Proceedings of the
47th IEEE Conference on Decision and Control
Cancun, Mexico, Dec. 9-11, 2008

Extension and Application of a Novel Hardware-in-the-Loop Simulator Design Methodology

Monte MacDiarmid, Marko Bacic, Ronald Daniel

Abstract— **This paper builds on previous work on optimal methodologies for the design of numerical subsystems for non-linear, uncertain hardware-in-the-loop (HWIL) simulators. Firstly, three important extensions to the existing methods are presented: the limitation to SISO systems is removed, allowing full MIMO design; a method for the design of an important and previously neglected component of the numerical system is described; and finally measurement noise and other unstructured uncertainty is tackled more rigorously and explicitly than previously. Secondly, as a case study the extended method is used to design a numerical system for a real HWIL simulator, and the results are shown to outperform those produced using classical design methods. The system used in the case study is a high-performance simulator for small aerodynamic objects that is in the final stages of development.**

I. Introduction

Hardware-in-the-loop (HWIL) simulation is a long-established[1] technique used to produce an estimate of the behaviour of a system, when only a subset of the hardware of a system is physically present. A HWIL simulator operates by replacing parts of a system that are well understood, or excessively expensive or awkward, by a numerical system that is interfaced to the hardware that is present through a set of interface transducers. In the literature, two philosophically similar yet practically distinct classes of HWIL simulators can be found. Examples of the first typically use HWIL simulation to verify the real-time behaviour of embedded computational devices (e.g. engine ECUs) using a model of the real plant. By contrast, examples of the second class of simulators utilise a piece of real physical hardware (e.g. a quarter-car suspension assembly), and require significant power transfers across the interface domain. It is this second class of simulator that is of particular interest to the control engineer, and which is the subject of this work. Example applications of this type of simulation can be found in [2], [3], [4], [5], [6], [7], [8], [9]

Previous work by the authors [10] has investigated the problem of designing the numerical part of a HWIL simulation. This problem was shown to present interesting design challenges to the control engineer. Firstly, the systems under simulation are always uncertain by definition, and are often non-linear. Secondly, common design goals such as robust stability lose meaning, as simulations can be assumed to be always of finite duration. Finally, the design of an optimal numerical simulation can almost be framed as a tracking problem, with one important complication; specifically, that the desired trajectory to be tracked (i.e. the behaviour of the real system) varies with the same underlying uncertainty as the behaviour of the plant to be controlled (i.e. the simulator, including the uncertain real hardware).

In [10], a design method was developed able to produce efficient, optimised numerical subsystems for general, non-linear HWIL simulators. This paper extends this method by removing the existing limitation to SISO systems, discussing the design of a previously neglected numerical component, and dealing with measurement noise and similar unstructured uncertainty more directly. The method is then applied to an interesting case study.

The paper proceeds as follows. Section II describes the background to the paper, and presents a summary of the method previously developed. Section III discusses the extensions to the method. Section IV describes the example application. Section V compares the results of the new and classical design methods in the example application. Finally, section VI concludes the paper.

II. Background and Previous Work

The work presented in this paper is a natural extension of previous work by the authors on designing optimal numerical subsystems for non-linear, uncertain, SISO HWIL simulators. This section summarises this previous work. Refer to reference [10] for full details.

A. Classical HWIL Design

To begin, a model of the SISO HWIL simulation problem is required. Figure 1 shows general diagrams of the real system to be simulated, and a HWIL simulator. All systems $\mathcal{W}$ represent non-linear, discrete time, SISO dynamical systems, that map an input sequence $\mathbf{a} = \begin{bmatrix} a[0] & \dots & a[N] \end{bmatrix}^T$ to an output sequence $\mathbf{b} = \begin{bmatrix} b[0] & \dots & b[N] \end{bmatrix}^T$. It is assumed that simulations are of interest over a finite time horizon N.

The real system has been broken down into subsystems $\mathcal{W}_U, \mathcal{W}_K, \mathcal{W}_O$. The subsystems are connected via scalars q_R and v_R, which represent particular quantities in the system (e.g. positions, angles, velocities, forces, etc.). $\mathcal{W}_U$ represents those dynamics whose behaviour is uncertain, and that will therefore be included as real hardware in the simulator. q_R represents the quantity upon which the behaviour of the uncertain hardware depends. v_R represents the output of the uncertain dynamics to the rest of the system. $\mathcal{W}_K$ represents the dynamics of the system that close the loop around the uncertain dynamics. Finally, $\mathcal{W}_O$ represents the dynamics

The authors are with the Dept. of Engineering Science, Oxford University, Oxford, OX1 3PJ, UK. Email:monte.macdiarmid(ron.daniel)(marko.bacic)@eng.ox.ac.uk.

This work was supported by EPSRC grant EP/C512146/1.

978-1-4244-3124-3/08/$25.00 ©2008 IEEE

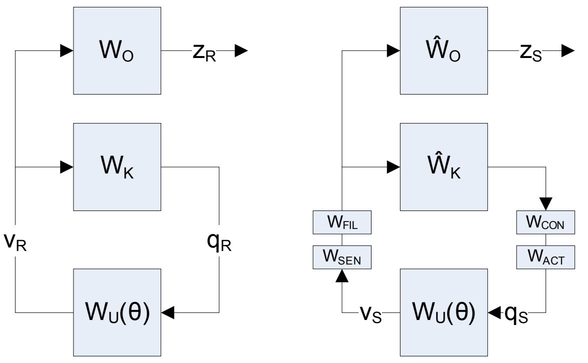

Fig. 1. Models of real system (left) and HWIL simulator (right)

that map the quantity v_R to the output quantity that the simulator must generate estimates of.

In the HWIL simulator, the interface between the uncertain hardware $\mathcal{W}_U$ and the rest of the system is broken, and the surrounding known system dynamics are replaced with a numerical system. In order to achieve this, interface actuators W_{ACT} and senors W_{SEN} are required. In a typical classical HWIL design, the numerical system will contain a filter/preprocessor W_{FIL} to generate an estimate $v_M \approx v_S$ based on measurements y, models $\hat{W}_K$ and $\hat{W}_O$ of the known system dynamics, and finally a controller $\mathcal{W}_{CON}$ to drive q_S to track the q_M produced by the model $\hat{W}_K$.

The classical approach to HWIL simulator design amounts to choosing the best available models $\hat{W}_K$, $\hat{W}_O$, and designing filters and controllers to neutralise the interface dynamics, i.e. produce $\mathcal{W}_{SEN} \circ \mathcal{W}_{FIL} \approx 0$ and $\mathcal{W}_{CON} \circ \mathcal{W}_{ACT} \approx 0$. While intuitively satisfying, this design approach is both inefficient and suboptimal. The classical approach is inefficient because it requires a complicated series of problem-specific design tasks, possibly including non-linear observer design, and/or controller design for an uncertain, non-linear system. This approach is suboptimal because the design of preprocessor, model and controller are all performed independently, rather than as a single unified whole. For an example of why this could potentially cause difficulties, consider the following scenario.

Imagine a HWIL simulator is to be designed for a particular system, for which the actuator selected has a significant lag. The classical design approach would respond to this by designing an aggressive controller to remove this lag and regain interface transparency. This could result in a problems with noise sensitivity, saturation, etc. Assume, in addition, that the model $\hat{W}_K$ of the known dynamics also posses a significant lag, which will in turn be included in the numerical system. There is a clear suboptimality in this design: the controller is designed to remove a large lag from the actuator, which is the reintroduced by the model, inside the same numerical system!

B. New Design Method

In order to address the problems of inefficiency and suboptimality in the classical approach to HWIL design, an novel design method has been developed. This new method is general, highly automated, and mathematically justified.

The method is based on the observation that the subdivision of the numerical system into preprocessor, model, and controller is artificial and unnecessary, as is the subdivision of the simulator hardware into actuators, included hardware and sensors. Instead, these system are unified into $\mathcal{W}_H$ (the simulator hardware), $\mathcal{W}_S$ (the closed loop numerical system) and $\mathcal{W}_P$ (the simulation postprocessing system, which produces the final simulation outputs). The design problem then amounts to the selection of $\mathcal{W}_S$ and $\mathcal{W}_P$ to produce the best possible simulator.

A very brief derivation of the actual design method is as follows. Again, refer to [10] for full details.

To begin, it is assumed that all systems are discretised. Next, it is assumed that the uncertainty in the shared hardware $\mathcal{W}_U$ (and hence also in $\mathcal{W}_H$) is parametric in nature, with probabilistic parameter vector $\theta \sim \mathcal{P}$, with $\mathcal{P}$ a probability distribution.

The concept of simulation optimality in general must then be defined. An optimal (SISO) simulation is defined as one that minimises the expected squared error between simulation output and the output of the real system, summed over the time horizon of interest, i.e.

$$(\mathcal{W}_S, \mathcal{W}_P)_{\text{OPT}} = \underset{(\mathcal{W}_S, \mathcal{W}_P)}{\arg\min}\, \mathbb{E}_\theta \left(\sum_{n=1}^{N} (z_S[n] - z_R(n))^2 \right) \tag{1}$$

However, while useful, this definition is incomplete in the HWIL context, as it ignores a fundamental goal of HWIL simulation. Unlike other forms of simulation, the goal of HWIL simulation is to not only produce an accurate estimate z_S, but to do so in such a way that the included hardware dynamics are driven through a trajectory that is as close as possible to what it would be in the real system. Therefore, $\mathcal{W}_S$ should in fact be designed first and foremost to minimise the error in q_S compared to q_R, under the assumption that $\mathcal{W}_P$ will then be able to produce an optimal output z_S. In other words, the output optimality definition alone will ensure a good simulation is produced; however, to ensure that the simulator is in fact a true *HWIL* simulator per se, rather than some other more general form of system identification / simulation, it is necessary to match q_S to q_R.

Follwing from this observation, an approximation is then made; for reasons of tractability, the final optimisation criteria used is to minimise error in inputs u to the simulation hardware, rather than q_S directly. This definition captures the underlying intuition of matching hardware trajectories, while allowing a useable design method to be developed.

Also for reasons of tractability, it is necessary to restrict the form of the numerical system $\mathcal{W}_S$. Specifically, $\mathcal{W}_S$ will be restricted to affine systems, i.e. systems in which the output at a particular time is a linear combination of previous inputs, plus some time-varying bias.

This results in a cost function to be minimised:

$$J = \mathbb{E}_\theta ||\mathbf{u} - \hat{\mathbf{u}}||_F^2 + \lambda^2 ||\mathbf{G}||_F^2 \tag{2}$$

where $\mathbf{u}$ represents the vector representation of the sequence of inputs for $n \in \{1, 2, \ldots, N\}$, $\hat{\mathbf{u}}$ represents the ideal sequence of inputs that would drive the shared hardware exactly as it would be driven in the real system. Note that both $\mathbf{u}$ and $\hat{\mathbf{u}}$ depend on the uncertain parameters θ, while $\mathbf{u}$ also depends on the numerical system. $\mathbf{G}$ represents the gain matrix of the affine numerical system $\mathcal{W}_S$. It can be shown that the action of any affine system over a finite horizon can be expressed as $\mathbf{u} = \mathbf{G}\mathbf{y} + \mathbf{b}$, where input and measurement sequences are seen as vectors $\mathbf{u}$ and $\mathbf{y}$ respectively. Furthermore, any causal system will produce a lower-triangular $\mathbf{G}$. The $||\mathbf{G}||$ penalises a numerical system that is sensitive to measurement noise, which is required as measurement noise is not modeled explicitly here.

Next, the uncertain parameters θ are restricted to a finite set of values (either naturally, or by approximation). This allows the expectation operator in the cost function to be replaced with a summation, which can then be consolidated into an overall matrix norm:

$$J = \left|\left|\left(\mathbf{U} - \hat{\mathbf{U}}\right)\mathbf{P}\right|\right|_F^2 + \lambda^2||\mathbf{G}||_F^2 \quad (3)$$

$$= \left|\left|\left(\bar{\mathbf{G}}\bar{\mathbf{Y}} - \hat{\mathbf{U}}\right)\mathbf{P}\right|\right|_F^2 + \lambda^2||\mathbf{G}||_F^2 \quad (4)$$

where $\mathbf{P}$ is the diagonalisation of the probability mass vector, the columns of any other matrix $\mathbf{A}$ represent the trajectories $\mathbf{a}$ resulting from a given realisation of θ, and $\bar{\mathbf{G}}$ and $\bar{\mathbf{Y}}$ are defined such that $\bar{\mathbf{G}}\bar{\mathbf{Y}} = \mathbf{G}\mathbf{Y} + \mathbf{1}\mathbf{b}$.

The minimisation of J as defined above can be proven to be convex for sufficiently large λ^2. Furthermore, the following iterated function system (IFS) can be proven to converge to the unique minimising numerical system $\bar{\mathbf{G}}_{\text{OPT}}$, again for sufficiently large λ^2:

$$\bar{\mathbf{G}}^{v+} = \phi\big(\bar{\mathbf{G}}^v\big) \quad (5)$$

where

$$\phi\big(\bar{\mathbf{G}}^v\big) = \left(\left(\frac{d\mathbf{f}^s\big(\bar{\mathbf{G}}^v\big)}{d\bar{\mathbf{G}}^v}\right)^T \mathbf{P}^{2\otimes}\,\mathbf{h}^*\big(\bar{\mathbf{G}}^v\big) + \lambda^2\mathbf{I}\right)^{-1} \times \left(\frac{d\mathbf{f}^s\big(\bar{\mathbf{G}}^v\big)}{d\bar{\mathbf{G}}^v}\right)^T \mathbf{P}^{2\otimes}\,\hat{\mathbf{U}}^s \quad (6)$$

In the above, $\bar{\mathbf{G}}^{v+}$ is the vectorisation of the non-zero elements of the matrix $\bar{\mathbf{G}}$, $(\cdot)^s$ is the matrix stack operator, $\mathbf{P}^{2\otimes} = (\mathbf{P} \otimes \mathbf{I})(\mathbf{P} \otimes \mathbf{I})$, and the functions $\mathbf{f}^s, \mathbf{h}^*$ are defined such that $\mathbf{U}^s = \mathbf{f}^s\big(\bar{\mathbf{G}}^v\big) = \mathbf{h}^*\big(\bar{\mathbf{G}}^v\big)\bar{\mathbf{G}}^v$.

While this solution method will produce the exact optimal design, the presence of the large Jacobian of $\mathbf{f}^s$ renders the method computationally difficult for practical trajectory lengths. In order to produce a more efficient method, the following approximation is introduced:

$$\frac{d\mathbf{f}^s\big(\bar{\mathbf{G}}^v\big)}{d\bar{\mathbf{G}}^v} \approx \mathbf{h}^*\big(\bar{\mathbf{G}}^v\big) \quad (7)$$

This results in the following IFS, which should be iterated to produce the approximate solution:

$$\bar{\mathbf{G}}^{v+} = \big(\mathbf{h}^{*T}\big(\bar{\mathbf{G}}^v\big)\mathbf{P}^{2\otimes}\,\mathbf{h}^*\big(\bar{\mathbf{G}}^v\big) + \lambda^2\mathbf{I}\big)^{-1} \times \mathbf{h}^{*T}\big(\bar{\mathbf{G}}^v\big)\mathbf{P}^{2\otimes}\,\mathbf{U}^s \quad (8)$$

This IFS can be implemented in practice very efficiently.

The theoretical validity of this approximation is currently under investigation, although in practice is has been successfully used in a variety of simulator designs.

III. Extensions

A. MIMO Systems

Although the design method given in section II is quite general in terms of acceptable system dynamics, a key limitation is the restriction to SISO systems. The majority of practical HWIL simulators are in fact MIMO, and therefore an extension of the methodology to deal with such systems was considered valuable.

To begin, the quantities u, q, v, y, z relevant to simulator design must be extended from scalars to vectors. As a result, where previously a finite-length trajectory in time was represented as a vector, now this quantity would naturally become a matrix, with the dimensionality of the quantity as one index and the sample index as the other. Finally, when many such trajectories, each associated with a different value of θ, are concatenated (as they must be for the algorithm to operate), the resulting quantity is in fact a rank-three tensor.

However, rather than deal directly with these tensor quantities, the problem will be kept in the domain of vectors and matrices via reshaping of the component arrays. Specifically, for a given quantity $\mathbf{A} \in (\mathbf{U}, \mathbf{Q}, \mathbf{V}, \mathbf{Y}, \mathbf{Z})$, a matrix form is defined:

$$\mathbf{A} = \begin{bmatrix} a_1^1[1] & \ldots & a_M^1[1] \\ \vdots & \ddots & \vdots \\ a_1^1[N] & \ldots & a_M^1[N] \\ & & \\ \vdots & \ddots & \vdots \\ & & \\ a_1^{S_a}[1] & \ldots & a_M^{S_a}[1] \\ \vdots & \ddots & \vdots \\ a_1^{S_a}[N] & \ldots & a_M^{S_a}[N] \end{bmatrix} \quad (9)$$

where an entry $a_m^{s_a}[n], (n, s_a, m) \in (N, S_a, M)$ represents the value of the s_ath component of a, at sample time n, under the value θ_m of the uncertain parameters. As a result, the gain matrix $\mathbf{G}$ and bias vector of the affine numerical system become

$$\mathbf{G} = \begin{bmatrix} g_{11}^{11} & \cdots & g_{N1}^{11} & \cdots & g_{1S_y}^{11} & \cdots & g_{NS_y}^{11} \\ \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ g_{11}^{N1} & \cdots & g_{N1}^{N1} & \cdots & g_{1S_y}^{N1} & \cdots & g_{NS_y}^{N1} \\ & & & & & & \\ \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ & & & & & & \\ g_{11}^{1S_u} & \cdots & g_{N1}^{1S_u} & \cdots & g_{1S_y}^{1S_u} & \cdots & g_{NS_y}^{1S_u} \\ \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ g_{11}^{NS_u} & \cdots & g_{N1}^{NS_u} & \cdots & g_{1S_y}^{NS_u} & \cdots & g_{NS_y}^{NS_u} \end{bmatrix} \tag{10}$$

$$\mathbf{b} = \begin{bmatrix} b_{11} & \ldots & b_{N1} & \ldots & b_{1S_y} & \ldots & b_{NS_y} \end{bmatrix} \tag{11}$$

$$\bar{\mathbf{G}} = \begin{bmatrix} \mathbf{b} \\ \mathbf{G} \end{bmatrix} \tag{12}$$

Under these definitions, the design method described in section II remains valid; one way to consider the effect of the above definitions is that a MIMO system has been converted into a SISO system by stacking the trajectories for each input or output end-to-end.

B. Design of Output System

The previous work has concentrated on the design of the closed loop numerical system $\mathcal{W}_S$, and has ignored the post-processing system $\mathcal{W}_P$. Although the closed loop effects make the design of $\mathcal{W}_S$ a more interesting and challenging problem in general, it is nonetheless important to design a suitable $\mathcal{W}_P$ in order to produce useful simulator outputs z_S from the measurements y.

In a classical design $\mathcal{W}_P$ appears naturally, as estimates v_M are already computed for $\mathcal{W}_S$, and values of z_S can be therefore be computed by passing v_M through the model $\hat{\mathcal{W}}_O$. However, in a simulator designed using the new method, this is no longer the case.

In accordance with (1), and subject to $\mathcal{W}_S$ being already designed, the optimal $\mathcal{W}_P$ is defined as

$$(\mathcal{W}_P)_{\text{OPT}} = \underset{\mathcal{W}_P}{\arg\min} \, ||(\mathbf{Z}_S - \mathbf{Z}_R)\mathbf{P}||_F^2 \tag{13}$$

Given that optimisation $\mathcal{W}_P$ over the full domain of arbitrary non-linear mappings is intractable, the form of $\mathcal{W}_P$ must be restricted (as was the case for $\mathcal{W}_S$). Again, the form chosen is that of affine mappings, i.e.

$$\mathbf{Z}_S = \bar{\mathbf{H}}\bar{\mathbf{Y}} \tag{14}$$

where $\bar{\mathbf{Y}}$ is defined as before, and $\bar{\mathbf{H}}$ is defined analagously to $\bar{\mathbf{G}}$, save over the dimensions of $\mathbf{Z}$ rather than $\mathbf{U}$.

The minimisation then becomes trivial; simply the solution of a linear least squares problem, $\min_{\bar{\mathbf{H}}} \left|\left|\bar{\mathbf{H}}\bar{\mathbf{Y}}\mathbf{P} - \mathbf{Z}_R\mathbf{P}\right|\right|_F^2$. Note that there are no restrictions on the structure of $\bar{\mathbf{H}}$ due to causality as there are on $\bar{\mathbf{G}}$; as $\mathcal{W}_P$ operates offline, the full output trajectories are accessible.

Finally, it should be noted that in practice a Tychonov regularizing term penalising the norm of $\bar{\mathbf{H}}$ with a weighting κ^2 is also used, as for the $\mathcal{W}_S$ design, to produce a design that is sufficiently robust to measurement noise when such noise is not explicitly included in the model.

C. Measurement Noise

The final extension described in this paper is to assist in dealing more explicitly with any unstructured uncertainty (e.g. measurement noise) present in the simulator, that exists outside of the θ-uncertainty.

The underlying concept is to attempt to include the unstructured uncertainty in the design in a similar manner as the structured uncertainty encoded in the θ parameter is included, i.e. by including the unstructured uncertainty in the expectation used to define the cost function. In order to do this a discretisation of this unstructured uncertainty is required. In general, it will not be computationally tractable to provide any sort of complete coverage of the domain of the realisations of the unstructured uncertainty, due to its massive dimensionality (e.g. for measurement noise, one dimension per channel per sample). Instead, a simple Monte Carlo approach will be used, where for each realisation of parameters θ, a number Υ of realisations of the unstructured uncertainty will sampled, and each simulated individually.

The end result of this process is that rather than having M columns, one for each realisation of θ, the various quantities ($\mathbf{U}$, $\mathbf{Y}$, etc.) will have $M\Upsilon$ columns, spanning both the shared, structured uncertainty and the unstructured uncertainty inherent in the simulator. The new, extended desired input $\hat{\mathbf{U}}$ that the simulator should match is simply formed of Υ copies of the original, as the unstructured uncertainty is associated with the simulator only.

In addition, the same approach can be extended to the design of $\mathcal{W}_P$; in fact, it is often possible to dispense entirely with the Tychonov parameter κ^2 from this part of the design, provided Υ is large enough to provide sufficient conditioning.

Another minor extension to method is to use different weighting λ^2, κ^2 for the gains associated with different measurement channels, to represent the fact that different noise amplitudes should be expected, and thus different sensitivities designed. Specifically, good results were obtained when the λ^2, κ^2 are proportional to the noise standard deviations on respective channels.

IV. Application to Aerodynamic HWIL Simulator

The simulation scenario described in this section is based on a real, experimental HWIL simulator that is in the final stages of commissioning. The simulator has been discussed previously in the literature [11]

A. System Modeling

The real system that is to be simulated is the motion of a simple aerodynamic object. Specifically, the system consists of a small airfoil section released into free fall, with a particular initial angle-of-attack relative to the vertical axis. The quantity of interest is the exact trajectory executed by the airfoil as it approaches stall. It is assumed that the airfoil is symmetrical about its section, so that all resulting motion

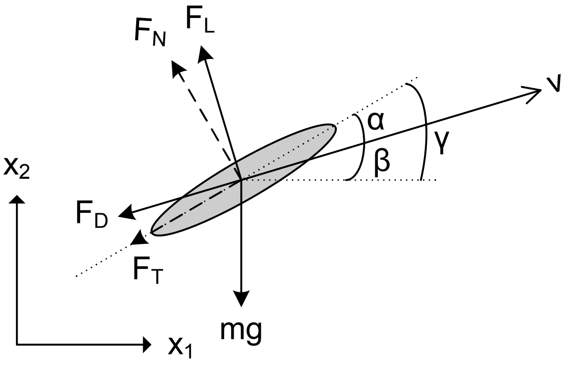

Fig. 2. Diagram of airfoil aerodynamics.

will be constrained to a single plane. A diagrams of the system can be seen in figure2.

A continuous-time state space model for this system is:

$$\frac{d}{dt}\begin{bmatrix} x_{1R} \\ x_{2R} \\ \gamma_R \\ \dot{x}_{1R} \\ \dot{x}_{2R} \\ \dot{\gamma}_R \end{bmatrix}(t) = \begin{bmatrix} \dot{x}_{1R} \\ \dot{x}_{2R} \\ \dot{\gamma}_R \\ -\frac{F_D \cos\beta_R + F_L \sin\beta_R}{m} \\ -\frac{F_D \sin\beta_R - F_L \cos\beta_R}{m} - g \\ \frac{M_T}{I_A} \end{bmatrix}(t) \qquad (15)$$

In the data set used in this example (see [12]), the lift force, drag force and pitching moment are defined to act at the quarter-chord point, and hence the total moment M_T around the centre of mass will not equal the quoted pitching moment M_P. In this example, it is assumed that the centre of mass is at the half-chord point for all airfoils; the resulting equation could be trivially modified were this not to be the case.

$$M_T = M_P + 0.25c(\sin\alpha\, F_D + \cos\alpha\, F_L) \qquad (16)$$

The aerodynamic forces F_D, F_L, M_P (drag force, lift force and pitching moment respectively) take the canonical form

$$F_D = 0.5\, C_D\, \rho\, \nu^2\, A \qquad (17)$$
$$F_L = 0.5\, C_L\, \rho\, \nu^2\, A \qquad (18)$$
$$M_P = 0.5\, C_M\, \rho\, \nu^2\, A\, c \qquad (19)$$

with C_D, C_L, C_M being the coefficients of drag, lift and pitching moment respectively, ν being the relative airspeed, A being the airfoil area, and c being the airfoil chord length. The aerodynamic coefficients are assumed to be described by non-linear functions of the angle of attack α, i.e.

$$C_D = c_D(\alpha), \quad C_L = c_L(\alpha), \quad C_M = c_M(\alpha) \qquad (20)$$

In the real system, the angles γ, β and α (being the absolute airfoil orientation, the absolute angle of the velocity vector, and the angle of attack respectively) are defined as follows

$$\alpha_R = \gamma_R - \beta_R \qquad (21)$$
$$\beta_R = \operatorname{atan2}(\dot{x}_{2R}, \dot{x}_{1R}) \qquad (22)$$

Also, the airspeed is defined as the magnitude of the airfoil velocity, $\nu_R = \sqrt{\dot{x}_{1R}^2 + \dot{x}_{2R}^2}$, i.e. the surrounding air is assumed to be still.

The output quantities of interest in this example are those defining the positional trajectory of the airfoil, i.e. x_{1R}, x_{2R} and orientation γ_R.

In order to simulate the behaviour of this system (which could be expensive, awkward and time-consuming to test physically), a HWIL simulator is to be designed. The physical design of this simulator has been performed *a priori*, and is based around a controlled wind tunnel combined with a three-axis motion simulator (of which only one axis will be used in this scenario). The airfoil is attached to the tool point of the motion simulator, through a force-torque sensor that is able to measure the forces F_T, F_N and torque M_S between the airfoil and the mounting. The sensor is calibrated to read zero at rest. The angle α_S of the motion simulator is also measured, as is the airspeed ν_S. All of these measurements are subject to independent gaussian measurement noises $\epsilon_i \sim N(0, \sigma_i^2)$. The measurements must be processed by the numerical subsystem (the design of which is the subject of this work), in order to produce inputs to the simulator hardware; the first input sets the torque M_U applied to the motion simulator, and the second input controls the voltage v applied to the fan driving the wind tunnel airspeed. Finally, these measurements must also be used to generate an estimate of the actual quantities of interest, namely the simulated positional trajectory $(x_{1S}, x_{2S}, \gamma_S)$.

The motion simulator has been designed for high levels of performance as well as ease of modeling. The dynamics of the axis itself can be modeled as a simple rotational inertia I_S. The effect of friction is negligible. The tool point is at the rotational centre of the device, and mounts to the centre of gravity of the airfoil. The axis is controlled by a direct-drive torque-controlled DC motor. The amplifier driving this motor is four quadrant and fully linear, and is current-limited to produce a maximum absolute torque of M_{max}.

The wind tunnel fan can be modeled as a first order system taking input voltage to airspeed, with a time constant τ_ν. The DC gain of the fan system depends on the amplifier settings, and has little effect beyond scaling the required voltage inputs as the maximum fan speed is significantly beyond the speeds reached in these experiments. Therefore, for simplicity, unity DC gain is assumed.

The continuous-time dynamics of the simulator can be expressed as

$$\frac{d}{dt}\begin{bmatrix} \alpha_S \\ \dot{\alpha}_S \\ \nu_S \end{bmatrix}(t) = \begin{bmatrix} \dot{\alpha}_S \\ \frac{M_T + M_U}{I_{SA}} \\ \tau_\nu(v - \nu_S) \end{bmatrix}(t) \qquad (23)$$

where $I_{SA} = I_S + I_A$. Measurements are formed via

$$F_T = F_D \cos\alpha_S - (F_L - mg)\sin\alpha_S \qquad (24)$$
$$F_N = F_D \sin\alpha_S + (F_L - mg)\cos\alpha_S - mg \qquad (25)$$
$$M_S = M_T - \frac{\ddot{\alpha}_S}{I_A} \qquad (26)$$

Note that the $(F_L - mg)$ term exists because in the simulator airflow is always horizontal, and thus lift always vertical and directly opposing gravity. The $-mg$ term at the end of the F_N equation exists because the force-torque sensor is always calibrated to read zero at rest, i.e. the F_N axis is shifted by the weight of the airfoil.

The purpose of the simulator is to allow the testing of a variety of different aerodynamic objects. Thus, the aerodynamic behaviour of the actual airfoil under test will be inherently uncertain. Specifically, for the purposes of this paper, a simulator is required that can test all of the airfoils described in [12]. These airfoils represent a sample of 50+ airfoils that are popular in small scale glider designs. Experimental data describing the C_D, C_L and C_M curves of these airfoils is available in [12]; this data is incorporated into the system model, with the uncertain parameter $\theta \in \{1, 2, 3, \ldots, M\}$ indicating which particular airfoil is currently under test. In the absence of any more specific prior information, all individual airfoils are assumed to be equally likely; therefore, θ is assumed to be uniformly distributed, i.e. $\Pr(\theta = i) = 1/M$, $\forall i \in \{1, 2, \ldots, M\}$.

Before a numerical model can be designed for this simulator, the above models must be discretised. In this example, discretisation is carried out using a simple zeroth-order method, i.e.

$$x[n+1] \approx x[n] + T\dot{x}[n] \tag{27}$$

where T represents the sample period, and x represents any of the states described above.

The system described above can be cast in the notation used in preceeding sections:

$$q_R = \begin{bmatrix} \alpha_R \\ \nu_R \end{bmatrix}, \quad q_S = \begin{bmatrix} \alpha_S \\ \nu_S \end{bmatrix} \tag{28}$$

$$v_R = \left.\begin{bmatrix} F_D \\ F_L \\ M_P \end{bmatrix}\right|_{\alpha=\alpha_R} \quad v_S = \left.\begin{bmatrix} F_D \\ F_L \\ M_P \end{bmatrix}\right|_{\alpha=\alpha_S} \tag{29}$$

$$z_R = \begin{bmatrix} x_{1R} \\ x_{2R} \\ \gamma_R \end{bmatrix}, \quad z_S = \begin{bmatrix} x_{1S} \\ x_{2S} \\ \gamma_S \end{bmatrix} \tag{30}$$

$$u = \begin{bmatrix} M_U \\ v \end{bmatrix}, \quad y = \begin{bmatrix} F_T + \epsilon_1 \\ F_N + \epsilon_2 \\ M_S + \epsilon_3 \\ \alpha_S + \epsilon_4 \\ \nu_S + \epsilon_5 \end{bmatrix} \tag{31}$$

Finally, the parameter values in the example system are:

$$\begin{aligned} \alpha_R[0] &= 5^\circ & I_A &= 0.00147kg\,m^2 \\ x_{1R}[0] &= 0m & g &= 9.81m\ s^{-2} \\ x_{2R}[0] &= 0m & \rho &= 1.204 \\ \gamma_R[0] &= -90^\circ + \alpha_R[0] & A &= 0.08m^2 \\ \dot{x}_{1R}[0] &= 0ms^{-1} & c &= 0.2m \\ \dot{x}_{1R}[0] &= 0ms^{-1} & \alpha_S[0] &= 5^\circ \\ \dot{\gamma}_R[0] &= 0s^{-1} & \dot{\alpha}_S[0] &= 0s^{-1} \\ T &= 0.002s & \nu_S[0] &= 0ms^{-1} \\ m &= 0.3kg & I_S &= 0.1kg\,m^2 \\ & & \tau_\nu &= 1s \\ \sigma_{1:5} &= \left[10^{-2}, 10^{-2}, 10^{-3}, 10^{-6}, 10^{-2}\right] \end{aligned} \tag{32}$$

Tests are conducted over a time horizon of $400ms$ (i.e. $N = 200$) and $M = 50$ different airfoils are considered.

B. Classical Simulator Design

For the purposes of comparison with the newly developed design method, this section describes a classical design for the numerical subsystem of the simulator discussed in section IV-A. The classical design consists of three core components. First of all, the measurements y must be processed in order to obtain an estimate of the outputs v_S of the shared hardware. These quantities v_S (i.e. aerodynamic forces) must then be fed into a model of the dynamics not included as hardware (i.e. the inertial behaviour of the airfoil), thereby producing outputs z_S, as well as quantities q_M on which future aerodynamic forces will depend. Finally, a controller is required to drive the simulator hardware in such a way that the actual quantities q_S track the reference q_M produced by the model. This controller must make used of the measurements y, and produce control inputs u.

Extracting an estimate of v_S based on measurements y requires the transformation of the forces F_T, F_N, M_S into F_D, F_L, M_T. In general this may involve filtering the noisy measurements beforehand. In this case, however, the negative performance impact of the filter lag was considered excessive, and the modeled airfoil inertia sufficient to damp out measurement noise.

The forces in the two frames are related by:

$$F_D = (F_N + mg)\sin\alpha_S + F_T\cos\alpha_S \tag{33}$$

$$F_L = (F_N + mg)\cos\alpha_S - F_T\sin\alpha_S + mg \tag{34}$$

$$M_T = M_S\left(1 + \frac{I_A}{I_S}\right) + M_U\frac{I_A}{I_S} \tag{35}$$

Therefore, estimates $\hat{F}_D, \hat{F}_L, \hat{M}_T$ for use by the model can be obtained via

$$\hat{F}_D = \left(y^2[n] + mg\right)\sin y^4[n] + y^1[n]\cos y^4[n] \tag{36}$$

$$\hat{F}_L = \left(y^2[n] + mg\right)\cos y^4[n] - y^1[n]\sin y^4[n] + mg \tag{37}$$

$$\hat{M}_T = y^3[n]\left(1 + \frac{I_A}{I_S}\right) + u^1[n-1]\frac{I_A}{I_S} \tag{38}$$

These forces are then applied to a discretised model of the inertial behaviour of the system:

$$\Delta \begin{bmatrix} x_{1M} \\ x_{2M} \\ \gamma_M \\ \dot{x}_{1M} \\ \dot{x}_{2M} \\ \dot{\gamma}_M \end{bmatrix} [n] = T \begin{bmatrix} \dot{x}_{1M} \\ \dot{x}_{2M} \\ \dot{\gamma}_M \\ -\frac{\hat{F}_D \cos\beta_M + \hat{F}_L \sin\beta_M}{m} \\ -\frac{\hat{F}_D \sin\beta_M - \hat{F}_L \cos\beta_M}{m} - g \\ \frac{M_T}{I_A} \end{bmatrix} [n] \quad (39)$$

Finally, controllers attempt to drive the quantities q_S (simulator hardware angle of attack and airspeed) to track the quantities q_M generated by the above model. The design of these controllers presents a key difficulty in the classical approach; the dynamics driving the q_S quantities will be determined by the dynamics of the entire system, including the non-linear, uncertain aerodynamics. Therefore, a rigorous design approach requires the linearisation of an uncertain non-linear system, followed by a controller design around these systems.

Consider the α_S axis first. An approximate transfer function from input torque $u^1 = M_U$ to simulator angle of attack $q^1 = \alpha_S$ can be obtained in a straightforward manner by linearising around $\alpha_S \approx 0$. This approximation results in

$$M_T \approx M_P + 0.25 c F_L \quad (40)$$

$$F_L \approx 0.1 q^1 \quad (41)$$

where $F_L \approx 0.1 q^1$ uses a commonly used estimate of the gradient of $C_L(\alpha)$, $\alpha \approx 0$. This results in a transfer function:

$$\frac{q^1}{u^1}(s) = \frac{1}{I_{SA} s^2 - 0.025c} \quad (42)$$

Note that, conveniently, these approximations result in a transfer function that is independent of the particular airfoil used, i.e. is independent of the uncertainty θ. Thus, robust control techniques will not be needed in this case, although in general most classical HWIL designs will require them.

A controller can be designed for this transfer function. Specifically, in this example a proportional-derivative form has been chosen, with parameters resulting in a phase margin $PM = 77°$ at a frequency of $10 rad/s$.

A controller for the wind tunnel airspeed is simpler to design, as the airspeed trajectory is dominated by the gravitational acceleration, and is largely unaffected by the aerodynamic behaviour. Thus, control can be provided in open loop provided the time constant τ_ν of the fan is known accurately, or with a dominant feedforward term combined with a small feedback term to correct any small control mismatches. In this design, a purely open loop controller will be used. Gravitational acceleration will produce:

$$q_M^2(s) \approx \frac{g}{s^2} \quad (43)$$

which can be combined with the fan dynamics to obtain the desired control trajectory

$$\frac{q_S^2}{u^2}(s) = \frac{\tau_\nu}{s + \tau_\nu} \quad (44)$$

$$\Rightarrow u^2(s) = \frac{g}{s^2} \frac{\tau_\nu}{s + \tau_\nu} \quad (45)$$

$$\Rightarrow u^2(t) = g(1 + t) \quad (46)$$

$$\Rightarrow u^2[n] = g(1 + Tn) \quad (47)$$

C. New Design

Again, the application of the new design method is straightforward, requiring little more than the selection of values for a handful of parameters. The values used here are:

$K = 15$	(iterations of the IFS)	(48)
$L = 30$	(non-zero gains allowed at each time)	(49)
$\Upsilon = 50$	(noise realisations)	(50)
$\lambda_b^2 = 6$	(base regulariser, is multiplied by σ_{s_y})	(51)
$\kappa_b^2 = 0$	(base regulariser, is multiplied by σ_{s_y})	(52)

Note that the parameters λ_b^2 and κ_b^2 represent base weights, which are multiplied by the standard deviations of the noise on each measurement channel.

The results of the two designs are compared in the following section.

V. Results

Figure 3 shows the trajectories executed by the airfoils in real free flight. These are the trajectories that the simulators are designed to predict, depending on which particular airfoil is under test (i.e. on the value of the uncertain parameter θ).

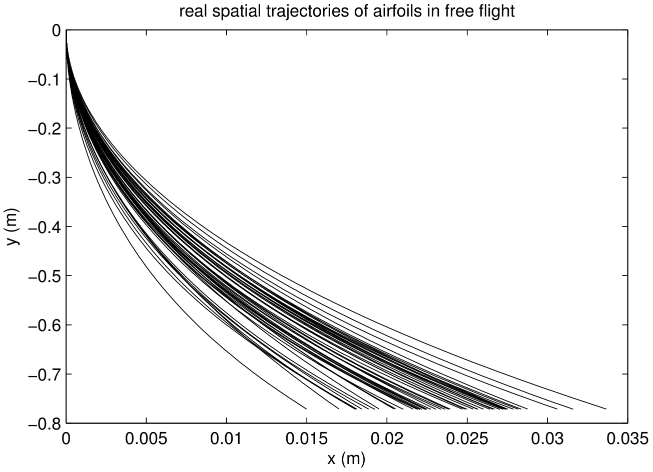

Fig. 3. Spatial trajectories of airfoils in real free flight.

Figure 4 shows the simulation errors (positions and orientation) for both the new and classical design methods. The plots show 90% confidence intervals, calculated by taking a large number of trials with a variety of noise realisations and basing statistics on this ensemble. These plots clearly

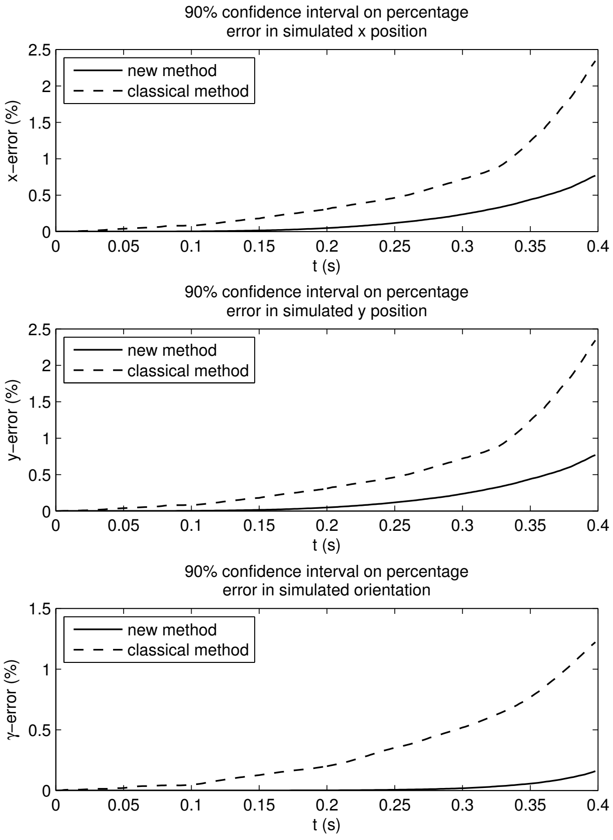

Fig. 4. 90% confidence intervals on percentage output errors in simulators designed using the classical and new methods.

demonstrate that the new design method outperforms the old method with a high degree of confidence.

In addition to increased simulation accuracy, the new method has a number of other advantages over the classical design method. The requirement of arduous problem-specific design work is vastly reduced, due to the automated nature of the method; also, the new method works in situations in which the classical design would be inherently difficult, for example when the transformation between measurements y to estimates of v is ill defined, as would be the case in this example if no torque was measured, but rather had to be observed from the dynamics of α. Finally, the numerical system designed by the new method results in a simple affine system that is extremely efficient to implement, whereas the classical design requires non-linear models and other complicated features that may not be practical inside an embedded computational environment.

VI. Conclusions and Future Work

In this paper, extensions to an existing proven HWIL design method have been described. Specifically, the method was extended to deal with MIMO systems, to fully design the output stage of the numerical system in addition to the online closed loop stage, and to explicitly deal with unstructured uncertainty.

Also, and example application was presented, in which the extended design method was compared to a competing classical design, in the context of a real, large scale HWIL simulator. The new method was found to outperform the classical design and provide other practical advantages.

In the near future, it is anticipated that the physical simulator will come online, at which time the design method will be tested on the real plant.

References

[1] J. M. Carter and K. E. Willis, “History of flight motion simulators used for hardware-in-the-loop testing of missile systems,” R. L. Murrer and Jr., Eds., vol. 3368, no. 1. SPIE, 1998, pp. 425–431. [Online]. Available: http://link.aip.org/link/?PSI/3368/425/1

[2] G. Babbitt, R. Bonomo, and J. Moskwa, “Design of an integrated control and data acquisition system for a high-bandwidth, hydrostatic, transient engine dynamometer,” in *Proceedings of the 1997 American Control Conference*, 1997, pp. 1157–1161.

[3] G. Babbitt and J. Moskwa, “Implementation details and test results for a transient engine dynamometer and hardware-in-the-loop vehicle model,” in *Proceedings of the 1999 IEEE International Symposium on Computer Aided Control System Design*, 1999, pp. 569–574.

[4] S. Brennan, A. Alleyne, and M. DePoorter, “The illinois roadway simulator - a hardware-in-the-loop testbed for vehicle dynamics and control,” *ASME Journal of Dynamics, Systems, Measurement and Control*, vol. 124.

[5] S. Brennan and A. Alleyne, “The Illinois Roadway Simulator: a mechatronic testbed for vehicledynamics and control,” *Mechatronics, IEEE/ASME Transactions on*, vol. 5, no. 4, pp. 349–359, 2000.

[6] S. B. Choi, Y. T. Choi, and D. W. Park, “A sliding mode control of a full-car electrorheological suspension system via hardware in-the-loop simulation,” *Journal of Dynamic Systems, Measurement, and Control*, vol. 122, no. 1, pp. 114–121, 2000.

[7] A. Darby, A. Blakeborough, and M. Williams, “Real-time substructure tests using hydraulic actuator,” *Journal of Engineering Mechanics*, vol. 125, no. 10, pp. 1133 – 1139, 1999.

[8] P. Gawthrop, M. Wallace, and D. Wagg, “Bond-graph based substructuring of dynamical systems,” *Earthquake Engineering and Structural Dynamics*, vol. 34, no. 6, pp. 687 – 703, 2005.

[9] F. Aghili, E. Dupuis, J.-C. Piedboeuf, and J. de Carufel, “Hardware-in-the-loop simulation of robots performing contact tasks,” *European Space Agency, (Special Publication) ESA SP*, no. 440, pp. 583 – 588, 1999.

[10] M. MacDiarmid, R. Daniel, and M. Bacic, “Design of numerical subsystems for hardware-in-the-loop simulators,” Tech. Rep., 2008, available at: http://users.ox.ac.uk/˜engs0582/pdf/msmhwil2008.pdf.

[11] M. Bacic and R. Daniel, “Towards a low-cost hardware-in-the-loop simulator for free flight simulation of unmanned air vehicles,” *2005 AIAA Modeling and Simulation Technologies Conference and Exhibit*, pp. 1–9, 2005.

[12] C. Lyon, A. Broeren, P. Giguere, A. Gopalarathnam, and M. Selig, “Summary of Low-Speed Airfoil Data, Vol. 3,” *Virginia Beach: SoarTech Publ*, vol. 417, 1997.