TALLINN UNIVERSITY OF TECHNOLOGY

Faculty of Information Technology

Department of Computer Control

Alexander Taranenko

Nonlinear Model Database for Education and Research

Bachelor's Thesis

Supervisor(s): Aleksei Tepljakov,

Juri Belikov,

Department of Computer Control,

Tallinn University of Technology

Tallinn 2016

Declaration: I hereby declare that this Bachelors thesis, my original investigation and achievement, submitted for the Bachelors degree at Tallinn University of Technology, has not been submitted for any degree or examination.

Deklareerin, et käesolev bakalaureusetöö, mis on minu iseseisva töö tulemus, on esitatud Tallinna Tehnikaülikooli bakalaureusekraadi taotlemiseks ja selle alusel ei ole varem taotletud akadeemilist kraadi.

Alexander Taranenko

Date: ....

Signature: ....

Contents

Abstract

Thesis topic

The purpose of this work was to create a database of nonlinear models, which included the models themselves, classification of the models, description of the key parameters and a reference to the paper of origin for the model. The system can be used to conveniently find examples to use in educational or research purposes based on specific criteria, as well as provide the source scientific paper.

The main objectives of this bachelor's thesis are to describe the theoretical information that was used during the classification of the models, the implementation and the structure of the database, and then discuss the possibilities for use of the database in education and academia by describing the process of taking the theoretical models from the database and creating a model file in Simulink, which can be later used for simulation purposes, although creating a working model for every entry in the database is out of scope of this thesis.

The thesis is in English, and contains 37 pages of text, 3 chapters, 2 tables and 22 figures.

Kokkuvõte

Käesoleva töö eesmärk oli luua mittelineaarsete mudelite andmebaasi, mis sisaldab mudeleid, mudeli klassifikatsioone, nende tähtsamate parameetrite kirjeldust ja päritoluartikklite viiteid. Süsteemi saab kasutada, et leida sobilikke näiteid haridusliku või uurimistöö eesmärkide jaoks konkreetsete kriteeriumite alusel. Andmebaasi samuti sisaldab teadusliku töö allikaid, kust näited pärinevad.

Selle bakalauruse töö peamine ülesanne on kirjeldada teoreetilist informatsiooni, mida oli kasutatud mudelite klassifikatsioonide jaoks; kirjeldada andmebaasi struktuuri ja kuidas rakendada seda praktikas. Samuti oli töö eesmärks arutada võimalusi kuidas kasutada andmebaasi hariduslikel ja akadeemilistel eesmärkidel, seletades lahti teoreetiliste mudelite võtmist andmebaasist ning sellega luues mudeli faili Simulink-is. Mudeli fail, mis oli loodud Simulink-is, saab hiljem kasutada simulatsiooni eesmärkidel. Mudeli loomine iga sisestuse jaoks andmebaasis ei sisaldu lõputöö teemas.

Lõputöö on Inglise keeles ja sisaldab kirjateksti 37 leheküljel, 3 peatükki, 2 tabelit ning 22 joonist.

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Goals of the work

During the process of writing an academic paper in the field of Control Theory, it if often needed to take an example model with specific properties and use it for demonstration of the principle or method described in the paper. Similarly, in education, when teaching Control Theory there is constant need for example models for illustrative purposes or as work assignments for simulation. Even though there exists an enormous amount of scientific papers that have such example systems in them and there are many resources that can provide an index and filtering criteria to find articles on the right topic, none of them have any information about the models that are used in these papers or if there are examples there in the first place. And even if the researcher already has a list of articles that have models which fit the general criteria he is looking for, there is still a need to open each individual paper, search through it to find the model, and then determine if it would be acceptable.

The goal of this thesis is to create a database that would provide a solution for the problems described above. Users of the database should be easily filter out only the relevant types of models and choose a suitable one, see all of the physical parameters and constants, as well as has the full information on the original article.

1.2 Thesis Outline

Chapter 1 of the thesis is the introduction for the work, covering the goals and the purpose of the work. It had all of the required meta information, such as the table of content, thesis outline, list of Figures etc.

In Chapter 2 the reader is presented with the theoretical background for this thesis. The method of database classification and the definitions of the terms used are provided, the database structure is presented and explained, including the filtering done in the front-end of the database.

In Chapter 3 the usefulness of the database is confirmed with three examples by taking models from the database and building simulatable objects in Simulink.

Chapter 2

Database creation

2.1 Classification

Any physical, biological or information system can have a mathematical representation—a model—that provides an overview of the system's behavior. In this work, the models from the articles were classified by a variety of parameters, including, but not limited to form, linearity, type, time domain, order etc.

There are several forms that can be used to present a model, and each of them have several advantages to the others, depending on the application and the model itself. In the database, the models are broken into three groups—input/output, state-space and transfer function.

2.1.1 Form

Input-output is the simplest way to present a system, and is most useful for the special class of linear, time-invariant systems. It is a view that represents a system as a series of components that are interconnected through their inputs and outputs, which allows complicated systems to be represented in a clear and more manageable fashion. In this form, the model is described with differential equations of the inputs and the outputs, with all other variables removed. The general form for a single-input, single-output model is described by (2.1) and (2.2):

$$f(y, \dot{y}, \ddot{y}, ..., y^{(n)}) = f(u, \dot{u}, \ddot{u}, ..., u^{(m)}) \tag{2.1}$$

$$a_0 y + a_1 \dot{y} + a_2 \ddot{y} + ... + a_n y^{(n)} = b_0 u + b_1 \dot{u} + b_2 \ddot{u} + ... + b_m u^{(m)}, \tag{2.2}$$

where $y^{(n)} = \frac{\mathrm{d}^n}{\mathrm{d}t^n}$ and $u^{(m)} = \frac{\mathrm{d}^m u}{\mathrm{d}t^m}$. Models presented in articles [4] and [19] can be considered typical examples of input-output models [2].

State-space representation is the most commonly used form used in academic papers on MIMO systems. State-space is a different approach to modeling, where the system is descried using states—a collection of variables that summarize the system. All state variables are gathered in the state vector $x \in \mathbb{R}^n$, the control variables in vector $u \in \mathbb{R}^p$, and the measured output by vector $y \in \mathbb{R}^q$, and the system is represented by equations (2.3) and (2.4):

$$\dot{x} = f(x, u) \tag{2.3}$$

$$y = q(x, u), \tag{2.4}$$

where $\dot{x} = \mathrm{d}x/\mathrm{d}t$. Both models used in [13] are prime examples of state-space models [8].

Transfer function is also a model representation form that is commonly used for Single-Input, Single-Output system analysis. A transfer function is the ratio of the output of the system to the input of the system, and is in the Laplace domain, unlike the previous two forms, which were in the time domain (2.5) [8].

$$H(s) = \frac{Y(s)}{U(s)}. \tag{2.5}$$

For typical transfer function models, please refer to [6] or [18].

2.1.2 Type

The second criteria by which models were categorized is type. The database allows one system to have multiple types at the same time to better describe the models. The categories used in the database are the following (every category is proceeded by a reference to an article that is an example of that kind of mode):

- Academic models are abstract mathematical models that do not necessary have a practical use or even a physical world counterpart, and are used in academic works as examples for scientific method applications [9].

- Biological models describe the processes inside various biological systems, like cells, homeostatic, endocrine and gene regulation [5].
- Educational systems are usually simplified models of real-world systems that are used in educational purposes to demonstrate properties of control systems, but cannot be implemented as a practical solution due to their limited nature [15].
- Industrial control systems usually implement distributed control design, where the data is gathered from spread out sources and fed into a remote control location. These systems are used in chemical, water and electrical industries [14].
- Mechanical models exists in the real world, actuators use physical forces to function, as well as the system itself is being acted upon by external forces, like gravity, wind, etc [10].
- Networked control systems are systems in which an additional element—the communication network—exists to close the feedback loop and enable the communication. At least on parts of such systems, signals travel as data packets [7].

2.1.3 Linearity

Another fundamental parameter of every system is its *linearity*. Linear systems have two properties, *superposition* and *homogeneity*, while the only unifying property of nonlinear system is the lack of one or both of those principles.

The principle of superposition states that a function of two combined inputs should be of equal value to the sum of the functions of individual inputs (2.6) [3].

$$f(x+y) = f(x) + f(y). \tag{2.6}$$

Similarly, homogeneity is the property of a function that states that scaling a function by a constant will be equal to the function of the input scaled by the same constant (2.7).

$$f(kx) = kf(x). \tag{2.7}$$

The article [12] presents a model that posesses both of these properties, which makes it linear by definition. A subset of Linear systems are Linear Time Invariant systems, or LTI. These systems have additional parameters that do not change over time.

2.1.4 Model order

Yet another important parameter of a system is its *order*, which is defined by the number of independent energy storage elements in the system. Practically, the definition of the order depends on which model is used:

- For models with transfer function representation, the order of the system is equal to the highest exponent in the function.
- For models described by state-space equations, the amount of state variables is equal to the order of the system.

2.1.5 Time domain

Time domain is an important property of a model, and two types can be defined—continuous time systems and discrete time systems.

They differ in the way that the signals are recorded, processed and communicated. Continuous time models are functions of time $t \in \mathbb{R}$, while $\mathbb{R}$ represents a set of real numbers, and are described by differential equations. Discrete time models are mathematical functions of the variable $k \in Z$, where Z is a set of integers, described by difference equations. In [16], the same model is presented both in continuous and discrete form.

A third option is also present in the database, called Hybrid time, for systems that exhibit both continuous and discrete behavior.

2.1.6 Time delay

Some systems, like the one used in [19], are time delayed, meaning the either the transmission or the feedback is delayed by a given period of time for continuous systems, or discretization steps for discrete systems. These models are tagged appropriately in the database.

2.2 Database Structure

The previous chapter described the parameters that were chosen for model classification. These parameters were used as foundation for the structure of the database used in this work. The structure itself is described in Figure 2.1.

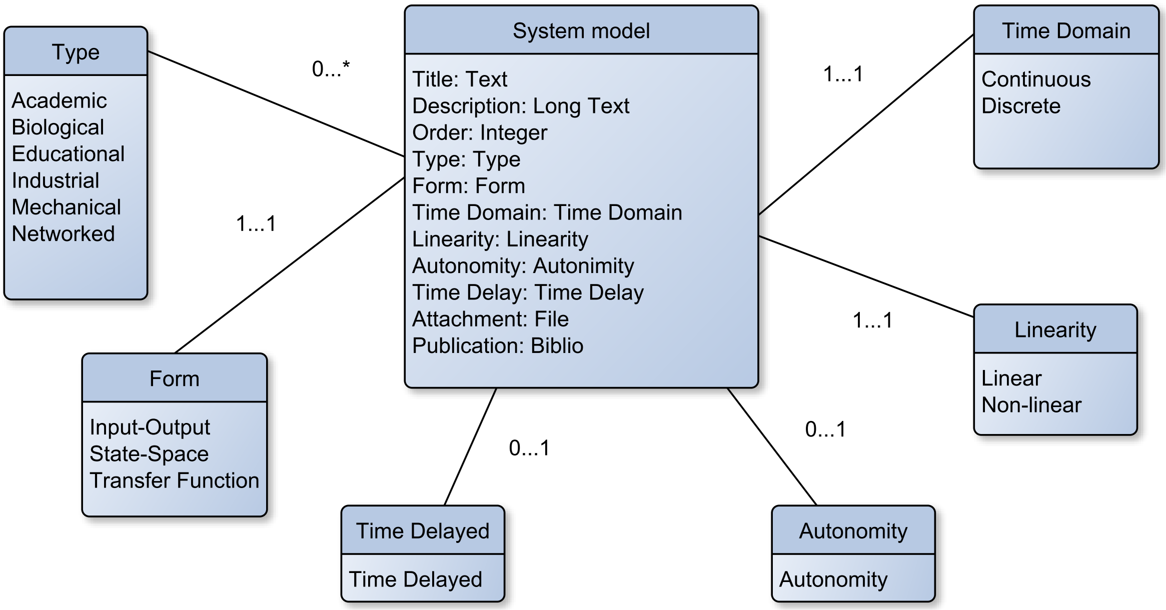

Figure 2.1: Database Structure

2.3 Filtering

The user interface of the database is a simple search form, which allows the user to to search by either title or description of a system, or filter the database based on any of the parameters that exist in the database. Underneath the search form, the first ten entries of the database are shown. After a search query is made, they will be replaced by the search results. Figure 2.2 is a presentation of the initial search conditions.

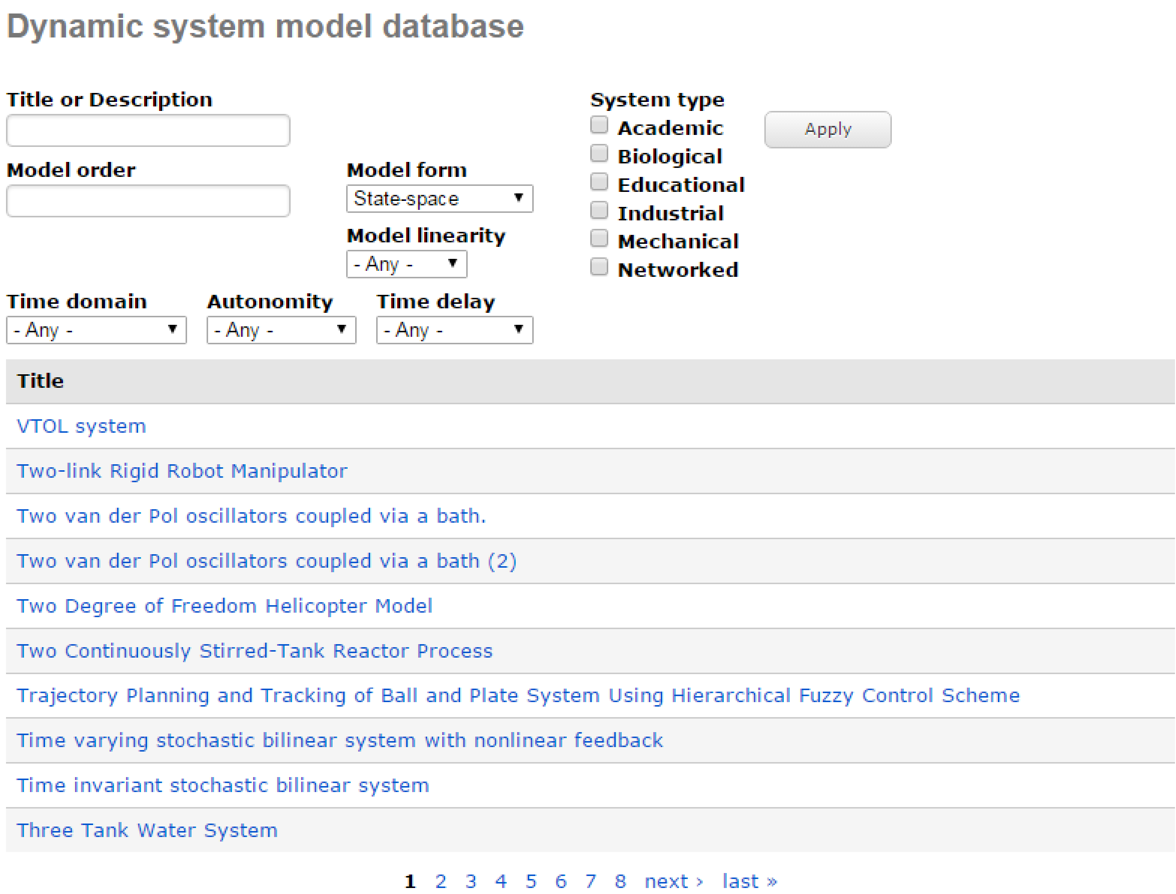

Figure 2.2: Front-end interface of the database

Chapter 3

Practical Application

As a practical part of the work, I will demonstrate how the database can be used in educational and research purposes. Several models will be selected from the database based on their metadata and built in Simulink. For the purposes of backwards compatibility, all models will be created using MATLAB version 2011b—it ensures that newer versions of MATLAB will be able to properly display the systems, and also support users with older versions of the program—the intended target audience.

The ambition is to create a Simulink file for every single model present in the database, but it is outside the scope of this work. The process of creating Simulink files will be described using four different models from the database. Using this work as reference, it will be possible for others to create the files as well and complete the database with little to none previous MATLAB experience.

There are several general steps for creating a model. First, the model is built from individual elements, after which they are grouped into a subsystem. The next step is to create a mask for this subsystem, in which the system parameters are described. A text description of the model is then added for the convenience of the end user.

3.1 Linear Transfer Function Model

The first model should be a linear system presented as a transfer function. In the database search parameters, I specify model form as “transfer function” and model linearity as “linear”. One of the search results is a model presented in [11]. The model has the form (3.1).

$$G(s) = \frac{c}{s(s+a)}, \tag{3.1}$$

where a and $c > 0$ are unknown constants.

First, a new Model needs to be created. It is achieved by selecting the **Model** option in the **New** sub-menu in the **File** menu as shown in Figure 3.1.

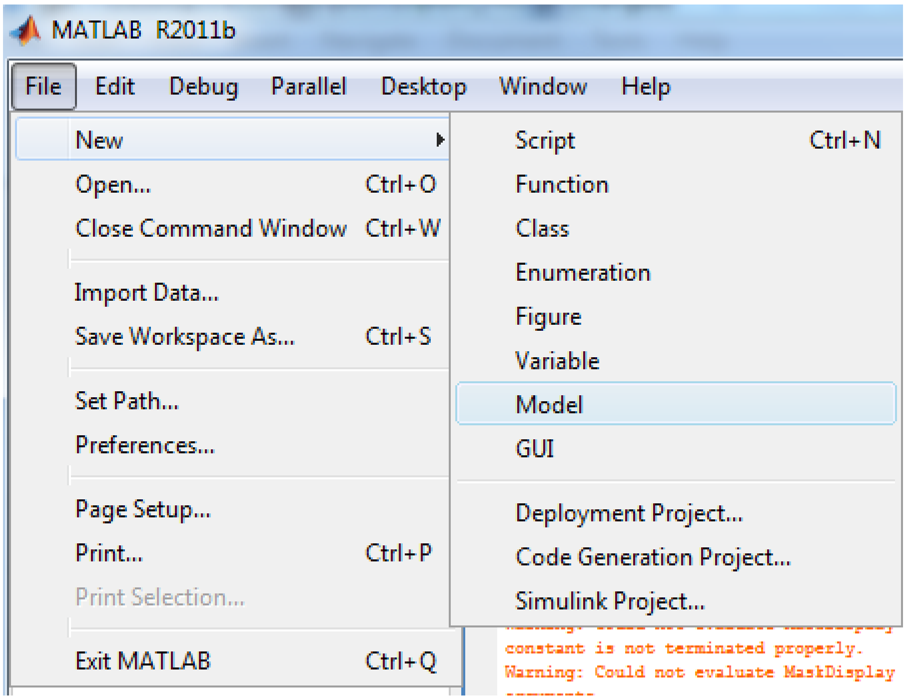

Figure 3.1: Creating a new model in Simulink

After that, the Library browser should be opened from the **View** → **Library Browser** option, see Figure 3.2.

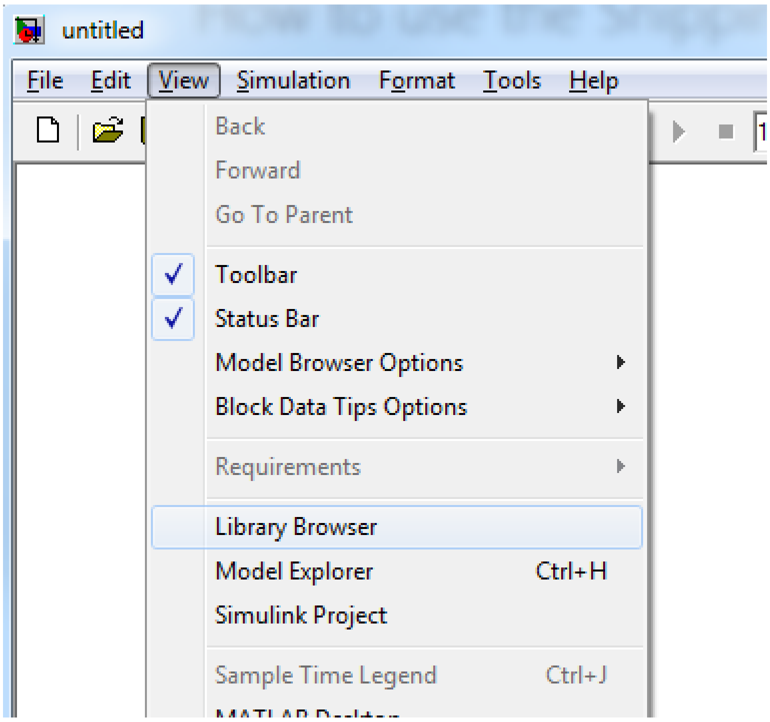

Figure 3.2: Opening the Library Browser

For this particular example, the first and only block that will be needed is the Transfer Function block. It is located in the the **Continuous** subsection of the Simulink library (Figure 3.3). Alternatively, it can be found by searching for "transfer fcn" in the search bar. It can be added to the model by selecting it and pressing CTRL+I or right clicking and selecting "Add to model".

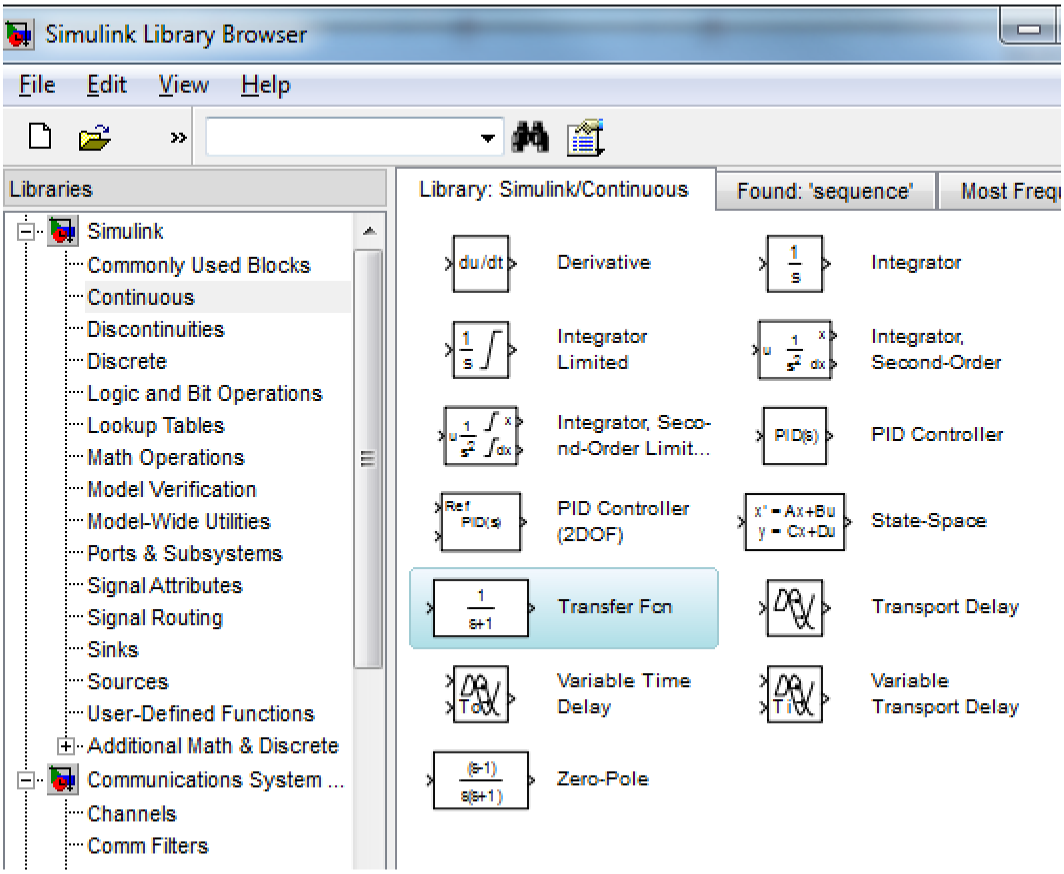

Figure 3.3: Transfer function block in the library browser

Once the transfer function block is added to the model, its parameters can be edited with a double click of the left mouse button. For this particular model, the numerator coefficient should be set to $\begin{bmatrix} c \end{bmatrix}$, while the denominator coefficients to $\begin{bmatrix} 1 & a & 0 \end{bmatrix}$. As a final touch, the transfer function name could also be changed. In this example, it was changed to "a linear system", as shown in Figure 3.4.

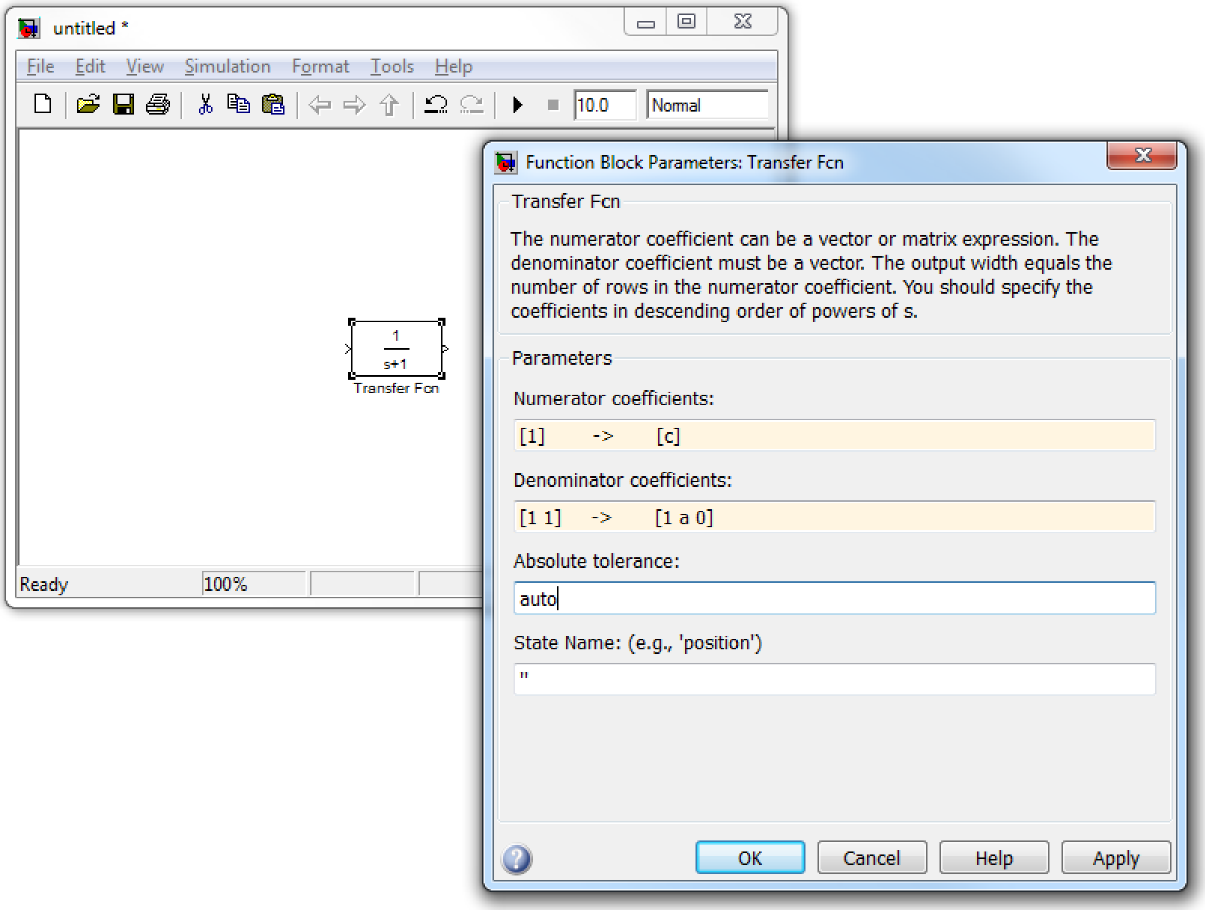

Figure 3.4: Setting up the parameters of the transfer function block

Now that the transfer function is present, it needs to be masked in order to allow semi-automatic control. Right-click the system and select the **Create Mask** option, which will open the mask creation menu. The **Icon & Ports** tab control the general appearance of the block. For a cleaner presentation, this particular system was modified with the command (3.2). This will create a $G(s)$ label for the block, indicating that it is a continuous transfer function.

$$disp('G(s)','texmode','on') \tag{3.2}$$

The **Parameters** tab is used to create the dialog window that appears when the block is clicked. The "Add parameter" button () is used to create new parameters. The **Prompt** field is the text that will be displayed next to the field, in **Variable** the destination variable is specified, and **Type** is the input method.

For this system, we need the user to enter values for the two constants, a and c. They were put into the Variable fields and "edit" type was selected. Prompts were also written to identify what fields would be entered. See Figure3.5 for the parameters tab and Figure3.6 for the finished result.

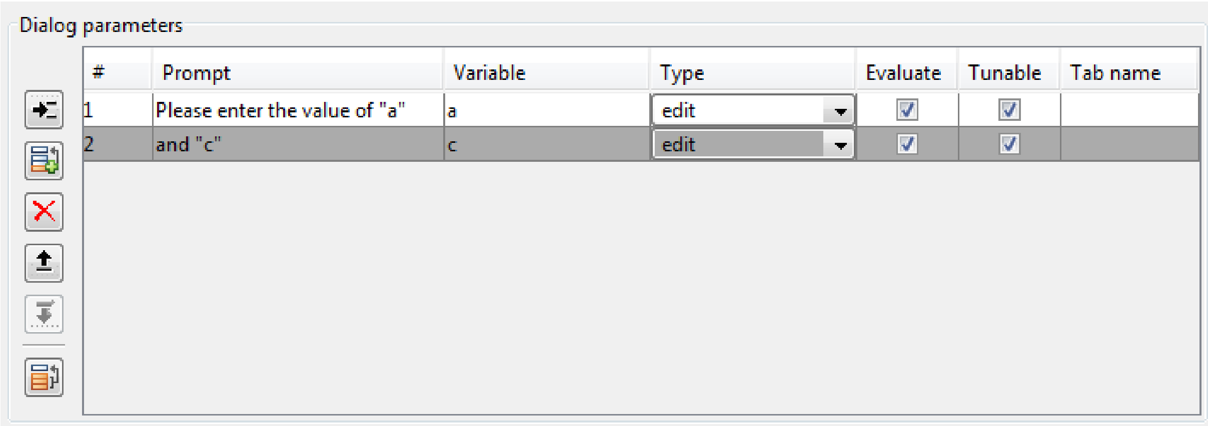

Figure 3.5: Parameters tab in the mask creation menu

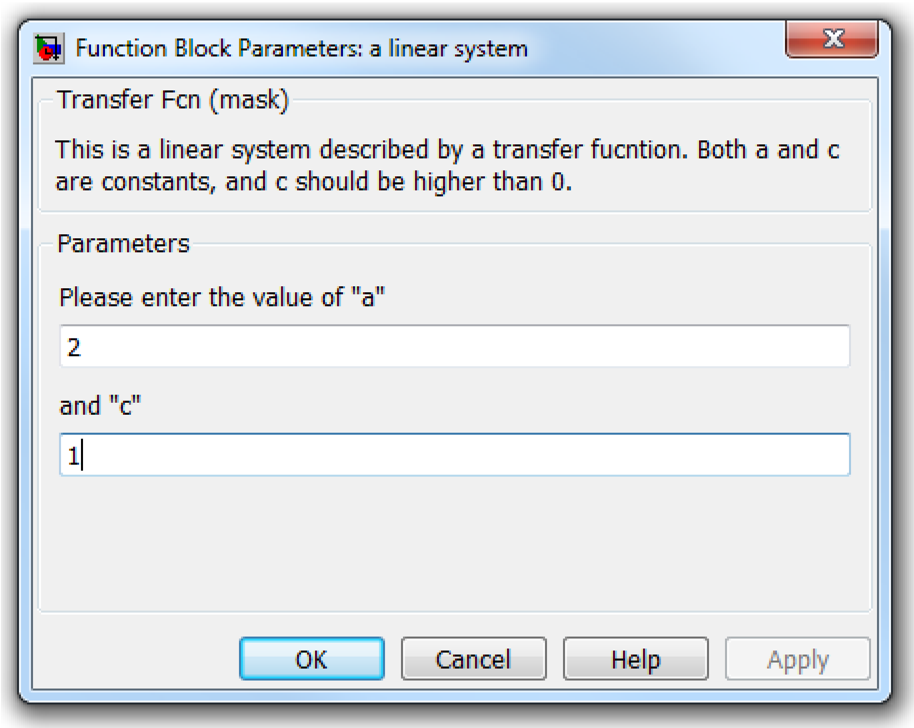

Figure 3.6: Finished dialog window after the mask is applied

The final step is to add a description. Simulink only offers limited TEX functionality, More on using TEX commands in annotations and the complete list of supported commands at [1]. The final result should look like Figure 3.7.

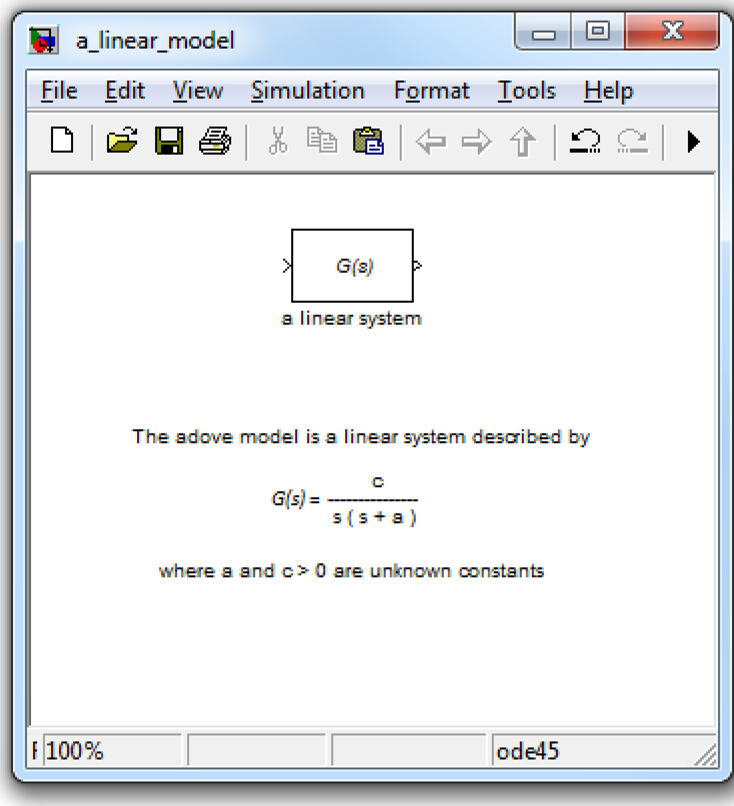

Figure 3.7: Finished model complete with description

To validate that the system is equivalent to the one presented in [11] and show that this solution is working, the system was simulated in a similar way to the original work. In the paper, the parameters were set to $a = 1$ and $c = 2$, and a bipolar fast rise, slow fall time pulse signal was applied to the input. The relevant excerpt from the article is presented in Figure 3.8.

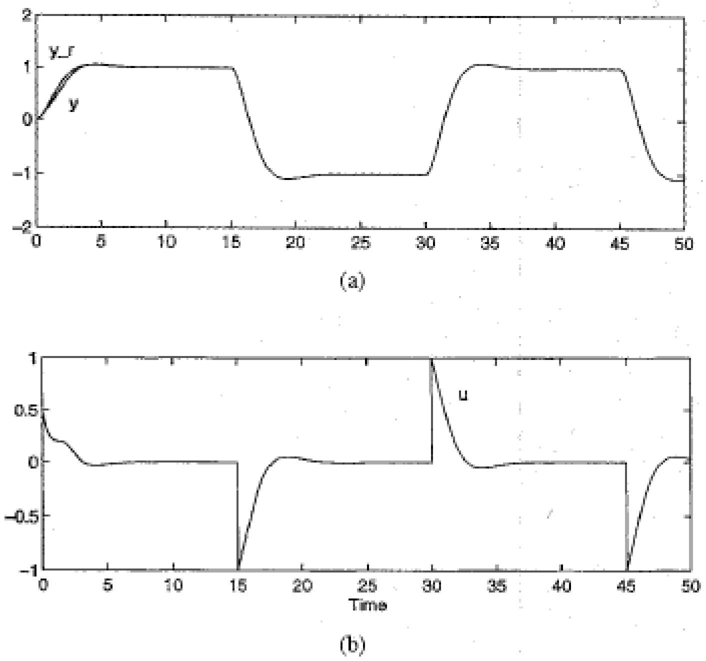

Fig. 1. Simulation results for Example 2 with $a = 1$, $c = 2$ (known), $\epsilon = 0.01$, and zero initial conditions. (a) Output y and reference signal y_r. (b) Control u.

Figure 3.8: Simulation conditions from the original paper

To verify the system, two additional components are needed. **Repeating Sequence Interpolated** block was added to the input of the system, and a **Scope** was connected to the output. A second input to the scope was created, to which the output of the signal generator was connected. The finished result is presented in Figure 3.9.

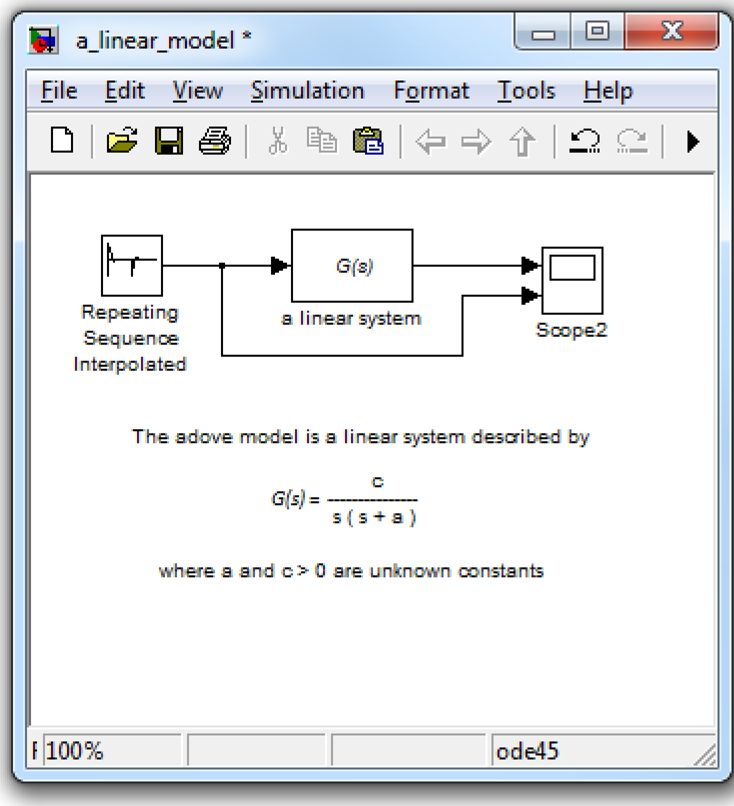

Figure 3.9: Schematic to verify the newly created system

The simulation results are presented in Figure 3.10. As it is clear to see, the results were almost identical to the system given in [11], from which it can be concluded that the system model is also valid.

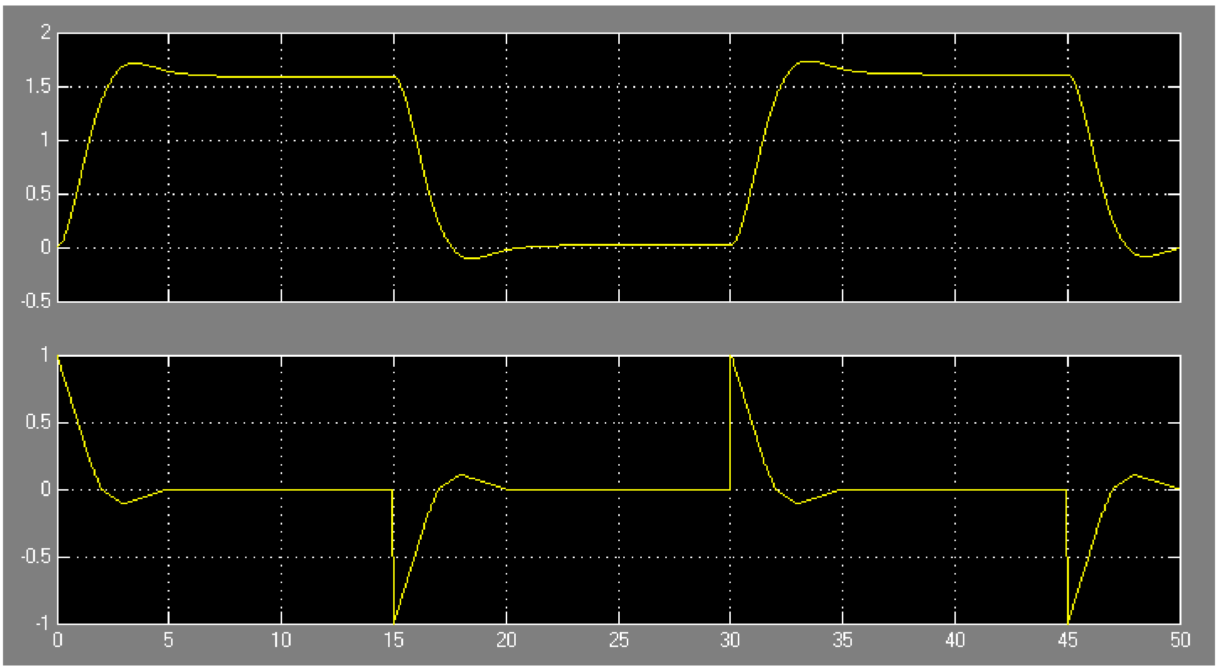

Figure 3.10: System simulation results

3.2 Linear State-Space Model

The next example is a 2-degree-of-freedom helicopter model presented in [20]—a linear state space model which has the matrices (3.3).

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -\frac{B_p}{J_{eq\_p}+m_{heli}{l_{cm}}^2} & 0 \\ 0 & 0 & 0 & -\frac{B_y}{J_{eq\_y}+m_{heli}{l_{cm}}^2} \end{bmatrix},$$

$$B = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ \frac{K_{pp}}{J_{eq\_p}+m_{heli}{l_{cm}}^2} & \frac{K_{py}}{J_{eq\_p}+m_{heli}{l_{cm}}^2} \\ \frac{K_{yp}}{J_{eq\_y}+m_{heli}{l_{cm}}^2} & \frac{K_{yy}}{J_{eq\_y}+m_{heli}{l_{cm}}^2} \end{bmatrix}, C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}, D = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}, \quad (3.3)$$

where $\theta(t)$ is the pitch angle and $\psi(t)$ is the yaw angle. The values of parameters used in this formula are written in the Table 3.1 below.

Table 3.1: Parameters of the experimental helicopter

K_{pp}	Pitch torque	0.204	$N.m/V$
K_{yy}	Yaw torque	0.072	$N.m/V$
K_{py}	Yaw on pitch torque	0.0068	$N.m/V$
K_{yp}	Pitch on yaw torque	0.0219	$N.m/V$
J_{eqp}	Total pitch moment of inertia	0.0384	$kg.m^2$
J_{eqy}	Total yaw moment of inertia	0.0432	$kg.m^2$
B_p	Pitch viscous damping	0.800	NN
B_y	Yaw viscous damping	0.318	NN
m_{heli}	Total moving mass	1.3872	kg
l_{cm}	Centre of mass length from pitch axis	0.186	m

To create this module in Simulink, a new blank model is created, and a State-Space block is added to it from the **Continuous** section in the Simulink Library browser. Because the module has two inputs and two outputs, two additional components are needed—a mux is placed before the input of the state-space block, and a demux—after the output. Both blocks are located in the **Signal Routing** section. The finished schematic is presented in Figure 3.11

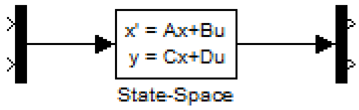

Figure 3.11: Individual blocks used for the 2DOF helicopter model

State matrices are described for the model by double-clicking the block. All formulas have to be written in-line (Figure 3.12).

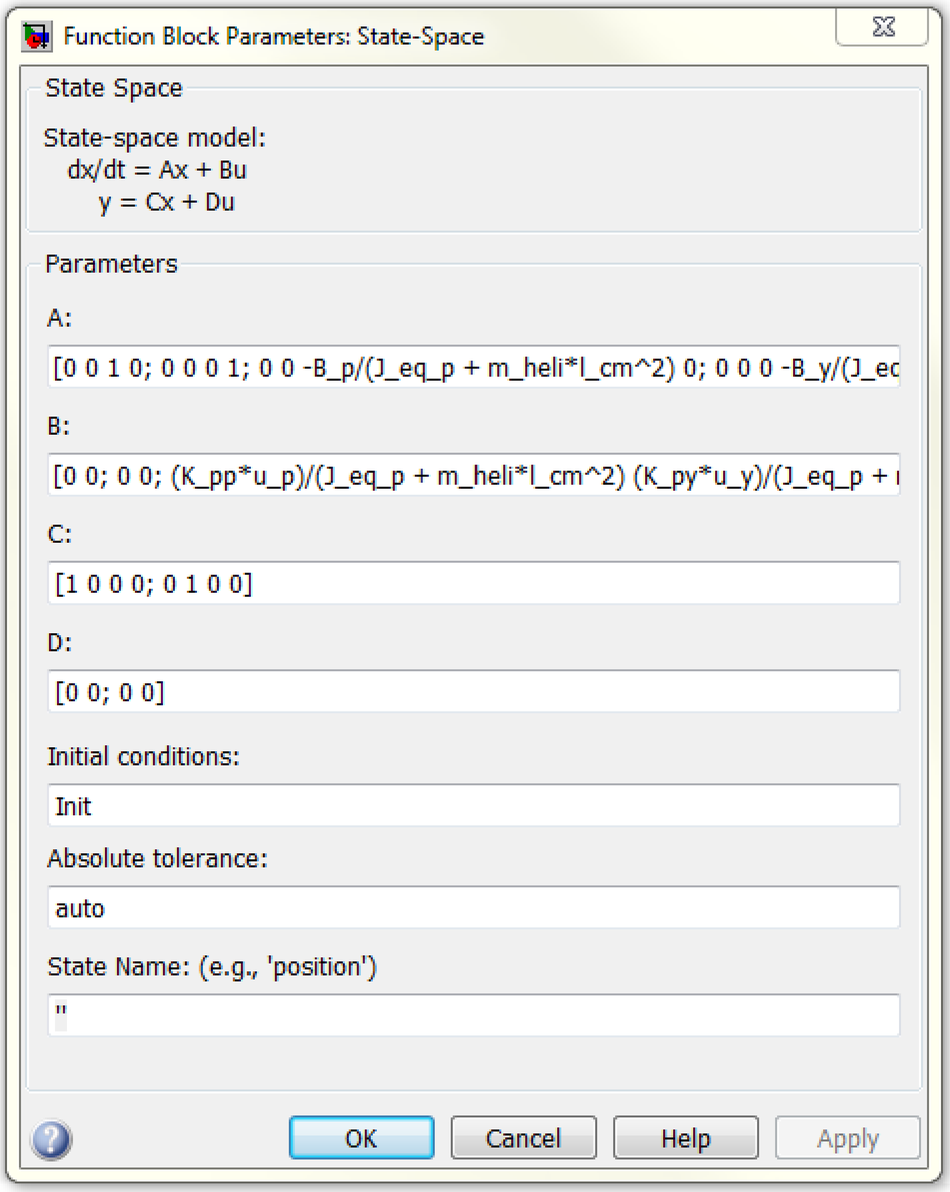

Figure 3.12: Setting up the parameters of the State-Space model

Just like with the previous example, all of the model's components are combined into a subsystem (select all components, **right click** → **create subsystem**) and a mask is applied to it (**right click** → **create mask**). In the Icons and Ports menu, in this case it makes more sense to title ports than to have just a general label (see Figure 3.13).

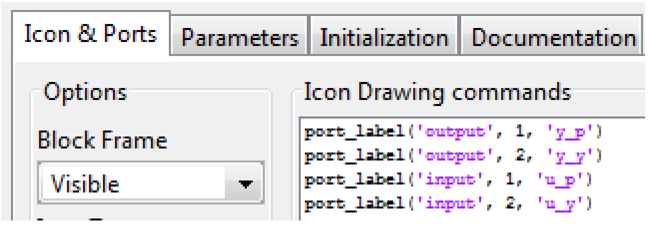

Figure 3.13: Icons and Ports tab of the mask creation menu

This model has quite a few modifiable parameters, so all of them are prompted from the user (see Figure 3.14).

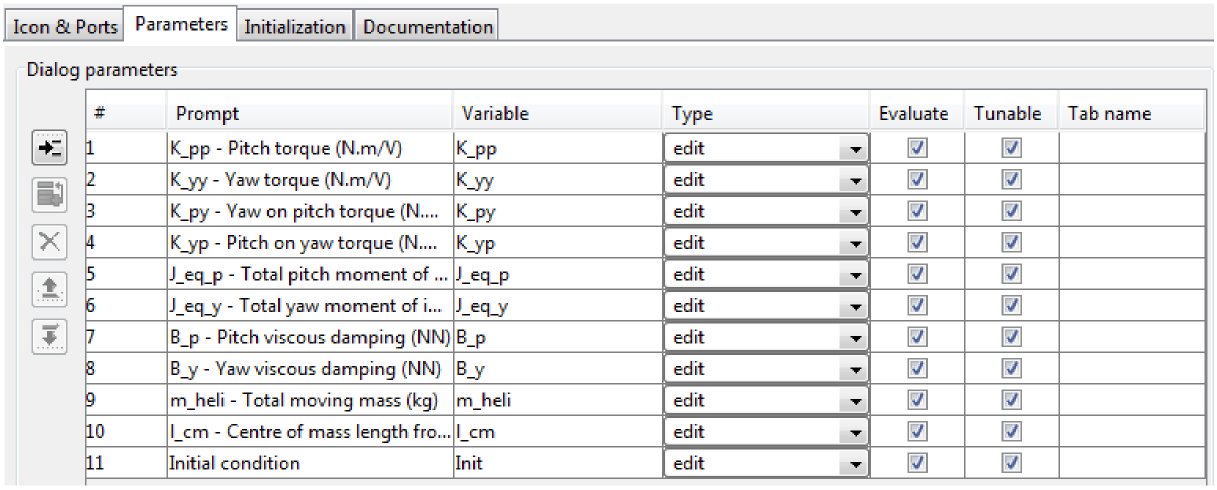

Figure 3.14: Prompts for the helicopter model created in the mask creation menu

Another additional step needed for this model is the initialization (see Figure 3.15).

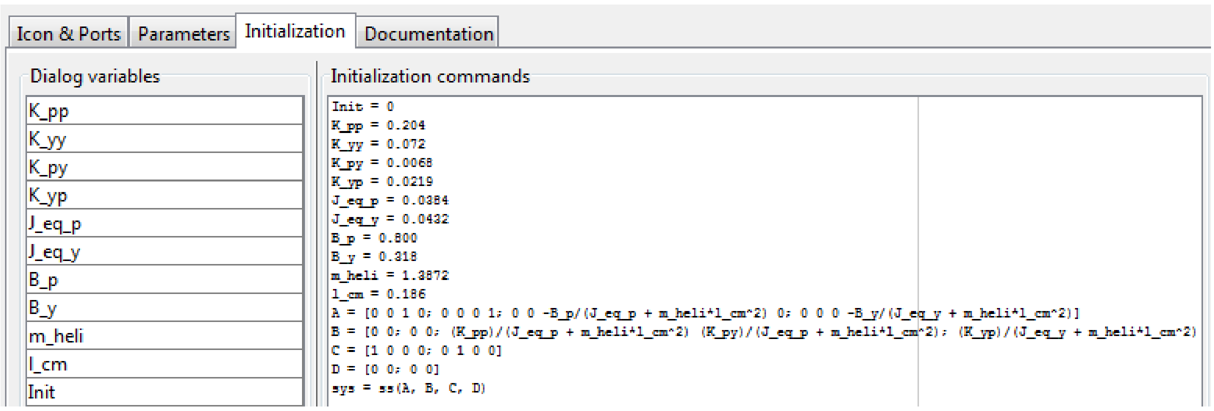

Figure 3.15: Initialization of the helicopter model in the mask creation menu

Finally, description text is added at the bottom of the newly created block to provide some additional description (see Figure 3.16)

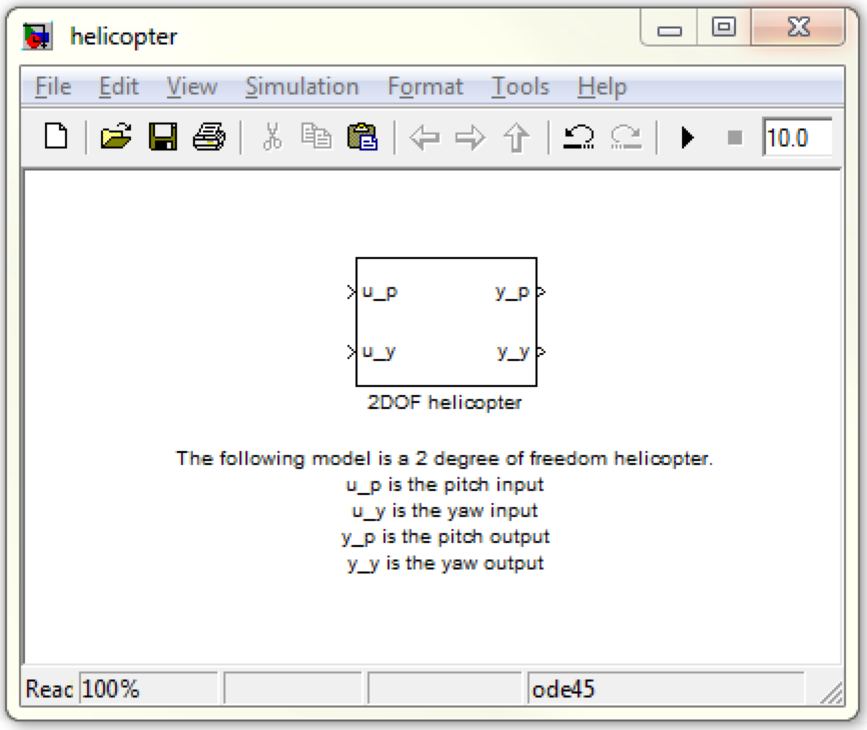

Figure 3.16: Finished view of the 2DOF helicopter model

3.3 Nonlinear Model

The final example is the inverted swinging pendulum described in [17], an inverted pendulum on a cart system. The system and its main parameters are shown on Figure 3.17. Its structure consists of a cart and pendulum where the pendulum is hinged to the cart via a pivot and only the cart is actuated.

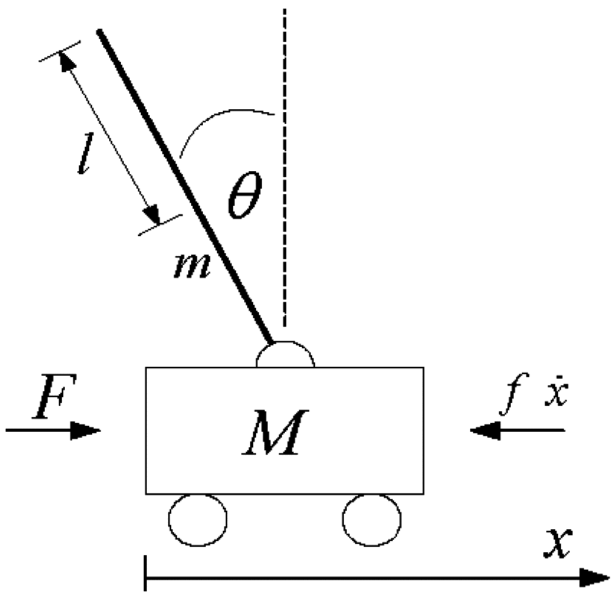

Figure 3.17: Main parameters of the inverted pendulum on a cart system

Lagrange's equations are applied with respect to θ and x coordinates. The nonlinear state space model of inverted pendulum on cart system can then be obtained as equations (3.4), (3.5), (3.6) and (3.7).

$$\dot{x_1} = x_2 \tag{3.4}$$

$$\dot{x_2} = \frac{\left(mg\sin x_1 + \dfrac{m\cos x_1 x_1(u - fx_4) - m^2 l x_2^2 \cos x_1 \sin x_1)}{(M+m)}\right)}{\left(\dfrac{4}{3}ml - \dfrac{m^2 l\cos^2 x_1}{(M+m)}\right)} \tag{3.5}$$

$$\dot{x_3} = x_4 \tag{3.6}$$

$$\dot{x_4} = \frac{\left(u - fx_4 + \dfrac{3}{4}mg\cos x_1 \sin x_1 - mlx_2^2 \sin x_1\right)}{\left((M+m) - \dfrac{3}{4}m\cos^2 x_1\right)}. \tag{3.7}$$

The state variables are consequently assigned as $x_1 = \theta$, $x_2 = \dot{\theta}$, $x_3 = x$ an $x_4 = \dot{x}$, and where the input u is the applied force F. The rest of the system parameters are described in Table 3.2.

Table 3.2: Parameters of the inverted pendulum on a cart

θ	Pendulum angle	rad
x	Cart position	m
M	Mass of the cart	kg
m	Mass of the pendulum	kg
l	Distance from turning center to center of mass	m
f	Cart's friction coefficient	kg/s
F	Force applied to the cart	N

As a nonlinear system, this one presented challenges in modeling. Instead of using a per-existing state space or transfer function blocks, all four equations are represented using simple mathematical operation blocks such as "add", "subtract", "product", "division", as well as specific functions such as "integrator", "constant", and using the "user-defined function" block for $\sin(\cdot)$ and $\cos(\cdot)$ functions. The finished schematic is shown in Figure 3.18. To simplify the readability, the schematic is color-coded.

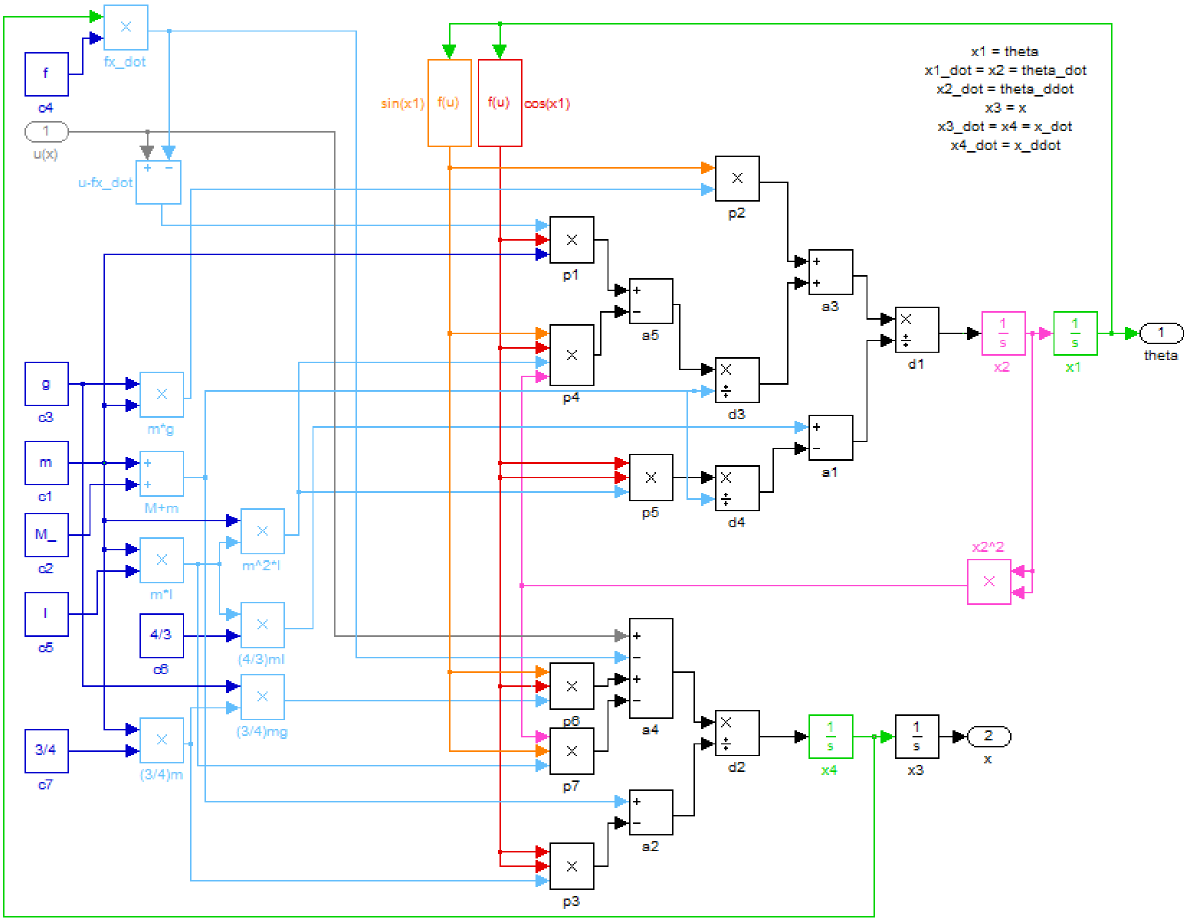

Figure 3.18: Inverted pendulum on a cart—Simulink schematic

Same as with two previous examples, the finished schematic is combined into a sub-system, and a mask is created with parameter initialization and a user-prompt. To verify the model, an input of a short impulse is sent to the system and the outputs are monitored in a scope. The end set-up is shown in Figure 3.19.

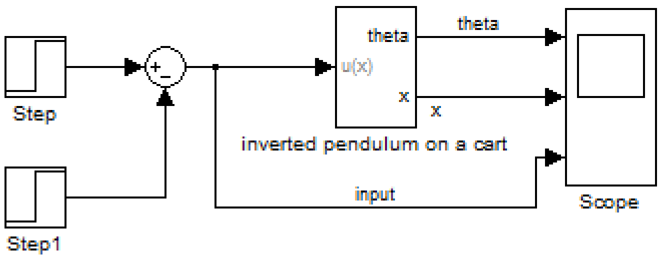

Figure 3.19: Inverted pendulum on a cart—finished model complete with verification blocks

The simulation results are shown in Figure (3.20). The model behaves as expected. Before the impulse, the pendulum is resting at the angle of 0 rad (upwards vertical position), and the cart position is also 0. When a positive force is applied, the pendulum starts to oscillate, almost reaches the value of 2π (a full circle), but not quite, then starts to rotate in the other direction, and goes back almost a full circle, but not reaching the original position. If the simulation runs for long enough, the final angle is π, which is vertically downwards. The cart, in the mean time, receives the positive force, moves forward a certain distance, then slows down and stops, only slightly moving forwards and backwards in response to the movement of the pendulum.

Based on the simulation result, it can be determined that the model was created correctly.

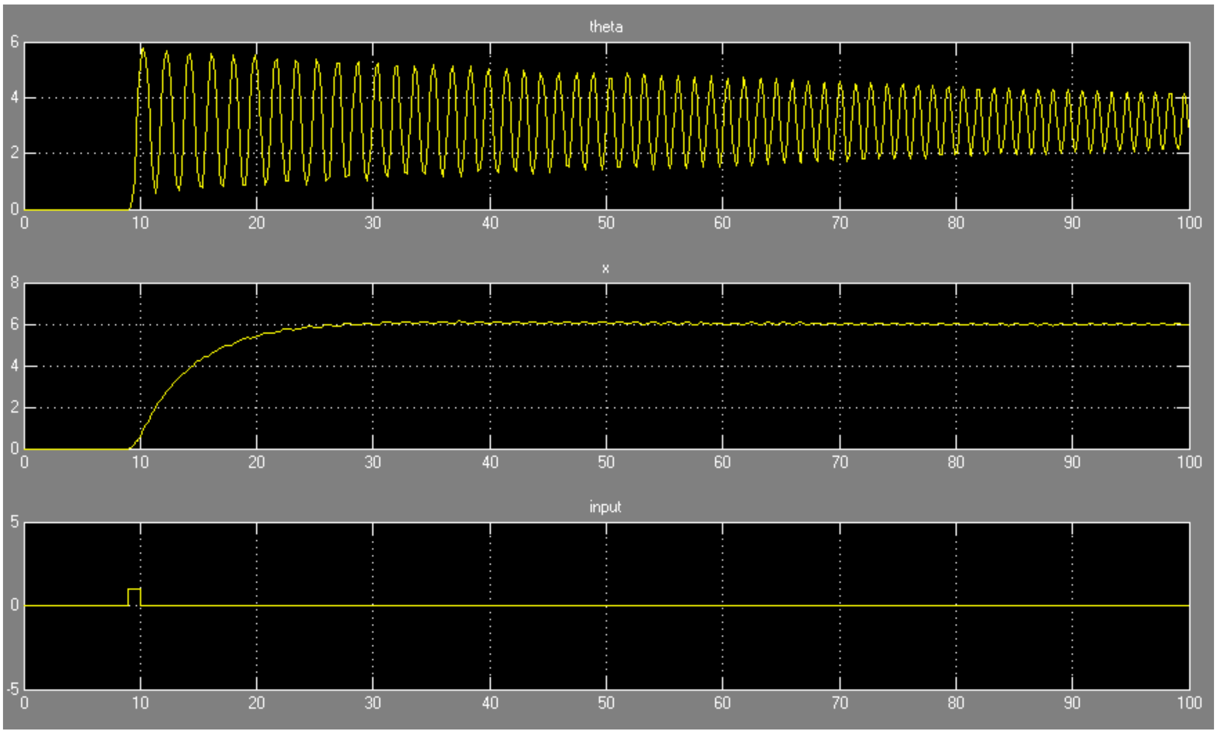

Figure 3.20: Inverted pendulum on a cart—simulation results of the original model

Conclusions

In the process of writing this thesis, the author has worked through almost 90 academic papers, most of which were written in a language of a much bigger complexity than initially anticipated. During the process of creating the database, the author learned to use the LaTeX document preparation system, as well as delve deeper into the world of nonlinear systems.

In the end, a fully functional database was developed that can be used in ways described in the introduction—the user of the database can search for a specific model based on a name or a needed parameter, get a list of models that all meet his requirement, and after selecting a specific model—get all of the necessary information about the model and the source article, as well as conveniently get a ".bib" file for the reference or a link to the original article itself.

This thesis can be viewed as a staring point for the database, which can from this point on be easily amended and updated by the staff of the Department of Computer Control.

Bibliography

[1] Use TeX Commands in an Annotation. `http://se.mathworks.com/help/simulink/ug/add-an-equation-to-an-annotation.html`. Retrieved 12.09.2015.

[2] Karl Johan Astrom and Richard M. Murray. *Analysis and Design of Feedback Systemscontrol*. CDS 101/110 Course Notes. California Institute of Technology, 2003.

[3] Karl Johan Astrom and Richard M. Murray. *Feedback Systems: An Introduction for Scientists and Engineers*. Princeton University Press, Princeton, NJ, USA, 2008.

[4] James F. Bartee and Christos Georgakis. Identification and control of bilinear systems. In *American Control Conference, 1992*, Chicago, Illinois, 06/1992 1992. IEEE, IEEE.

[5] Oana-Teodora Chis, Julio R. Banga, and Eva Balsa-Canto. Structural identifiability of systems biology models: A critical comparison of methods. *PLoS ONE*, 6:1–16, 10/2011 2011.

[6] H. Garnier, P. Sibille, and H.L. Nguyen. A new bias-compensating least-squares method for continuous-time mimo system identification applied to a laboratory-scale process. In *Proceedings of the Third IEEE Conference on Control Applications, 1994.*, Glasgow, 08/1994 1994. IEEE, IEEE.

[7] Ruben Garrido. Adaptive neural control of nonlinear systems. In Ieroham Baruch, Jose Martin Flores, and Federico Thomas, editors, *International Conference on Artificial Neural Networks - ICANN 2001*, Vienna, Austria, 08/2001 2001. Springer, Springer.

[8] Farid Golnaraghi and Benjamin C. Kuo. *Automatic Control Systems*. Wiley Publishing, 9th edition, 2009.

[9] Zhongsheng Hou and ShangTai Jin. Data-driven model-free adaptive control for a class of mimo nonlinear discrete-time systems. *IEEE Transactions on Neural Networks*, 22:2173–2188, 11/2011 2011.

[10] H.R Karimi and M.R.J Motlagh. Robust feedback linearization control for a non linearizable mimo nonlinear system in the presence of model uncertainties. In *IEEE International Conference on Service Operations and Logistics, and Informatics*, Shanghai, 06/2006 2006. IEEE, IEEE.

[11] H.K. Khalil. Adaptive output feedback control of nonlinear systems represented by input-output models. *IEEE Transactions on Automatic Control*, 41:177–188, 02/1996 1996.

[12] B. Kiss and E. Szadeczky-Kardoss. On-line time-scaling control of a kinematic car with one input. In *Mediterranean Conference on Control & Automation, 2007.*, Athens, 06/2007 2007. IEEE, IEEE.

[13] K. Kvaternik and A.F. Lynch. Global tracking via output feedback for nonlinear mimo systems. *IEEE Transactions on Automatic Control*, 56:2179–2184, 05/2011 2011.

[14] Hyeongcheol Lee. Robust adaptive fuzzy control by backstepping for a class of mimo nonlinear systems. *IEEE Transactions on Fuzzy Systems*, 19:265 – 275, 11/2010 2010.

[15] M. Nishi, M. Ishitobi, and S. Kunimatsu. Nonlinear sampled-data models and zero dynamics. In *International Conference on Networking, Sensing and Control, 2009. ICNSC '09.*, Okayama, 03/2009 2009. IEEE, IEEE.

[16] M. Nishi, M. Ishitobi, and S. Kunimatsu. Sampled-data model for nonlinear coupled Van der Pol oscillators. In *2011 Proceedings of SICE Annual Conference (SICE)*. IEEE, IEEE, 09/2011 2011.

[17] S Nundrakwang, T Benjanarasuth, J Ngamwiwit, and N Kominet. Hybrid controller for swinging up inverted pendulum system. In *2005 Fifth International Conference on Information, Communications and Signal Processing*, Bangkok, 2005. IEEE, IEEE.

[18] S. Villwock and M. Pacas. Application of the welch-method for the identification of two- and three-mass-systems. *IEEE Transactions on Industrial Electronics*, 55:457–466, 01/2008 2008.

[19] Y.K. Yeo and D.C. Williams. Adaptive bilinear model predictive control. In *American Control Conference, 1986*, Seattle, WA, 06/1986 1986. IEEE, IEEE.

[20] A.H. Zaeri, S.B. Mohd-Noor, M.M. Isa, F.S Taip, and A.E. Marnani. Disturbance rejection for a 2-dof nonlinear helicopter model by using mimo fuzzy sliding mode control with boundary layer. In *Third International Conference on Intelligent Systems, Modelling and Simulation (ISMS), 2012*, Kota Kinabalu, 02/2012 2012. IEEE, IEEE.