Structure and Parameter Identification of Process Models with Hard Non-linearities for Industrial Drive Trains by Means of Degenerate Genetic Programming

Mathias Tantau[1], Lars Perner[2], Mark Wielitzka[1] and Tobias Ortmaier[1]
[1]*Institute of Mechatronic Systems, Leibniz University Hanover, Appelstr. 11a, 30165 Hannover, Germany*
[2]*Lenze Automation GmbH, Am Alten Bahnhof 11, D-38122 Braunschweig, Germany*

Keywords: Genetic Programming, Modelling, Simultaneous Identification of Structure and Parameters, Phenomenological Models, Backlash, Multiple-mass Resonators.

Abstract: The derivation of bright-grey box models for electric drives with coupled mechanics, such as stacker cranes, robots and linear gantries is an important step in control design but often too time-consuming for the ordinary commissioning process. It requires structure and parameter identification in repeated trial and error loops. In this paper an automated genetic programming solution is proposed that can cope with various features, including highly non-linear mechanics (friction, backlash). The generated state space representation can readily be used for stability analysis, state control, Kalman filtering, etc. This, however, requires several special rules in the genetic programming procedure and an automated integration of features into the defining state space form. Simulations are carried out with industrial data to investigate the performance and robustness.

1 INTRODUCTION

For control design, Kalman filtering, model-based fault diagnosis (Witczak et al., 2002) in the field of electric drive trains detailed process models are essential. For these applications it is important that the models do not only approximate the systems accurately but that they are also physically correct with interpretable parameters (bright-grey box modelling). Such phenomenological models (in contrast to black box models) help to better understand the system of interest and they allow for advanced techniques from control theory, such as state control, online parameter tuning and flatness based control, as they are used in stacker cranes, robots and linear gantries.

Unfortunately their derivation is very time-consuming because iteratively different possible models must be evaluated and then rejected if their complexity is inappropriate or they build on irrelevant system properties. Genetic programming (GP) offers a solution to the automated identification of model structures, but classical GP is limited to simple functions (Koza, 1994). It cannot create the kind of process models as they are known from electric drive trains with flexible mechanics, friction or backlash. Often they approximate only static functions (Toropov and Alvarez, 1998).

Extensions to dynamical systems can be found, often in combination with black-box models that have a parameter-linear structure (dos Santos Coelho and Pessôa, 2009). This property facilitates the selection of important predictors, which can be seen as basic structure optimization.

If the dynamic models are not linear in parameters, parameters can be included in the form of terminal nodes that are altered by mutations (Winkler et al., 2004).

In the work of Marenbach et.al. (Marenbach et al., 1995) the parameter sets of the dynamic transfer function models are optimized in each step of the GP, which is time-consuming. In (Gray et al., 1997) a similar concept with non-linear models like saturation is presented. The problem remains that the resulting process models can hardly be interpreted physically and they are not given in a structured form as required for example for control design.

The aim of this paper is to derive transparent, physically motivated process models for electric drive trains by combining a-priori knowledge with GP structure identification. The function set is tailored to the specific properties of electric drives with imperfect mechanics. The resulting models are organized in

Tantau, M., Perner, L., Wielitzka, M. and Ortmaier, T.
Structure and Parameter Identification of Process Models with Hard Non-linearities for Industrial Drive Trains by Means of Degenerate Genetic Programming.
DOI: 10.5220/0007949003680376
In *Proceedings of the 16th International Conference on Informatics in Control, Automation and Robotics (ICINCO 2019)*, pages 368-376
ISBN: 978-989-758-380-3

the form of state space representations that can be the starting point for the above mentioned techniques or for further investigations like controllability and stability analysis. The intend is that the resulting models look similar to those designed by an experienced engineer although being created automatically. The structures are not, however, claimed to be the best possible way of modelling and equivalent models may exist as the state space description is per se not unique.

2 PHENOMENOLOGICAL MODELLING OF ELECTRIC DRIVES

Structure optimization by GP in this paper means optimizing the discrete quantity of different *subsections* of the overall process model. These subsections are also called *submodels*, associated with known *physical effects* as described in the following. In genetic programming terms they define the function set that the algorithm can choose from when building the individuals of the population. When the quantity of a certain submodel is changed from 0 to 1, it means that the associated physical effect is now included in the model. Quantities greater than one are also conceivable, e.g. for simultaneous friction at different locations.

2.1 Physical Effects

Physical effects, also called *properties* or *features* define the behaviour of the overall model by their compilation. The following list enumerates all the properties incorporated in the function set of this paper. Future extensions are possible, but it is believed that the given set comprises a reasonably comprehensive selection of typically considered physical effects, while still maintaining structural output distinguishability (s.o.d.) (Walter et al., 1984). Although not proven, the function set is chosen such that s.o.d. should be respected because for each submodel it is possible to describe its characteristic, distinct contribution to the input-output behaviour. Further restrictions of diversity will be introduced in section 3.1.

Multiple-mass Systems with Different Numbers of Masses: Figure 1 is a translational sketch of the considered class of rotatory multiple mass systems, showing the angular coordinates q_i, the moments of inertia J_i, the spring constants c_i and damper constants d_i. $N \in \{1,2,3,4\}$ is the number of masses. The sensor signals $q, \dot{q}$ and $\ddot{q}$ are strictly bound to the first inertia in this paper, as is the actuator with its torque M_M. Other configurations are possible but would easily lead to ambiguous, indistinguishable transfer functions and it is believed that this collocated structure, sometimes called ladder structure can be found in most electric drive trains anyway, because the collocation facilitates the control design (Berglund and Hovland, 2000). The set of estimation parameters is $\{J_i, c_i, d_i\}$.

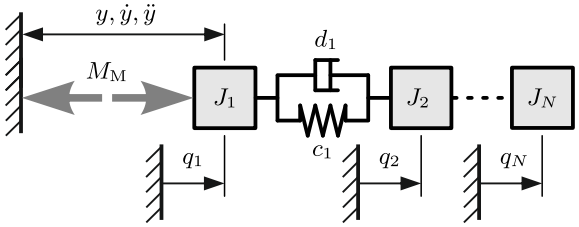

Figure 1: Class of rotatory multiple-mass systems with motor and position/velocity/acceleration sensor at the first mass, drawn translational for simplicity.

Multiple mass systems with arbitrary actuator and sensor positions can be modelled with mass $\boldsymbol{M}$, damping $\boldsymbol{D}$ and stiffness matrix $\boldsymbol{C}$:

$$\boldsymbol{M}\ddot{\boldsymbol{q}} + \boldsymbol{D}\dot{\boldsymbol{q}} + \boldsymbol{C}\boldsymbol{q} = \boldsymbol{F}. \tag{1}$$

Static Friction Model with Three Independent Components: In agreement with the summands of the equation

$$M_\mathrm{F} = \underbrace{-f_\mathrm{v}\dot{q}_i}_{\text{viscous}} \underbrace{-\tanh\left(f_\mathrm{tanh}\dot{q}_i\right)\left[M_\mathrm{C} + M_\mathrm{S}\mathrm{e}^{-\dot{q}_i/\dot{q}_{i,0}}\right]}_{\text{Coulomb and Stribeck}} \tag{2}$$

the friction torque can be divided into three parts: viscous, Coulomb and Stribeck friction (Schütte, 2003). The gain f_tanh is defined upfront in this study to allow a stable simulation, leaving only four estimation parameters: $\{f_\mathrm{v}, M_\mathrm{C}, M_\mathrm{S}, \dot{q}_{i,0}\}$. Index i determines the inertia J_i with friction.

Gravity: For gravity a constant torque M_G is added to the one of the masses so that only the one estimation parameter $\{M_\mathrm{G}\}$ adds to the overall model.

Backlash: Backlash is included between the first and second inertia in figure 2. Following the *physical backlash model* (Nordin et al., 1997; Zemke, 2012) the backlash element with a width of 2α is connected in series with the spring damper element. Coordinate λ is the position in the backlash gap. When the element is fully extended, it holds that $\lambda = 0$, so the valid range for λ is $-2\alpha \leq \lambda \leq 0$. Usually a symmetric range is chosen (Zemke, 2012), but here the deliberately asymmetric range facilitates simulation,

because now the state $\lambda = 0$ corresponds to a practically reproducible initial position.

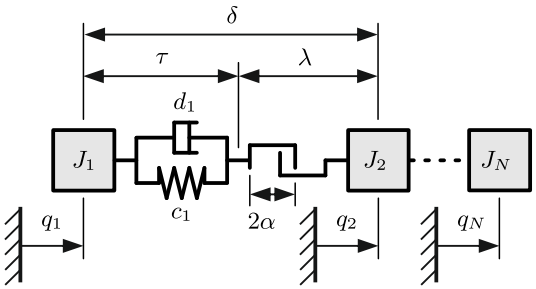

Figure 2: Backlash between the first and second mass. The system is considered rotary.

In general, when backlash is added to the spring damper element with index k between inertias i and j, [1] the following equation describes the torque that presses the spring damper element against the adjacent inertias:

$$M_{B,k} = c_k \tau_k + d_k \dot{\tau}_k \tag{3}$$

$$\text{with} \quad \tau_k = \delta_k - \lambda_k, \delta_k = q_i - q_j. \tag{4}$$

Coordinate λ_k follows the differential equation (k being omitted)

$$\dot{\lambda} = f_\lambda(.) = \begin{cases} \max\left(0, \dot{\delta} + \frac{c}{d}(\delta - \lambda)\right) & \text{for } \lambda = -2\alpha \\ \dot{\delta} + \frac{c}{d}(\delta - \lambda) & \text{for } -2\alpha < \lambda < 0 \\ \min\left(0, \dot{\delta} + \frac{c}{d}(\delta - \lambda)\right) & \text{for } \lambda = 0. \end{cases} \tag{5}$$

It represents an integrator with saturation (Nordin et al., 1997; Zemke, 2012). The only estimation parameter is $\{\alpha\}$

Figure 3 shows the hysteresis loop of the element. The re-engagement points at the margins of the backlash gap may deviate slightly from -2α or 0 if the spring damper element in preloaded.

Current Control: The current control loop is approximated by a P_{T1}-element. This is commonly done because the electric constants are much smaller than the mechanical time constants (Schütte, 2003). The new estimation parameter is the reset time $\{T_1\}$.

Delay Time: Only one external delay is considered. The transfer function has one parameter $\{T_d\}$ in the case of single-input single-output (SISO) systems:

$$G_{dead} = e^{-T_d s}. \tag{6}$$

[1] Index j is not necessarily $i+1$ if spring damper elements are allowed to be spanned between arbitrary combinations of masses, for example mass 1 and 3.

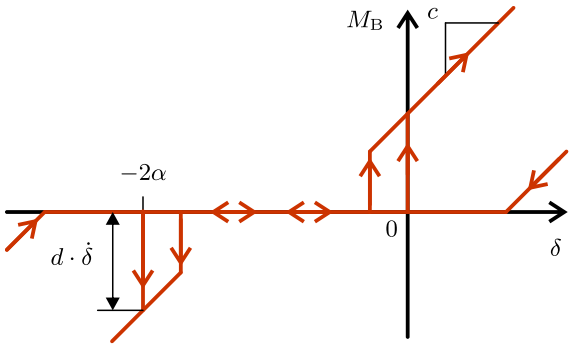

Figure 3: Hysteresis loop of the spring damper element with backlash.

2.2 Interchangeable Construction Kit

In order to automate the process of modelling and structure identification it is necessary to define submodels for the physical effects that can be assembled by an algorithm that has no physical understanding. The interfaces must be defined in a way that all permissible combinations of submodels lead to valid, executable simulation models when combined and no additional adjustments are required. This is a challenge because the various submodels are given in different forms (equations of motion, transfer functions, piecewisely defined functions). Also, the concept must allow for interchangeable extensions to the function set defined so far.

As a solution a kind of "construction kit" modelling in state space form is proposed. Figure 4 shows the general structure. Matrices $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}$ and $\boldsymbol{D}$ represent the linear part, while the system function $\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u})$ and output function $\boldsymbol{g}(\boldsymbol{x}, \boldsymbol{u})$ form the non-linear part of the model. Time delay is provided for at the input and output.

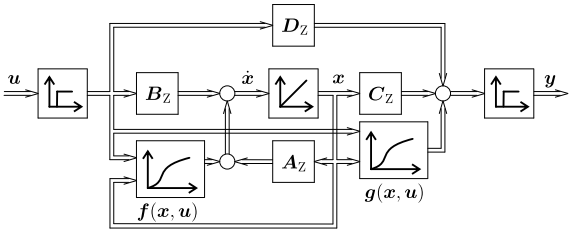

Figure 4: State space representation of the construction kit modelling with linear and non-linear part.

Initially the state space form is empty having no states. Then the submodels are added as described below, in the given order. The operations do not change the general shape, but only the content.

Multiple-mass Systems with Different Numbers of Masses: From $\boldsymbol{M}, \boldsymbol{D}$ and $\boldsymbol{C}$ a linear state space form

can be derived:

$$\boldsymbol{A}_Z = \begin{pmatrix} \mathbf{0}_{N\times N} & \mathbf{I}_{N\times N} \\ -\boldsymbol{M}^{-1}\boldsymbol{C} & -\boldsymbol{M}^{-1}\boldsymbol{D} \end{pmatrix}, \tag{7}$$

$$\boldsymbol{B}_Z = \begin{pmatrix} \mathbf{0}_{N\times N} \\ \boldsymbol{M}^{-1} \end{pmatrix}, \tag{8}$$

$$\boldsymbol{C}_Z = \begin{pmatrix} \mathbf{I}_{2N\times 2N} \\ (\ \mathbf{0}_{N\times N} \quad \mathbf{I}_{N\times N}\ )\boldsymbol{A}_Z \end{pmatrix}, \tag{9}$$

$$\boldsymbol{D}_Z = \begin{pmatrix} \mathbf{0}_{2N\times N} \\ (\ \mathbf{0}_{N\times N} \quad \mathbf{I}_{N\times N}\ )\boldsymbol{B}_Z \end{pmatrix}, \tag{10}$$

The symbols $\mathbf{I}$ and $\mathbf{0}$ represent unity matrix or zero matrix, respectively. The input of the resulting MIMO system is a vector of torques, one for each inertia. The output is a vector composed of $3N$ elements for position, velocity and acceleration $(\boldsymbol{q},\dot{\boldsymbol{q}},\ddot{\boldsymbol{q}})^{\mathrm{T}}$. Later, the actual actuator and sensor positions will be defined.

Static Friction Model with Three Independent Components: The friction torque M_{F} introduced above acts as a non-linear state feedback, which doesn't exist in the state space form from figure 4. It must be calculated into a system function and added to $\boldsymbol{f}(\boldsymbol{x},\boldsymbol{u})$:

$$\boldsymbol{f}(\boldsymbol{x},\boldsymbol{u}) := \boldsymbol{f}(\boldsymbol{x},\boldsymbol{u}) + \boldsymbol{B}_Z \begin{pmatrix} \mathbf{0}_{i-1\times 1} \\ M_{\mathrm{F}}(\dot{q}_i) \\ \mathbf{0}_{N-i\times 1} \end{pmatrix}. \tag{11}$$

Gravity: Gravity can be interpreted as a non-linear feedback at the input i, similar to friction. The only difference is that the gravity torque is independent of the system states.

Backlash: The physical backlash model is more complicated to integrate because it affects the number of states. Again the case is considered that backlash is added to the spring damper element k that connects inertia i with inertia j. The procedure follows five steps:

1. State λ_k is added to the state space form without any connections to inputs, states and outputs.
2. In order to simplify the calculations only a subset of the state space form is further addressed, defined by the reduced inputs, states and outputs, and correspondingly $\boldsymbol{A}_{\mathrm{red}}$, $\boldsymbol{B}_{\mathrm{red}}$, $\boldsymbol{C}_{\mathrm{red}}$, $\boldsymbol{D}_{\mathrm{red}}$, $\boldsymbol{f}_{\mathrm{red}}$, $\boldsymbol{g}_{\mathrm{red}}$:

$$\boldsymbol{u}_{\mathrm{red}} = \{\}, \tag{12}$$

$$\boldsymbol{x}_{\mathrm{red}} = \{q_i, q_j, \lambda_k, \dot{q}_i, \dot{q}_j\}, \tag{13}$$

$$\boldsymbol{y}_{\mathrm{red}} = \{\ddot{q}_i, \ddot{q}_j\}. \tag{14}$$

   By this means the next steps are independent of all properties that may have been added before.
3. This step incorporates the linear spring feedback at state λ_k. To do so, a matrix $\boldsymbol{A}_{\mathrm{add}}$ of appropriate size is added to the system matrix $\boldsymbol{A}_{\mathrm{red}}$. It has only zeros, except for $A_{\mathrm{add}\,4,3} = A_{\mathrm{red}\,4,2}$ and $A_{\mathrm{add}\,5,3} = -A_{\mathrm{red}\,5,1}$. By reading the entries of the system matrix the resulting spring damper element with backlash will have the same stiffness constant as the original spring damper element.
4. The non-linear system function

$$\boldsymbol{f}_{\mathrm{add}} = (0, 0, f_\lambda, A_{\mathrm{red}\,4,5} f_\lambda, -A_{\mathrm{red}\,5,4} f_\lambda)^{\mathrm{T}} \tag{15}$$

   is added to the previously defined subset of the system function. The third element f_λ is the non-linear equation (5). The last two entries incorporate the damping feedback. Because $\dot{\lambda}_k$ is not available as a state, its system function f_λ is used instead.
5. Finally the output matrix and output function must be updated because system matrix and system function have changed: $\boldsymbol{C}_{\mathrm{red}} = \boldsymbol{A}_{\mathrm{red}\,4-5}$, $\boldsymbol{g}_{\mathrm{red}} = \boldsymbol{f}_{\mathrm{red}4-5}$. The notation means that rows 4 and 5 are used.

The procedure can be repeated, adding backlash also to other spring damper elements.

Input and Output Selection Matrix: Next, the sensor and actuator positions are defined. The actuators are defined by the input selection matrix with N rows and as many columns as there are actuators. It is multiplied at the input of the state space representation. The sensors are defined by an output multiplication with the output selection matrix with $3N$ rows and as many columns as there are sensors. Input and output selection matrix are always required and they are therefore not explicitly included in the GP function set.

Current Control: Our approximation as a P_{T1} element can be regarded as a series connection from the left. The linear state space form of the current control is given by

$$\boldsymbol{A}_{\mathrm{L}} = -1/T_1, \quad \boldsymbol{B}_{\mathrm{L}} = 1/T_1, \quad \boldsymbol{C}_{\mathrm{L}} = 1, \quad \boldsymbol{D}_{\mathrm{L}} = 0. \tag{16}$$

Delay Time: In figure 4 the two blocks on the right and on the left are assigned to delay time. In the SISO case it is irrelevant which of them is actually used. For time-domain simulation the signal is resampled generating intermediate steps by means of linear interpolation.

3 GENETIC PROGRAMMING

The aim is to identify the structure and parameters of a given *reference system*, of which the time domain response to an excitation has been measured previously. The reference system is either a testbed or only another simulation model. With the previously defined submodels the process can be formalized as genetic programming. Submodels from the function set can be included or excluded automatically and no further adjustments are necessary. For each constructed simulation model the estimation parameters are identified with a global parameter optimization algorithm, before the fitness of the individual is evaluated.

3.1 Procedure of Genetic Programming

When contrasted with classical genetic programming as described for example in (Koza, 1994), a few special cases must be considered making this genetic programming *degenerate*. Usually the order of the nodes in a GP tree is part of the function definition and the output of one node is the input of another node. That is different here: Each node is assigned a *model type* and the number of nodes with a certain model type defines the multiplicity of this submodel. Connections between nodes do not represent the flow of information but merrily the genetic connection as bases on a chromosome, which is relevant for the mechanisms of evolution, see below. The number of branches originating from one node is chosen randomly.

The procedure is sketched in figure 5. In step 1 the function set is defined and for all estimation parameters included in the function set lower bound and upper bound are set. For the function set see section 2. It is further restricted according to these rules:

- There is only one sensor and one actuator at inertia 1.
- Only mass 1 may be subject to gravity and friction.
- Backlash occurs only between inertias 1 and 2.

These restrictions have been set in order to keep the construction rules, see below, manageable, and also to avoid the occurrence of practically indistinguishable models. In this first step the user has the opportunity to integrate prior knowledge by further restricting the function set and by defining application-specified parameter ranges.

Step 2 defines the initial population by creating trees with a randomly chosen number of nodes, ranging from 1 to the maximal possible number of submodels that leads to a model with all features enabled. For each node a model type is chosen by random from

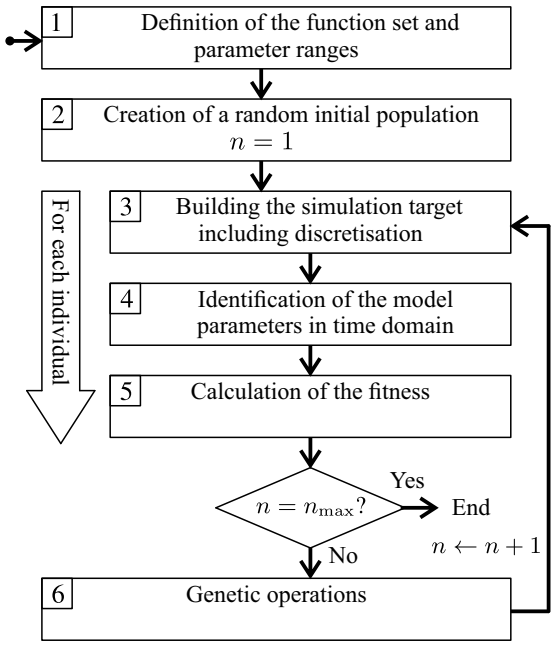

Figure 5: Flowchart of genetic programming.

a subset of the function set that leads to a syntactically valid model following the construction rules, see below. This process may be terminated early if the required number of nodes cannot be reached because the features exclude each other, e.g. a one-mass system cannot have backlash between masses. The origin of each new node is chosen randomly from one of the existing nodes. Initial parameters are chosen randomly within the permissible range.

In step 3 the time domain identification in Matlab is prepared by converting the state space form into system function and output function:

$$\dot{\boldsymbol{x}} = \boldsymbol{f}_{\text{sys}}(\boldsymbol{x}, u) = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}u + \boldsymbol{f}(\boldsymbol{x}, u) \tag{17}$$

$$y = \boldsymbol{f}_{\text{sys}}(\boldsymbol{x}, u) = \boldsymbol{C}\boldsymbol{x} + \boldsymbol{D}u + \boldsymbol{g}(\boldsymbol{x}, u). \tag{18}$$

Function $\boldsymbol{f}_{\text{out}}$ is discretised using Euler method and the delay time in included by means of linear interpolation.

In step 4 the set of estimation parameters of each model is identified by repeatedly simulating the model for different parameter sets. Penalty for the optimizer is the quadratic error between the output of the simulation model and the reference model. The optimizer *patternsearch* is used because it considers initial values of parameters as well as parameter bounds, while being robust to local minima. The initial system states are set corresponding to the real states of the testbed.

Step 5 is the calculation of fitness under consider-

ation of the model complexity:

$$fitness = \frac{1}{RSS \cdot (1 + k_P d)}, \tag{19}$$

where d is the number of estimation parameters, RSS is the residual sum of squares of the error in time domain. With k_P the trade-off between model complexity and accuracy can be adjusted. Here, it has been set to 2. Information criteria could be used instead, but they require knowledge of the measurement noise (James et al., 2013) and experiments have shown that they lead to rather high model complexities if the number of samples is large.

The genetic operations of step 6 are explained in the following section.

3.2 Genetic Operations

In step 6 of figure 5 genetic operations are performed on the current population to create the offspring. First, new individuals are created by means of *recombination*, also called *crossover* or also by means of *cloning*. The percentage that an individual is just cloned from the parent population is p_{clone}. So the number of cloned individuals stems from a Binomial distribution $N_{\text{clone}} \sim \mathrm{B}(n_P, p_{\text{clone}})$ with n_P the size of the population. A total of $N_P - N_{\text{clone}}$ individuals is generated by means of recombination: The first parent is copied up to a randomly chosen node. The branch behind this node is replaced by a branch of the second parent, the origin of which is also chosen randomly. Each new node is not added but dropped if a construction rule would be violated by adding it. At the end a node with a one-mass system is added if no multiple-mass system exists.

Choosing a parent either for cloning or for reproduction is done via *Roulette Wheel Selection* based on the fitness, see (Nelles, 2001).

After the new population has been created, different kinds of mutations are performed with a certain probability:

- **Point Mutation:** One randomly chosen node is assigned a random, new model type that satisfies the construction rules. The new estimation parameters are defined randomly within the permissible range.
- **Insertion Mutation:** A node is added, if possible, originating from a randomly chosen existing node. Its model type is random but satisfies the construction rules.
- **Deletion Mutation:** A randomly chosen node is deleted, again only if the construction rules are respected by the operation.
- **Chromosome Mutation:** In this mutation a node is also chosen randomly, but the mutation is not limited to the one node. Instead, the node and the whole branch originating from it is replaced by a new, randomly grown branch that satisfies the construction rules. The size of the resulting tree is set randomly, but not less than the original tree without the new branch. This kind of mutation is inspired by (Koza, 1994).

Construction Rules. Whenever a genetic operation is performed or when the initial population is created, the construction rules must be considered:

- Each model can be included a minimum of 0 times and a maximum of 1 time.
- Exactly one model of the general type 'multiple-mass system' must be included in all individuals.
- The model type 'one-mass system' is mutually exclusive with the model type 'backlash'.

It is believed that the mutations play an important role in the parameter identification. When nodes are copied from the parent population in cloning or recombination, the current parameter estimates are also copied and used as a starting point for the optimization in the next generation. As a consequence, previously found good solutions are passed on to the next generation shortening optimization time and preserving good solutions. However, the drawback is that the optimizer is subject to premature convergence due to this procedure. When new, random parameters are reintroduced by the mutations, local minima can be escaped. The same is true for the structure optimization.

4 SIMULATION RESULTS

A reference model is simulated and its output is used instead of real measurements. The reference model is a 2-mass resonating system with all three friction components but no gravity, current control or delay time, no noise. The excitation for the reference model and the optimization models is a torque signal that would lead to a standard industrial jerk-limited reversing motion if applied to a single moving mass, see figure 6.

As output the velocity at mass 1 is used. The lower and upper bounds of the estimation parameters are set to 50 %, resp. 200 % of the assumed values. Only the gravity torque has a range centred around 0 because gravity is allowed to be both positive and negative. The GP algorithm is run for 20 generations with 6 individuals. For the parameter optimization a maximum

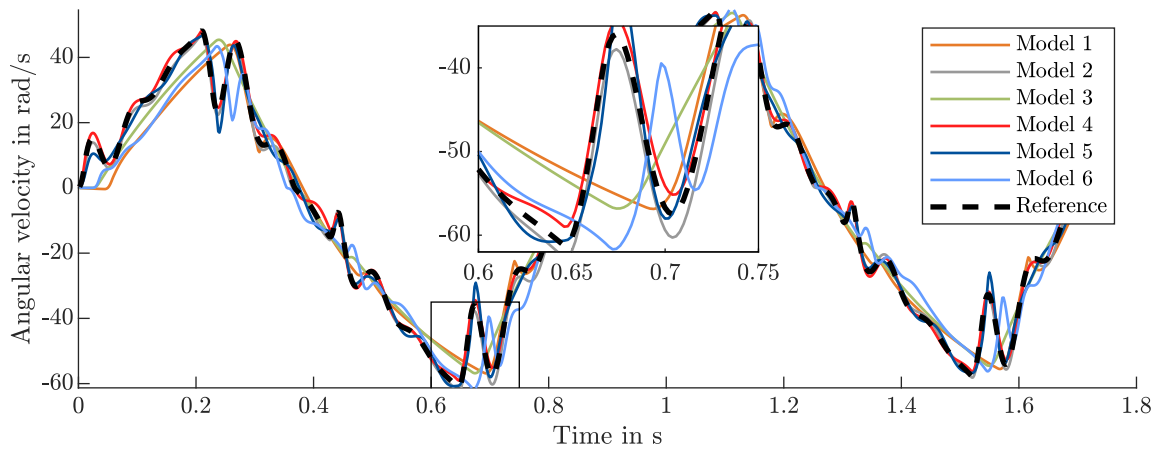

Figure 6: Output velocities of the 6 models of the initial population.

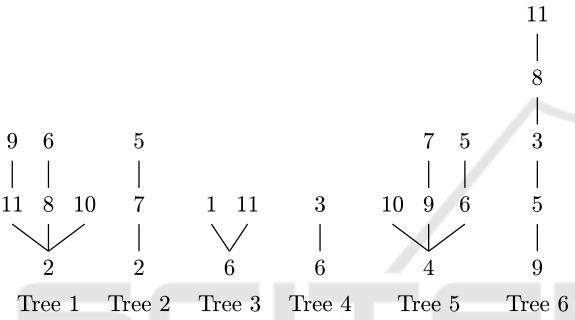

Figure 7: Initial population. 1: 1-mass system, 2: 2-mass system, 3: 3-mass system, 4: 4-mass system, 5: Coulomb friction, 6: viscous friction, 7: Stribeck friction, 8: gravity, 9: backlash, 10: current control, 11: delay time. The order has no meaning.

of 500 iterations is allowed for each individual of each generation.

Figure 7 shows the six trees of the initial population. Model types are represented by numbers, see figure description.

With this initial population the outputs in figure 6 have been retrieved after performing steps 3 and 4 in figure 5. It can be seen that some of the models fit better than others. In general, the differences from the reference model are obvious.

After 8 generations a population as in figure 8 has evolved, which includes the correct model (2,5,6,7) two times. The remaining four individuals look relatively similar which indicates that they have a common parent and mutations have caused the differences.

Figure 9 shows the outputs of the 6 models from figure 8. Most of these models are so close to the reference model trajectory that they are not distinguishable. Only model 2 with delay time, behaves clearly distinct.

Although the correct model has been found in generation 8, all 20 generations are performed, because in a real scenario the optimal model is unknown. Repeating the simulation shows that sometimes the resulting model has extra gravity or more than two masses. In this case the fitness is inferior but visually the trajectories can hardly be distinguished.

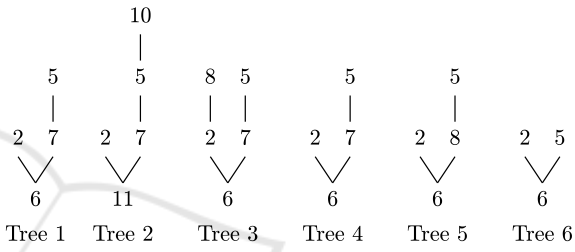

Figure 8: Generation 8. The numbers stand for the model types, see figure 7.

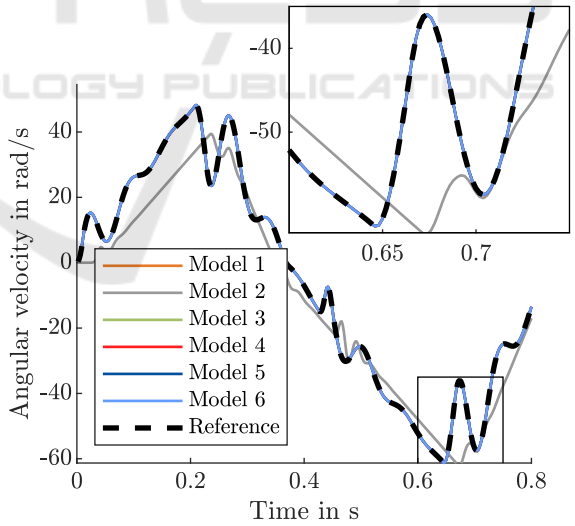

Figure 9: Output velocities of the 6 models of the eighth population. For space reasons only the first 0.8 s are displayed.

What has been shown for a 2-mass system with friction is also applicable to other compilations of the above candidate functions. The success depends on the estimation parameter ranges, the excitation trajectory and on the reference model parameters. E.g. a delay time of 50 ms can be detected more easily that a delay time of 1 ms.

5 DISCUSSION

Although the success is remarkable given the short excitation of $\approx 2\,\mathrm{s}$ and the difficulty of deriving knowledge about the inner structure from an observation of the input-output behaviour, the results of the method should be interpreted with caution. Clearly, the distinguishability of structures depends on many factors and it has not been proven that it is given for all candidate models considered here. In fact, the simulations have shown that repetitions can lead to different results, which may also be enforced by the inherently random nature of the optimization. As a mitigation, the small numbers of 20 iterations and 6 individuals could by increased. Also, an explicit separation into training and test data could improve the reliability of the result. But still, the veracity of the output models should be reviewed cautiously before any conclusions are drawn.

Another limitation of the approach is that it incorporates only little prior knowledge and accordingly the resulting models can only be relatively simple. Application-specific knowledge about special effects, such as position dependencies, non-linear springs etc. cannot readily be included. As a consequence, several restrictions of diversity had to be introduced such as the restriction to one external delay, or to collocated multiple-mass systems.

It must however be stressed that the intend of this paper was not to identify a unique state space description, which would be impossible, but only a model in a commonly acknowledged form with physically interpretable parameters. On that basis, the results can be seen as a success.

The GP algorithm described here is unusual in the way that the exact order of notes has no influence on the resulting model. So the benefit of the tree structure over a simple list of functions is not fully utilized. But this is also the case in other applications of genetic programming, when e.g. the order of multiplication or summation elements is ”optimized” although it is irrelevant. The tree structure still develops a merit when mutations or crossover are performed.

6 CONCLUSIONS AND FUTURE WORKS

6.1 Conclusions

Dynamic structure identification by means of genetic programming has been extended to models of electric drives with hard non-linearities. The generated state space representation can readily be used for control design and analysis. Its parsimony is optimized, i.e. the number of estimation parameters is minimized while maintaining accuracy. The concept can be extended interchangeably.

Because of the intricate engagement of the various kinds of submodels into the overall model several additional grammar rules must be incorporated into the GP algorithm. Successful operation has been shown exemplarily but cannot be guaranteed because of the stochastic nature of the algorithm.

6.2 Future Works

Future works include the generalisation to other sensor and actuator configurations and the inclusion of branching multiple-mass systems instead of the purely linear ladder structure. The structure identification could be carried out on the basis of adding and removing single spring-damper elements. Furthermore, a systematic tuning of the excitation signal to excite all parameters equally well is conceivable.

ACKNOWLEDGEMENTS

This work was sponsored by the German Forschungsvereinigung Antriebstechnik e.V. (FVA).

REFERENCES

Berglund, E. and Hovland, G. E. (2000). Automatic elasticity tuning of industrial robot manipulators. In *Proceedings of the 39th Conference on Decision and Control*, volume 5, pages 5091–5096. IEEE.

dos Santos Coelho, L. and Pessôa, M. W. (2009). Nonlinear model identification of an experimental ball-and-tube system using a genetic programming approach. *Mechanical Systems and Signal Processing*, 23(5):1434–1446.

Gray, G. J., Weinbrenner, T., Murray-Smith, D. J., Li, Y., and Sharman, K. C. (1997). Issues in nonlinear model structure identification using genetic programming. In *Genetic Algorithms in Engineering Systems: Innovations and Applications*, volume 446, pages 308–313.

James, G., Witten, D., Hastie, T., and Tibshirani, R. (2013). *An introduction to statistical learning*. Springer.

Koza, J. R. (1994). *Genetic programming II*, volume 17. MIT press Cambridge, MA.

Marenbach, P., Bettenhausen, K. D., and Cuno, B. (1995). Selbstorganisierende Generierung strukturierter Prozeßmodelle. *at-Automatisierungstechnik*, 43(6):277–288.

Nelles, O. (2001). *Nonlinear system identification: from classical approaches to neural networks and fuzzy models*. Springer-Verlag, Berlin.

Nordin, M., Galic', J., and Gutman, P.-O. (1997). New models for backlash and gear play. *International journal of adaptive control and signal processing*, 11(1):49–63.

Schütte, F. (2003). *Automatisierte Reglerinbetriebnahme für elektrische Antriebe mit schwingungsfähiger Mechanik*. Shaker.

Toropov, V. and Alvarez, L. (1998). Approximation model building for design optimization using genetic programming methodology. In *7th Symposium on Multidisciplinary Analysis and Optimization*, pages 490–498.

Walter, E., Lecourtier, Y., and Happel, J. (1984). On the structural output distinguishability of parametric models, and its relations with structural identifiability. *IEEE Transactions on Automatic Control*, 29(1):56–57.

Winkler, S., Affenzeller, M., and Wagner, S. (2004). *Identifiying nonlinear model structures using genetic programming techniques*. Citeseer.

Witczak, M., Obuchowicz, A., and Korbicz, J. (2002). Genetic programming based approaches to identification and fault diagnosis of non-linear dynamic systems. *International Journal of Control*, 75(13):1012–1031.

Zemke, S. (2012). *Analyse und modellbasierte Regelung von Ruckelschwingungen im Antriebsstrang von Kraftfahrzeugen*. PhD thesis, Leibniz Universität Hannover.