GAIN-SCHEDULED PID FOR IMBALANCE COMPENSATION OF A MAGNETIC BEARING

Laleh Hosseini-Ravanbod, Dominikus Noll
Université Paul Sabatier, Institut de Mathématiques, Toulouse, France
lalehravanbod@yahoo.fr, noll@mip.ups-tlse.fr

Keywords: Scheduled controller for magnetic bearing, $H_\infty$ optimal decentralized PID controller, robust control, distance to instability, switching, hysteresis, interpolation

Abstract: Control of a magnetic bearing device is addressed by parameter varying control. Within the structure of decentralized PID controllers we compare linear interpolation and switching strategies with and without hysteresis. Piecewise LPV decentralized PID controllers are found to be an interesting alternative. Our method exploits the possibility to pre-compute for every parameter value an $H_\infty$ optimal decentralized PID controller, and to use this ideal model to construct practical scheduled controllers with an acceptable $H_\infty$ performance.

1 INTRODUCTION

Magnetic bearings (MB) increasingly become the choice for high-speed, high-performance rotating machinery because of their frictionless characteristics. They utilize a magnetic field generated by radially or axially placed electromagnets to generate the forces necessary to suspend and support a shaft without any contact with its environment. Thus, magnetic bearings are particularly useful in very high or very low temperature conditions where a lubrication-free environment is necessary. The advantages of magnetic bearings are primarily their very low power consumption and their very long maintenance-free life. Some applications where magnetic bearings offer distinct advantages are high speed turbo machinery, precision milling spindles, and combined attitude control and energy storage for spacecraft and satellites. A disadvantage of magnetic bearings is that they require continuous power input and active control to hold the load stable.

Active magnetic bearings (AMB) can support rotors without friction but require a sophisticated control system because specific performance requirements such as automatic balancing of the shaft, rejection of unwanted disturbances and vibration isolation are required. Many of the controllers proposed assume a linear time-invariant model, an assumption which is no longer accurate when the rotational speed varies.

Control techniques from linear robust control (Mohamed and Busch-Vishniac, 1995), $H_\infty$ loop shaping and $\mu$ synthesis (Lanzon and Tsiotras, 2005) as well as adaptive control methods have been used to attack this problem. Robust control is often overly conservative as it fails to account for the actual time variation of the rotor speed, which in addition is measurable.

Another approach is gain-scheduled $H_\infty$ controllers for linear parameter varying (LPV) systems based on LMI techniques (Tsiotras and Mason, 1996; Packard, 1994; Apkarian and Gahinet, 1996; Apkarian et al., 1995). Here the idea is to solve a series of standard $H_\infty$ problems at a pre-specified number of operating speeds. Using a single Lyapunov function to show stability and finite $L_2$-gain at these selected points, one guarantees that these properties will also hold for all operating speeds which are linear combinations of the selected speeds (interpolation). Unfortunately, this strategy is only valid if the controller is of the same order as the plant. Moreover, due to the choice of a single Lyapunov function this also tends to be conservative. In AMB systems there is strong interest to use small order controllers or other simple structures like PID.

High rotor speeds are gaining importance, and
DECENTRALIZED PID DESIGN

PID are still the controllers of choice due to consolidated hardware and software tools for design and hardware embedding, and the implication that more complex controllers may have. The drawback of PIDs in scheduling may be a significant loss of performance, and sometimes even worse, loss of internal stability. Here we present a new method to design scheduled decentralized PIDs for a parameter varying MB system, which allows to avoid these fallacies. The central idea of our approach is to precompute the \( H_{\infty} \)-optimal decentralized PID controller \( K^*(p) \) for every parameter value \( p \), and to use this ideal information to construct a parameter dependent decentralized PID \( K(p) \), which allows practical hardware embedding, and at the same time does not fall behind the ideal \( K^*(p) \) in closed-loop performance for more than an allowed level of \( \alpha \% \).

The paper is organized as follows. Section 3 represents the open-loop MB system. In section 4 the \( H_{\infty} \) performance channel is discussed, and section 5 gives the state-space form of the decentralized PID. Section 6 explains the role of the reference curve \( K^*(p) \) and explains the rationale of the robustification method. Re-centralizing plant and controller models of MB are explained in sections 6.1 and 6.2 and are needed to apply the LPV procedure in sections 7 and 8. This is carried out in section 8 by solving a mixed \( H_{\infty}/H_\infty \) program (11) based on the semi-structured stability radius (see Hinrichsen and Pritchard, 1986a; Hinrichsen and Pritchard, 1986b; Karow et al., 2010; Lawrence et al., 2000)). Experimental results are presented in section 9.

OPEN LOOP

In an AMB the axis of rotation does not coincide with the geometric axis of the rotor. The goal of control is to effect a rotation about the principal inertial axis to eliminate the radial centrifugal force, so that the principal axis of inertia becomes the center of mass of a cross-section of the rotor. The situation is shown schematically in Figure 1.
and $\phi = 0.05$. The system is therefore described by the joint system state $x_{sys} = [x_{mb}, x_{dist}]^T$ with dynamics

$$
\dot{x}_{sys} = A_{sys}(p)x_{sys} + B_{1sys}\delta + B_{2sys}u
$$

$$
y = C_{sys}x_{sys}
$$

where $A_{sys}(p) = \text{diag}(A_{mb}(p), A_{dist}(p)) \in \mathbb{R}^{8 \times 8}$, $B_{1sys} = [b_{0x1}, B_{dist}(p)]^T$, $B_{2sys} = [b_{mb}, 0_{2 \times 2}]$, $C_{sys} = [C_{mb}, C_{dist}]$. For more explanation on system modeling see (Lanzon and Tsiotras, 2005), (Tsiotras and Mason, 1996) and (Smith and Weldon, 1995). In the latter reference a more complete model is presented, in which $p$ is one of the states. This means that we expect $p$ to vary continuously in time.

4 PERFORMANCE INDEX

In our next step we have to define controlled outputs $z \in \mathbb{R}^4$ to assess the system performance. We use the control configuration shown in Figure 2. Following (Tsiotras and Mason, 1996), the controlled output $z$ regroups $u$ and $y$ with appropriate frequency weighing filters: $W_u = \text{diag}(W_{u1}, W_{u2})$ with $W_{u1} = W_{u2} = 0.01 \left(\frac{s+1500}{s+1000}\right)^2$ and $W_y = \text{diag}(0.5, 0.5)$ which up to a factor $10^{-4}$ are as in (Tsiotras and Mason, 1996). Altogether this adds 4 states to the 8 states of the open loop system. We add a reference signal $r \in \mathbb{R}^2$ for $y$ to the exogenous inputs, which leads to $w = (r_1, r_2, \delta) \in \mathbb{R}^3$. Altogether we obtain a parameter varying linear fractional transform (LFT)

$$
P(p) : \begin{bmatrix}
\dot{x} \\
z_{per}
\end{bmatrix} = \begin{bmatrix}
A(p) & B_1 \\
C_1 & D_{11} & D_{12} \\
C_2 & D_{21} & 0
\end{bmatrix}
\begin{bmatrix}
x \\
w_{per} \\
u
\end{bmatrix}
$$

with dimensions $x \in \mathbb{R}^{12}$, $z_{per} \in \mathbb{R}^4$, $y \in \mathbb{R}^2$, $w_{per} \in \mathbb{R}^3$ and $u \in \mathbb{R}^2$. We shall refer to $w_{per} \rightarrow z_{per}$ as the performance channel.

5 CONTROLLER PARAMETRIZATION

In this study we design a decentralized PID controller which depends on the scheduling parameter $p$. Recall that a SISO PID controller $K_{pid}(s) = d + r_i/s + r_d/(s + \tau)$ has the state space representation

$$
K_{pid} : \begin{bmatrix}
0 & 0 & r_i \\
-\tau & -\tau & r_d \\
1 & 1 & d
\end{bmatrix}.
$$

A parameter dependent state-space representation of the decentralized PID is therefore obtained as

$$
K(p) = \begin{bmatrix}
0 & 0 & 0 & 0 & r_i(p) & 0 \\
0 & -\tau(p) & 0 & 0 & r_d(p) & 0 \\
0 & 0 & 0 & 0 & 0 & r'_i(p) \\
0 & 0 & 0 & -\tau(p) & 0 & r'_d(p) \\
1 & 1 & 0 & 0 & d(p) & 0 \\
0 & 0 & 1 & 1 & 0 & d'(p)
\end{bmatrix},
$$

with 8 scheduling functions $r_i(p), \ldots, r'_d(p)$ to be determined. If we assume an affine parametrization, then we have 16 free parameters to determine.

6 RATIONALE

For every parameter value $p$ we consider the $H_\infty$-synthesis problem

$$
\text{minimize } ||T_{w_{per} \rightarrow z_{per}}(P(p), K)||_\infty
$$

subject to $K$ has decentralized (4)

$$
K \text{ stabilizes } P(p) \text{ internally}
$$

Let $K^*(p)$ be the solution of (5), which we compute by the Matlab function hinstruct. This furnishes 8 optimal parameter values $r_i^*(p), \ldots, d'^*(p)$. The curve $p \mapsto ||T_{w_{per} \rightarrow z_{per}}(P(p), K^*(p))||_\infty =: P^*(p)$ gives the best possible $H_\infty$ performance plotted over the interval $p \in [315, 1100]$. Clearly the parametrization $p \mapsto K^*(p)$ is not practical, and we need approximations of the mapping $K^*(\cdot)$ which can be stored conveniently. This leads to a trade-off between the quantity to be stored and the unavoidable loss of performance. In order to control the loss of performance we adopt the following convention. Fixing $\alpha > 0$, we call a parametrization $p \mapsto K(p)$ acceptable, if

(i) $K(p)$ has structure (4),

(ii) $K(p)$ stabilizes $P(p)$ internally for every $p$, and

(iii) $||T_{w_{per} \rightarrow z_{per}}(P(p), K(p))||_\infty \leq (1 + \alpha)||T_{w_{per} \rightarrow z_{per}}(P(p), K^*(p))||_\infty$ for every $p$. 

Figure 2: Block diagram of $H_\infty$ system.
To get a scheduling function \( K(\cdot) \) which needs as few elements to store (to embed) as possible, we discuss two approaches, which use either interpolation, or switching. For switching we identify subintervals \( I = [p_1, p_2] \subset \Pi \) as large as possible on which we can represent \( K(p) \) as an affine function without violating criteria (i) - (iii). Then we cover \( \Pi \) with as few as possible of these subintervals \( I_1, \ldots, I_N \). The rule to control \( P(p) \) is then by switching between these \( I_i \). The way to construct \( I \) together with an affine representation \( K(p) \) valid on \( I \) is given in the next section. A variation which uses interpolation is given in section 9.

**Remark.** We recall that condition (ii) is only necessary but not sufficient for stability of the switched or interpolated closed loop system. In general it is difficult to prove stability over the parameter domain if nothing is known about the parameter trajectory \( p(t) \). Sufficient conditions based on prior bounds \( |p(t)| \leq \gamma \) can be stated, but are generally hard to establish due to the inherent conservatism. In the same vein, condition (iii) should be understood as a worst case point of view. Namely, unlike in chess where it would be enough to know the best move \( K^*(p) \) in any given position \( p \), the situation here is more complicated, because the payoff may be different in different parameter regions, and the seemingly "best" move \( K^*(p) \) at \( p \) might lead to an unfortunate parameter trajectory in the future. (A similar situation would arise in chess as soon as e.g. winning with more material on the board would count more than winning with few pieces left. The notion of a "best" move would then have to be re-defined).

### 6.1 LPV model of Magnetic Bearing

For fixed \( p_0 \in \Pi \) we wish to find an interval \( I(p_0) \) containing \( p_0 \) as large as possible such that we have a simple controller parametrization \( K(p) \) valid in the sense of (i) - (iii) on \( I(p_0) \). In order to construct these intervals, it is helpful to change the parametrization and write \( p = p_0 + \Delta z \). The new parameter \( \Delta \) is now centered at 0 and more symmetric, \( |\Delta| \leq r \), and we wish to have a range of validity \( r \) as large as possible. We re-write the open-loop system accordingly. For example, for the third state in (3) we have

\[
\dot{x}_3 = -\frac{4c_2}{m} x_1 - \frac{m_4}{J_r} x_4 + \frac{2c_1}{m} x_5 + \frac{(p_0(1+\delta))/J_r}{m} x_4 + \frac{2c_1}{m} x_5.
\]

Introducing an auxiliary input \( w_{\text{rob}} \) and output \( z_{\text{rob}} \) via \( w_{\text{rob}} = -\frac{J_r}{J_r} p_0 x_4 \) and \( z_{\text{rob}} = \delta \cdot w_{\text{rob}}, \) we obtain

\[
\dot{x}_3 = -\frac{4c_2}{m} x_1 - \frac{m_4}{J_r} x_4 + \frac{2c_1}{m} x_5 + z_{\text{rob}}.
\]

Repeating the same thing for all states leads to a standard representation of \( P(p) \), \( p = p_0(1+\delta) \), as an uncertain system:

\[
\begin{bmatrix}
\dot{x} \\
\dot{z}_{\text{rob}} \\
y
\end{bmatrix} = \begin{bmatrix}
A_{\text{sys}}(p_0) & B & B_{1,\text{sys}} & B_{2,\text{sys}} \\
C_{2,\text{sys}} & 0 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
x \\
w_{\text{rob}} \\
d \\
u
\end{bmatrix}
\]

where \( z_{\text{rob}} \in \mathbb{R}^4 \), \( w_{\text{rob}} \in \mathbb{R}^4 \), \( w_{\text{rob}} = \Delta z_{\text{rob}} \) with \( \Delta \in \mathbb{R}^4 \) and:

\[
B = \begin{bmatrix}
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix},
\]

\[
C = \begin{bmatrix}
0 & 0 & 0 & -\frac{J_r}{J_r} \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}
\times p_0.
\]

### 6.2 LPV model of decentralized PID controller

We repeat the same procedure for the controller parametrization (4). We write each of the 8 scheduling functions as \( r_i(p) = r_{0i} + \Delta \cdot r_{1i} \), and so on until \( d_i(p) = d_{0i} + \Delta \cdot d_{1i} \), with 16 parameters \( r_{0i}, \ldots, d_{1i} \) to be determined for each \( I(p_0) \). Notice that \( r_{0i} = r_i(p_0) \) depends on the choice of \( p_0 \), and similarly for the other scheduling functions. It is helpful to denote this controller as \( K(p_0, \delta) = K(p_0) + \delta \cdot dK(p_0) \), even though it still has the form (4).

| \( c_1 \) | 1.9715e3 Wb | \( N \) | 400 |
|\hline\( c_2 \) | 325.047 Wb/m | \( J_\alpha \) | 0.0136 Kg.m² |
|\hline\( d_1 \) | 2.1001 e3 Ω Wb/H m | \( J_r \) | 0.333 Kg.m² |
|\hline\( d_2 \) | 7.9804 e3 Ω Wb/H m | \( m \) | 19.7041 Kg |

Table 1: System constants (Tsiotras and Mason, 1996)
Having symmetrized the parameter, we get a similar uncertain block $\Delta_2 = \delta \cdot I_8$ with $\delta$ repeated 8 times, 4 times for each PID. Altogether this leads to the scheduled controller structure shown in Figure 3.

\[
\begin{bmatrix}
\dot{\xi}_1 \\
\dot{\xi}_2 \\
\vdots \\
\dot{\xi}_8 \\
u
\end{bmatrix} =
\begin{bmatrix}
0 & 0 & 1 & 0 & 0 & 0 & 1 & r_{d1} \\
0 & -r_0 & 0 & 1 & 1 & 0 & 0 & r_{d2} \\
0 & -r_1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & r_{d3} \\
0 & 0 & 0 & 0 & 0 & d_0 & 0 & d_0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
\xi_1 \\
\xi_2 \\
\vdots \\
\xi_8 \\
u
\end{bmatrix},
\]

where $\Delta_2 = \delta \cdot I_8$. We get a similar block for the second PID with primed parameters in (4). This leads to $w_{rob} = \Delta_2 \xi_{rob}$, where $\Delta_2 = \text{diag}(\Delta_1, \Delta_2) = \delta I_8$ for $K$. We then apply a standard procedure to the whole LFT which gives us what is on the right of figure 3. The controller is now independent of $\Delta$ and gathers the 16 unknown coefficients of the scheduling functions, or rather, the free parameters in (4). The uncertain block is of size $\Delta = \text{diag}(\Delta_1, \Delta_2) = \delta \cdot I_{12}$.

\section{Robustness Index}

The constellation on the right of Figure 3 corresponds to the LFT

\[
P_{rob}(p_0) : \begin{bmatrix}
\dot{x} \\
y_{rob}
\end{bmatrix} = \begin{bmatrix}
A(p_0) & B_0 & B_1 \\
C_0 & D_{00} & D_{02} \\
C_2 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
x \\
y_{rob} \\
u
\end{bmatrix},
\]

\[
w_{rob} = \Delta \xi_{rob}, \quad u = K(y).
\]

The matrix dimensions are $K(p_0) \in \mathbb{R}^{14 \times 14}$, $A(p_0) \in \mathbb{R}^{12 \times 12}$, $\Delta \in \mathbb{R}^{12 \times 12}$, etc. Notice that $\tilde{K}(p_0)$ carries the same information as $K(p_0, \delta) = K(p_0) + \delta \cdot dK(p_0)$, that is, it regroups the 16 unknown coefficients of $K(p_0)$ and $dK(p_0)$ which we want to compute. This is now a known situation in robust respectively LPV control. If $K$ in (7) was not structured, we could for instance use mu-tools to design full-order parametric robust controller of size $14 \times 14$ which makes the interval of robustness $[\delta]$ as large as possible. Similarly, the LPV procedure of Scherer (Scherer, 2003) would lead to a full order solution, (see also the monograph (Chesi et al., 2009) for the robustness approach to uncertain systems). The necessity to satisfy the structural constraint (7) complicates matters, and we have to take recourse to a heuristic method.

We concentrate on stabilizing $K(p, \delta)$ respectively $\tilde{K}(p)$ on as large an interval $I(p)$ as possible, so we consider (8) without performance channel. The closed-loop system matrix is then

\[
A(\tilde{K}, \delta) = A + B_2 K C_2 + (B_0 + B_2 K D_{20}) \Delta \times \ldots \times (I - D_{00} \Delta - D_{02} K D_{20} \Delta)^{-1}(C_0 + D_{02} K C_2)\]

we re-write as

\[
A(\tilde{K}, \delta) = \tilde{A} + \tilde{B} \Delta(1 - \tilde{D} \Delta)^{-1} \tilde{C}
\]

with $\tilde{A} = A + B_2 \tilde{K} C_2$, etc. Our goal is to guarantee stability of this matrix for as large a range $[\delta]$ as possible. We interprete (10) as the semi-structured complex stability radius $r_c(A + B_2 \tilde{K} C_2)$ of the nominal closed-loop matrix $\tilde{A} = A + B_2 \tilde{K} C_2$. It is well-known that

\[
r_c^{-1}(\tilde{A}) = \|C(1 - \tilde{A})^{-1} \tilde{B} + \tilde{D} \|_\infty,
\]

so that computing the stability radius of $A + B_2 \tilde{K} C_2$ amounts to computing an $H_{\infty}$-norm. Altogether, we have

\[
r_c^{-1}(\tilde{A}) = \|T_{rob}^{-1}(P_{rob}(p_0), \tilde{K})\|_\infty,
\]

where $P_{rob}(p_0)$ is the plant in (8).

\section{Optimization Program}

For every fixed $p_0$ we now compute the solution $K_{rob}(p_0)$ of the following mixed $H_{\infty}/H_{\infty}$ optimization program:

\[
\begin{align*}
\text{minimize} & \quad R(K) = \|T_{rob}^{-1}(P_{rob}(p_0), \tilde{K})\|_\infty \\
\text{subject to} & \quad P(K) = \|T_{per}^{-1}(P(p_0), K)\|_\infty \leq (1 + \beta)P^*(p_0)
\end{align*}
\]

where $P^*(p_0)$ is the nominal performance at $p_0$, that is, $P^*(p_0) = \|T_{per}^{-1}(P(p_0), K^*(p_0))\|_\infty$. This program presents a trade-off between performance and
robustness in the sense of \( r_C \). Namely, as we know, the best possible performance at \( p_0 \) is obtained by \( K^*(p_0) \), which corresponds to choosing \( r_0 = r^*_0(p_0) \), etc. and \( \delta = 0 \). In (11) we accept a loss of 100 \( \% \) performance over the nominal value \( P^*(p_0) \) and use this freedom to buy some additional robustness in the sense of \( r_C \), hoping that this will lead to a controller \( K_{rob}(p_0) \) with as large an interval of validity \( I(p_0) \) as possible. Clearly, in order to respect rule (iii), be to choose \( \beta < \alpha \), and in our experiments we use \( \beta = \alpha/2 \). For more information on this type of trade-off between performance and robustness see (Hosseini-Ravanbod et al., 2011b; Hosseini-Ravanbod et al., 2011a).

**Algorithm 1.** Algorithm to compute parametrized PID with switching

**Parameters:** \( \alpha > 0, 0 < \beta < \alpha \).

1. Pre-compute approximation of optimal curve \( K^*(p) \) using `hinfsyn).
2. For a sufficient number of parameters \( p \) solve mixed \( H_\infty/H_\omega \) program (11). The solution curve is \( K_{rob}(p) \).
3. For every \( p \) find interval of validity \( I(p) \) of \( K_{rob}(p) \) using conditions (i) – (iii).
4. Remove a small portion on each side of \( I(p) \) and call the shrunk interval \( I^*(p) \).
5. Select minimum number of \( I^*(p) \), \( v = 1, \ldots, N \), covering \( \Pi \). This means the intervals \( I(p_v) \) cover \( \Pi \) with some slight overlap.

The gain scheduling function is now as follows. Let \( v(p) \) be such that \( p \in I(p_v) \) and define the control law as \( K(p) = K_{rob}(p_{v(p)}) \). If there are several \( v \) with \( p \in I(p_v) \), then use the hysteresis rule. That is, stay on the interval in which the trajectory moves, and only jump on a new interval when the boundary of the first is reached. This means that for a value \( p \) in the intersection \( I(p_v) \cap I(p_{v'}) \) the actual control law changes depending whether one arrives from the right or from the left.

**9 EXPERIMENTAL RESULTS**

The measured parameter \( p \) varies in \( \Pi = [315, 1100] \) rad/sec. We use `hinfsyn` at 51 equidistant points in \( \Pi \) to compute the optimal decentralized PID control curve \( K^*(p) \).

For each of the 51 values \( p_0 \) we use program (11) with \( \beta = \alpha/2 \) to compute 51 controllers \( K_{rob}(p) \), satisfying (i) – (iii) in tandem with their intervals of validity \( I(p) \).

First study:
Our first test uses \( \alpha = 0.38 \) which allows to cover \( \Pi \) with only two intervals \( I(315) \) and \( I(1100) \). As one can see in Figure 4, the switched controller (continuous line) maintains stability over the entire \( \Pi \), but interpolating \( K^*(315) \) and \( K^*(1100) \) leads to a loss of stability (broken line). This indicates that interpolation of controllers is more delicate to arrange for than switching.

Second study:
Our second test uses \( \alpha = 0.1 \), which leads to more realistic results. We seek a switching controller and compute intervals \( I(p) \) according to algorithm 1 for 51 equidistant values \( p \in \Pi \).

![Figure 4](image) **Figure 4:** Interpolating linearly between the two controllers \( K^*(315) \) and \( K^*(1100) \) (dashed line) fails due to a loss of stability, while switching using \( K_{rob}(315) \) and \( K_{rob}(1100) \) holds the stability (solid line).

![Figure 5](image) **Figure 5:** (a) and (c): Validity regions \( I(p) \) plotted against \( p \). This allows to read off the intervals needed to cover \( \Pi \). (b) and (d) the final performance obtained for each case. Figure 5 (a) shows the validity region for each of the 51 controllers \( K_{rob}(p) \), that is, the values \( p^* \) for which controller \( K(p) \) satisfies (i) – (iii). For each...
of the 51 controllers $K_{rob}(p)$ the zone where (i) - (iii) hold corresponds to a horizontal array marked by $. As can be seen, by following the thick line, we can find a way covering all the variation of $p$. This requires only two controllers $K_{rob}(707.5)$ and $K_{rob}(1084.3)$ with switching at $p = 723.2$ rad/s (controllers number 26 and 50).

If we insist on a non-negligible overlap between the intervals $I(p)$, using the $I^2(p)$ as in the algorithm, then Figure 5 (c) shows that we need three controllers $K_{rob}(707.5)$, $K_{rob}(801.7)$ and $K_{rob}(1084.3)$, to cover $\Pi$ (controllers number 26, 32 and 50). The same Figures 5 (b) and (d) show the performance obtained in each case.

Third study: Our third test still uses $\alpha = 0.1$, but now we interpolate optimal controllers $K^*(p)$. Start at the left end $p_1 = 315$. Having found $p_k$, we examine for $p > p_k$ the closed-loop performance curve obtained by the controller $K_{rob}(p_k, p)$ interpolating between $K^*(p_k)$ and $K^*(p)$. As $p$ increases, this curve eventually hits the upper limit curve $(1 + \alpha_p)(K^*(p'))$ at some $p'$ between $p_k$ and $p$. We put $p_{k+1} = p$ and continue until the right end point is reached. A similar procedure starts at the right end point and moves downward.

Figure 6 gives the results of this algorithm in our study. Upwards interpolation needs 3 controllers $K^*(p)$ at $p = 315, 903.7, 1100$, while downwards interpolation requires more, namely $K^*(p)$ at $p = 1100, 1037.7, 931.7, 892.2, 817.9, 790.9, 707.5, 644.65, 315$.

Figure 7 shows a simulation in closed-loop where the scheduling function $K(p)$ uses three robust controllers $K_{rob}(p)$, and where $p(t)$ increases within 1.2 sec from 720 to 780 and then decreases back to 710.

**10 CONCLUSION**

Several methods to compute a parameter varying decentralized PID for a magnetic bearing device were compared. Performance was measured in the $H_\infty$ norm, and the curve $K^*(p)$ of optimal $H_\infty$-controllers was taken as a reference to assess the performance of the different parameterizations $K(p)$. If parameterizations $K(p)$ with a maximum loss of 10% over $K^*(p)$ were allowed, switching between piecewise affine controllers on subintervals was found to perform best, but needs solving a mixed $H_\infty$/$H_\infty$ synthesis program. Interpolation based on computing various $K^*(p)$ was an interesting alternative, even though it was observed that interpolation seems to have a stronger tendency to lose stability and important dependence at the beginning point. While the switching technique covers over to 2D parameter sets, there is no obvious way to extend interpolation into two dimensions.

**ACKNOWLEDGEMENT**

This work was supported by research grants Technicom from Fondation d’Entreprise EADS, and Survol from Fondation de Recherche pour l’Aéronautique et l’Espace (FNRAE).
REFERENCES


