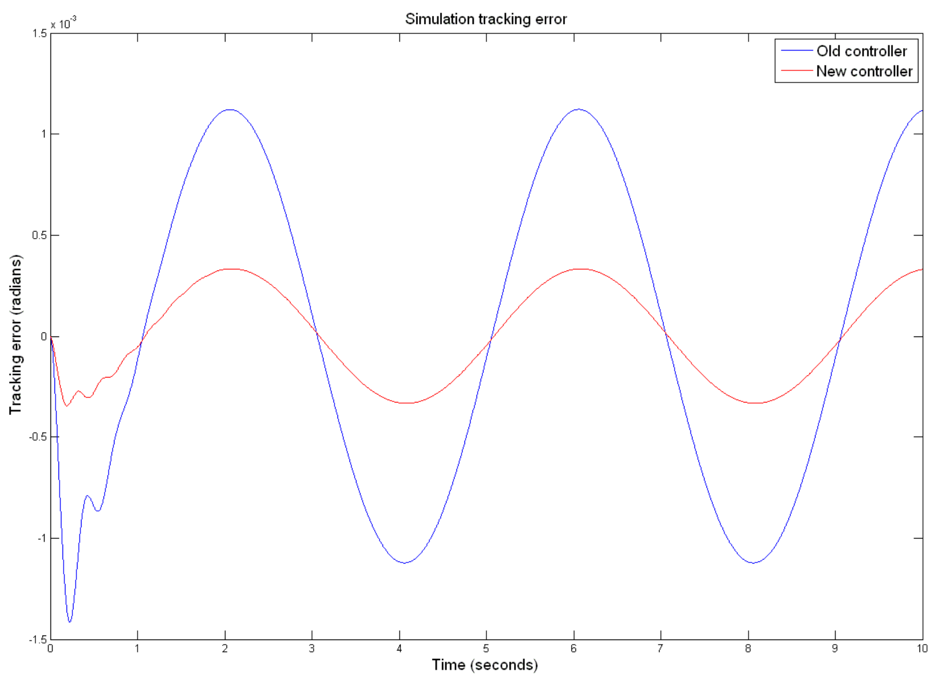
Nyquist plot



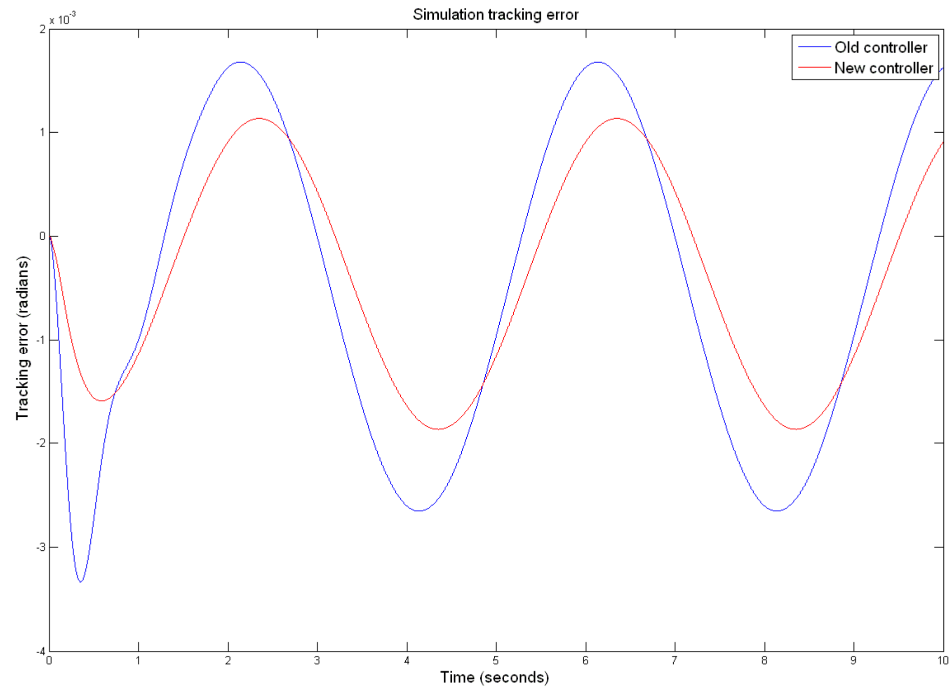
# Appendix C – Simulation results



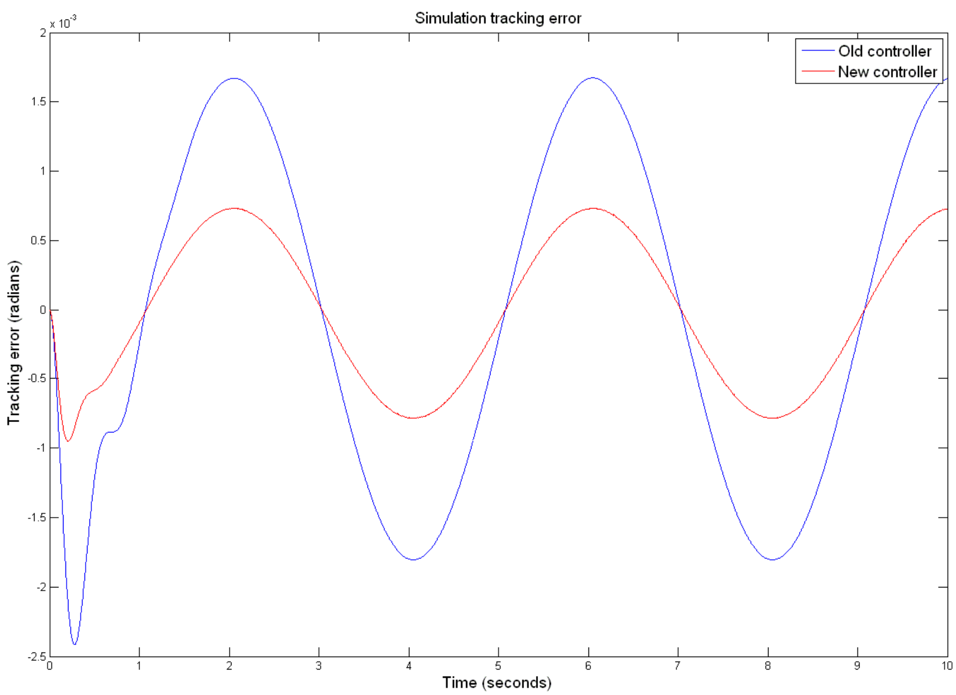

*Right hip z rotation simulated tracking error*



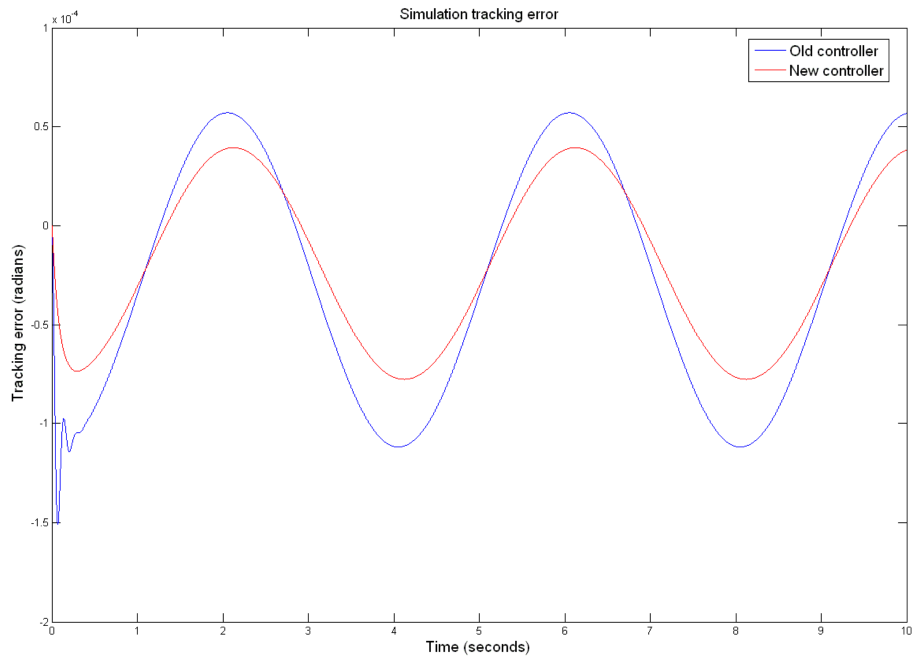

*Right hip x rotation simulated tracking error*



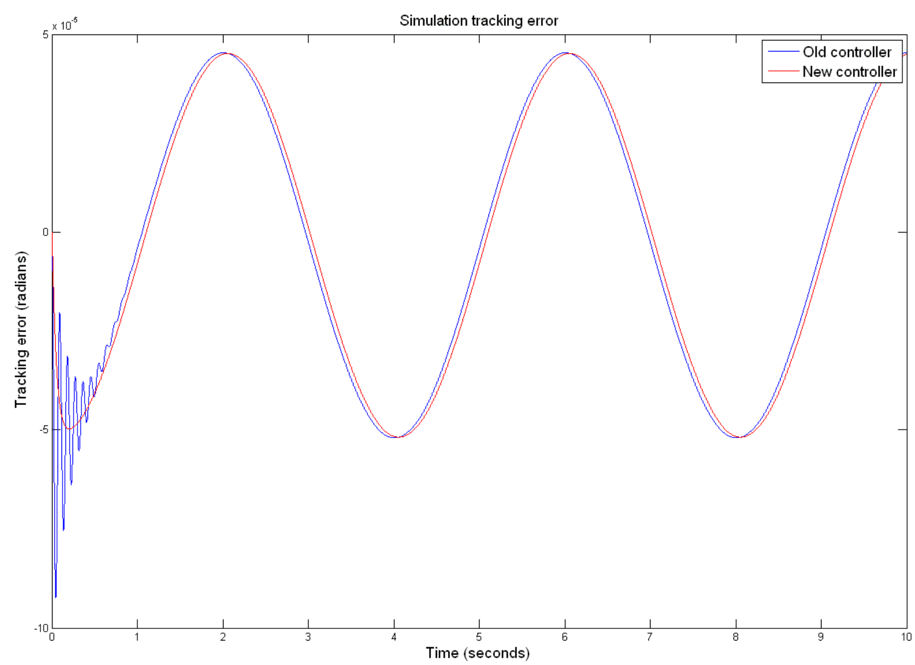

*Right hip y rotation simulated tracking error*




*Right ankle y rotation simulated tracking error*

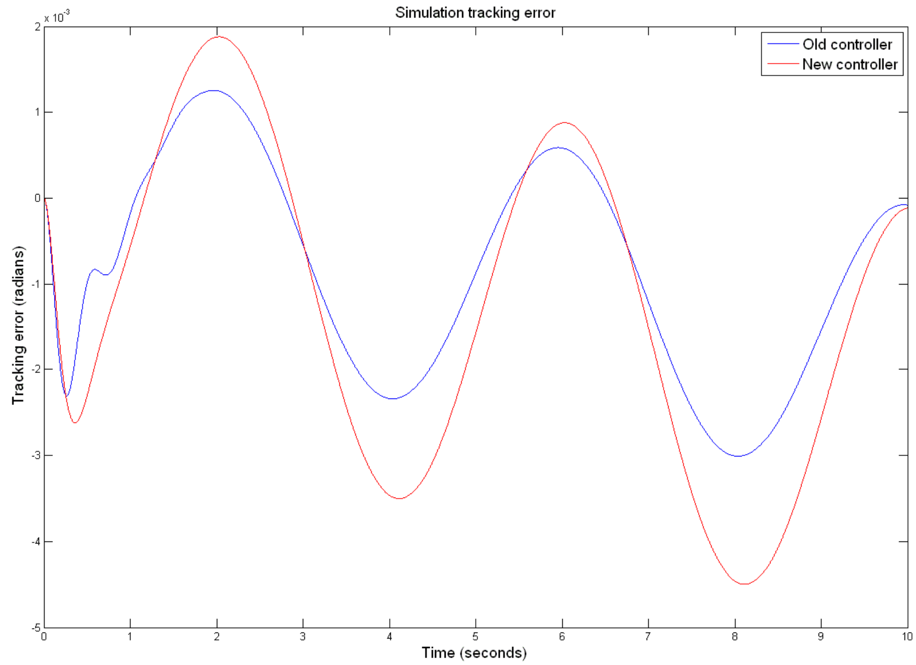



*Left hip z rotation simulated tracking error*



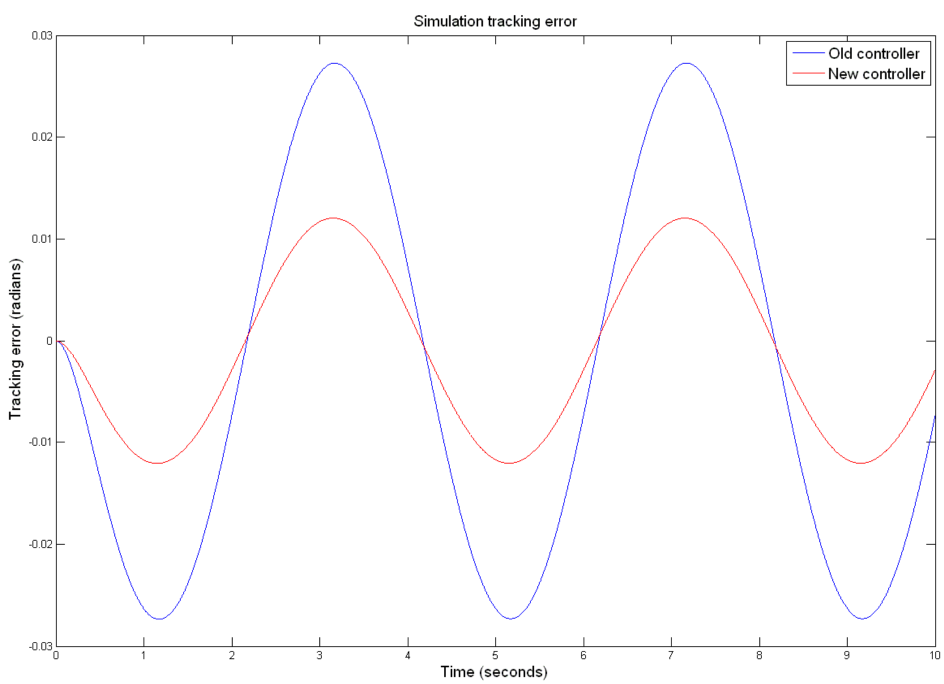

*Left hip x rotation simulated tracking error*

The simulation shows some strange behavior which was also encountered during measurements.




*Left hip y rotation simulated tracking error*

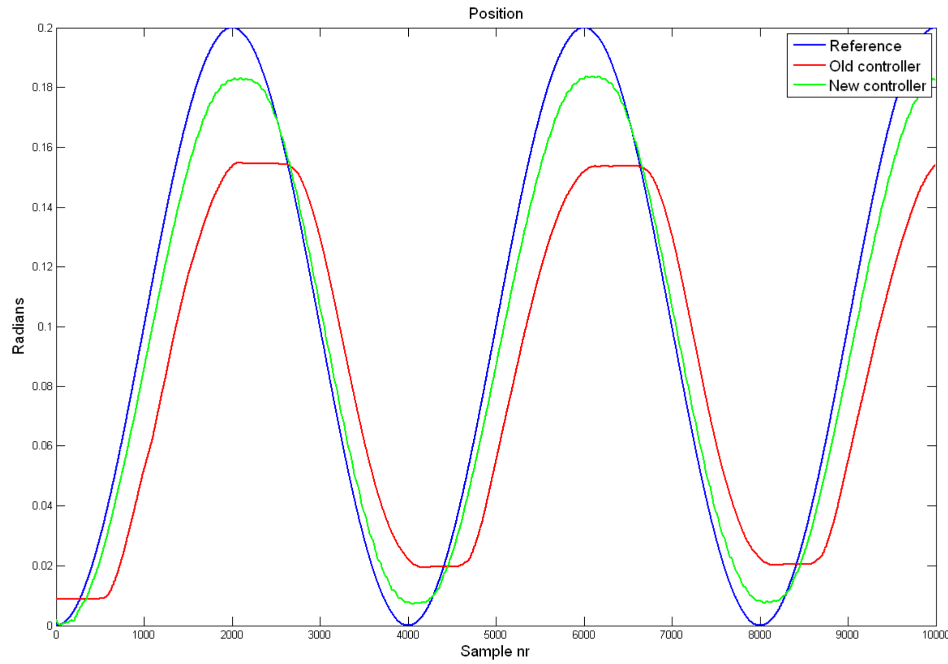
It can be seen that the simulated behavior is unstable, the maximum error increases over time.

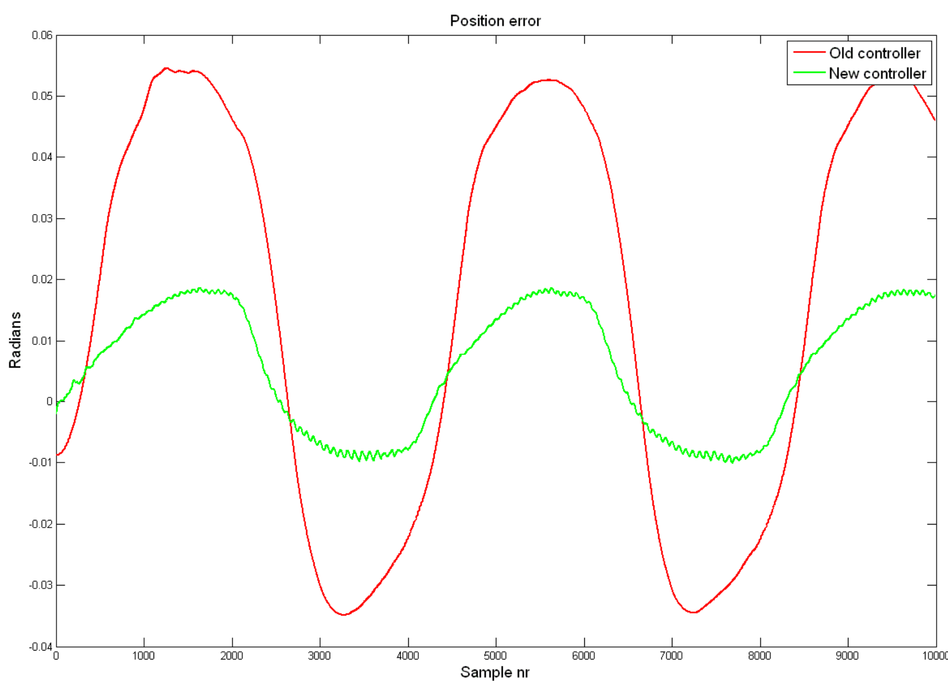



*Left ankle y rotation simulated tracking error*

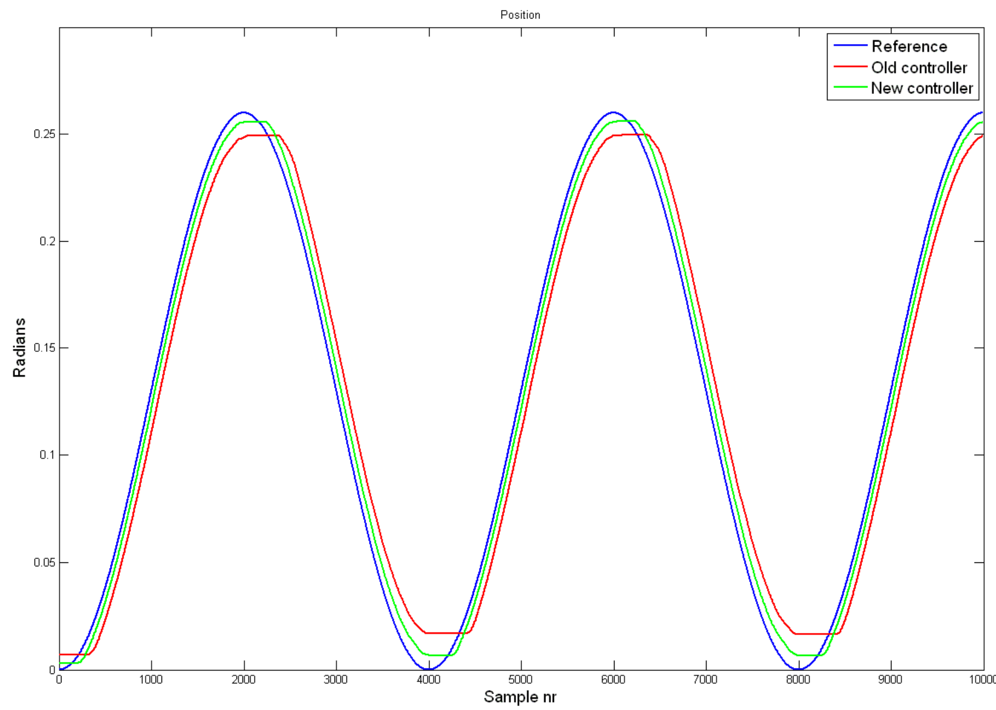
# Appendix D – Test results

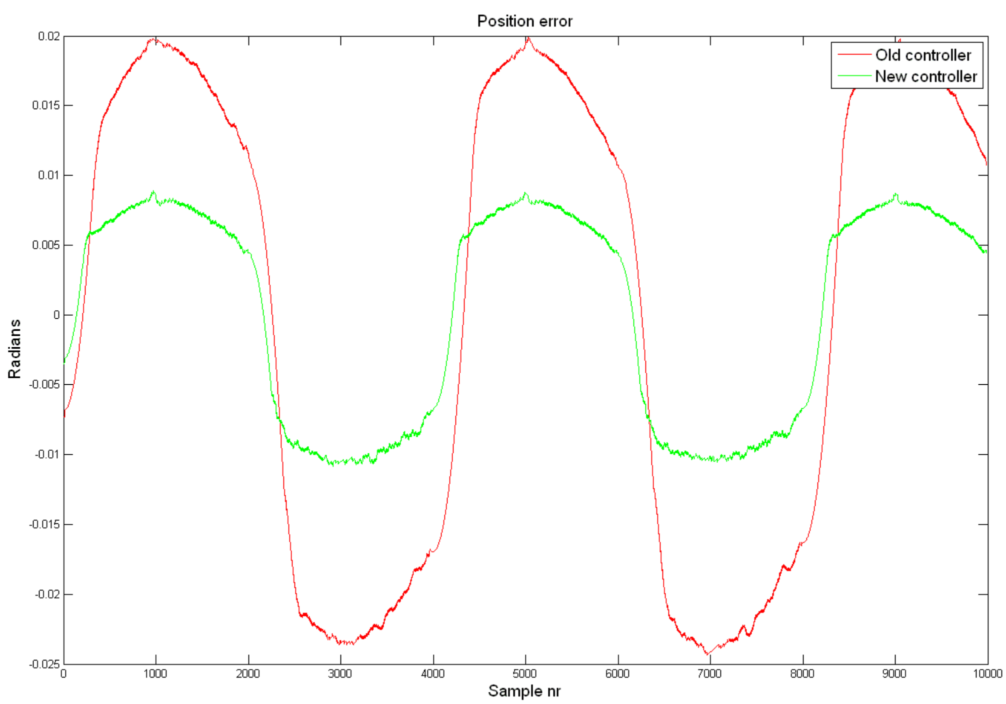
Right hip y rotation



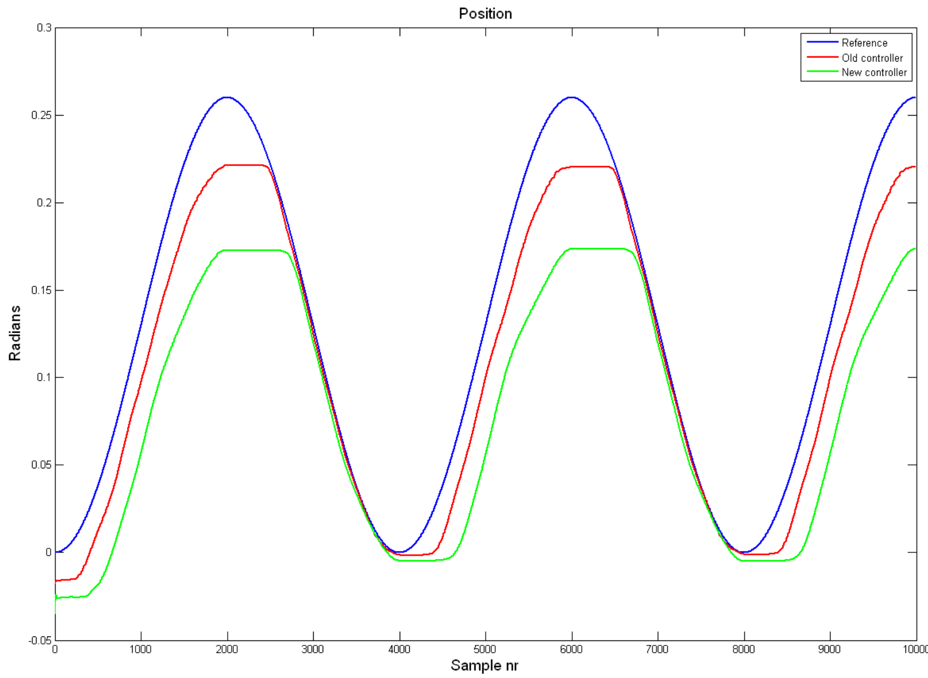
Right ankle y rotation

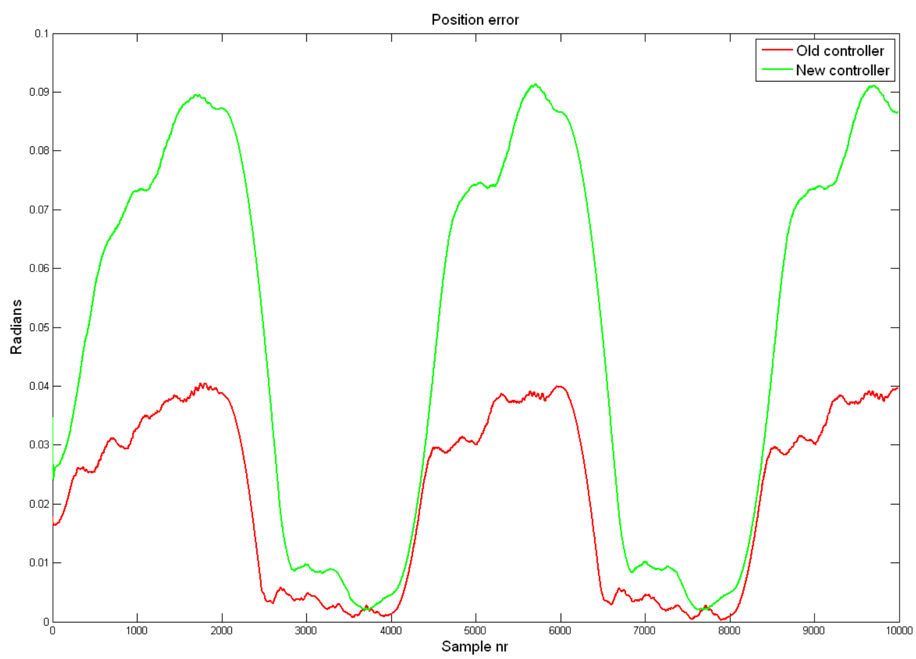


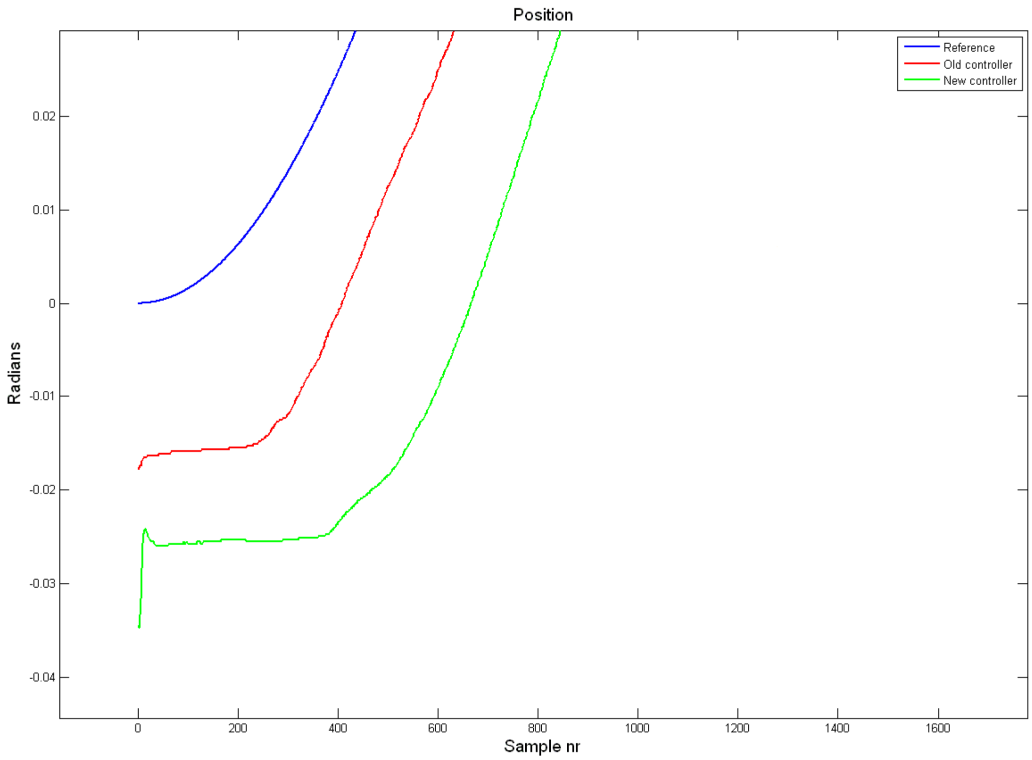



Left hip x rotation

Testing of this joint was cancelled at 40% gain, because the robot showed a strange jump in the start-up of the motion. An additional zoomed in figure is added where this phenomenon is visible.




Position
Reference
Old controller
New controller
0.02
0.01
0
-0.01
-0.02
-0.03
-0.04
Radians
0
200
400
600
800
1000
1200
1400
1600
Sample nr

## Appendix E – Matlab file

This is the Matlab-file for generating the sine signal used in FRF measurements.

```
%% % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
%
%       File for generating identification signals
%       using msinclip.m
%
%       (function for optimizing the energy spectrum of a signal)
%
%       Additional information:
%       http://dali.feld.cvut.cz/ucebna/matlab/toolbox/fdident/msinclip.html
%
%       2-10-2009
%       J. Scholtes
%       Matlab 2008b
%
%% % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
clear all; clc; close all;

% Example 1: energy distribution per frequency
freq_inc    = .25;
freq_start  = freq_inc;
freq_end    = 5;

freqv = freq_start:freq_inc:freq_end;       % frequency vector
ampv = ones(length(freqv),1);


tf =[];          % complex transfer function values at the given frequencies
gmod = 'ZOH';    % stepwise function will be designed (prepared for a D/A
converter),
                 % the number of samples will not be rounded up to the next
power of
                 % two, and the maximum overshoots between samples will not be
calculated
itno = 1500;     % maximum number of iteration cycles
ovs  = [2];      % minimum resolution of the peak factor calculation; fs >=
ovs*2*fmax
N = [];          % offers a direct control of the length of the time series
                 % used for crest factor calculation. If N is not given,
chosen to provide
                 % that each frequency be an integer multiple of fs/N.
cl0 = [];        % initial clipping level of the algorithm between (0,1).

[cx,crx] = msinclip(freqv,ampv,tf,gmod,itno,cl0);

% plot results:
ts = 1/1000;            % sample time
T0 = 1/freq_start;      % Total time of 1 chirp
nperiod = 1;            % number of periods
tend = nperiod*T0;      % end time
tvec = 0:ts:tend-ts;    % time vector
amplitude = 25;         % max amplitude
```

```
%(convert complex numbers to amplitude vector of the signal )
resp = zeros(size(tvec));
for q = 1:length(freqv);                        % length vector
    om = 2*pi*freqv(q);                         % freq omega
    cabs = abs(cx(q))/2;                        % absolute value cx per freq.
    cang = angle(cx(q));                        % phase value cx per freq.
    resp = resp+cabs*cos(om.*tvec+cang);        % freq domain to time domain
end
resp=resp/max(abs(resp));


resp=resp * amplitude * pi/180;

%% plot identification signal
close
figure(1);plot(tvec,resp*180/pi,'linewidth',2.0); grid on;
xlabel('Time [s]','fontsize',14);
ylabel('Input [deg]','fontsize',14);
title('Generated Input Signal')

%% plot power spectrum
Fs = 1/ts;       % sample frequency
NFFT = Fs*2;     % number of FFT points used to calculate the PSD
Y = fft(1000*resp,NFFT)/(length(tvec));
f = Fs/2*linspace(0,1,NFFT/2);
resp2 = 2*abs(Y(1:NFFT/2));

%% calculate velocity:

n = length(resp);
vel = [0, diff(resp)/ts];

%% save signals

ident_signal = dataset({resp.','Position'},{vel.','Velocity'});

resp=resp';
vel=vel';

resul_matr=[resp vel];

save -ascii -double input.txt resul_matr
```