CREATING A SIMSCAPE HYDRAULIC LIBRARY FOR PHYSICAL MODELLING

Bernhard Heinzl[1], Günther Zauner[2], Felix Breitenecker[1], Matthias Rößler[1]

[1]Vienna University of Technology, Institute for Analysis and Scientific Computing
Wiedner Hauptstraße 8-10, 1040 Vienna, Austria
[2]dwh Simulation Services
Neustiftgasse 57-59, 1070 Vienna, Austria

bernhard.heinzl@tuwien.ac.at (Bernhard Heinzl)

Abstract

Modelling and simulation of physical systems is more than ever an important topic in scientific computing. There is not only a wide range of application possibilities in various branches of physics, but computer simulation can also be used for different kinds of tasks, like optimization and acceleration of development processes or finding design errors. New approaches based on physical networks are making it easier for modelers to implement physical models as it allows them to focus on the physical structure rather than the underlying mathematical equations.
This paper describes a way how one can create reusable components representing hydraulic elements for modelling and simulation of hydraulic systems with MATLAB/Simscape™ using the physical network approach. Therefore, it is also defined a hydraulic domain in Simscape, describing how energy is transmitted throughout the physical system.
Since the balance equations used for the hydraulic components are non-linear, whereas the balance equations of mechanical or electrical systems are typically linear, it is presented a fundamental distinction between these classical physical domains and the hydraulic domain. To verify the created Simscape library, the rear part of this paper shows an example of a typical hydraulic system test environment, which has been modeled and simulated in Simscape.

Keywords: Simscape, physical networks modelling, hydraulic library, Simscape language

Presenting Author's Biography

Bernhard Heinzl received the BSc degree in technical mathematics and in electrical engineering from the Vienna University of Technology in February 2010 and in August 2010, respectively. He is currently continuing his studies in technical mathematics and automation towards a MSc degree.

1 Introduction

Simscape extends MATLAB/Simulink™ with tools for physical modelling in various domains, such as mechanical, electrical or thermal environments. Unlike modelling with Simulink blocks, which represent mathematical operations and operate on signals, Simscape employs a physical network approach, where blocks directly corresponding to physical elements, such as motors, pumps or hydraulic cylinders, are connected by lines corresponding to the physical connections that transmit energy. Therefore, this approach describes the physical structure of a system rather than the underlying mathematical equations [1, 2, 3].

While Simscape provides libraries with fundamental building blocks [4], the user can create new libraries, models of physical components and new physical domains using the Simscape language, an object-oriented modelling language based on MATLAB [5].

2 Hydraulic Domain

Although there exists a predefined domain for hydraulic components in Simscape, it was more suitable for this work to define a separate hydraulic domain in order to be more independent regarding choice of domain variables and parameters. In the hydraulic domain, two variables are declared which represent the energy flow between the physical components. In the Simscape language, these variables are called `across` and `through` variables [2]:

- `through` variables are measured with a gauge connected in series to an element
- `across` variables are measured with a gauge connected parallel to an element

In our hydraulic domain, the `through` variable is represented by the volume flow with the SI unit cubic meters per second, and the `across` variable is defined as the pressure in the SI unit Pascal [6].

These variables are sufficient to describe the energy flow, however, it turned out to be advantageous to define the cross section of the ports as a further variable so that each port has a specified cross section and directly linking components with different cross sections is prohibited. Otherwise, such a situation could have violated the law of conservation of energy because the absolute pressure is equal at each connected node (across variable), whereas the kinetic energy is different for different cross sections due to different flow velocities. However, to connect elements with different cross sections, one can use the `diffusor` element, which is described in section 3.2 and which causes a simultaneous change of pressure and flow velocity as the cross section continuously changes.

Furthermore, the domain defines the following constant parameters, which can be used in the equations defining the component behaviour:

- acceleration of gravity: $g = 9.81\,\mathrm{m/s^2}$,
- fluid density: $\rho = 850\,\mathrm{kg/m^3}$.

These definitions have to be implemented in the so-called domain file using the Simscape language [5]. This domain file has the Simscape-typical extension `.ssc` and has to be placed somewhere in the working directory.

To specify that a certain node of a component model belongs to this hydraulic domain, one has to point to the directory containing the domain file by using the following code segment in the declaration section of the component file (cf. [2]):

```
nodes
    port = hydraulic.hydraulic;
end
```

This means that the domain file `hydraulic.ssc` is located in the subdirectory `hydraulic` of the working directory.

3 Physical Components

Each component is implemented in a different so-called component file. Like the domain file, it also has the extension `.ssc` and has to be located in the working directory.

This file also contains an `equation` section with all necessary equations for describing the component behaviour. According to the acausal modelling approach, these equations are represented as acausal implicit differential algebraic equations (DAEs).

The components implemented in the Simscape hydraulic library are shown with their Simscape blocks in Fig. 1.

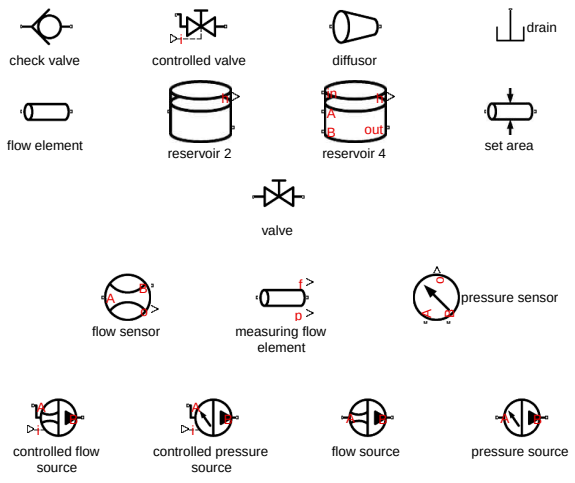

Fig. 1 Basic hydraulic elements, sensors and sources in the Simscape hydraulic library

These Simscape blocks include basic hydraulic elements, like valves, tanks or a pipe, as well as hydraulic

flow and pressure sources and also hydraulic sensors for measuring pressures and volume flows. These components and the necessary equations are described in the following sections.

3.1 Flow Element

One of the most important components implemented in the hydraulic library is the `flow element`. It represents a pipe with constant cross section A and is used to connect other hydraulic components. It defines a dependency between the volume flow q through the pipe and the pressure drop p along the pipe.

To derive the essential equation, we consider a hydraulic system with a pipe connected to a tank, see Fig. 2.

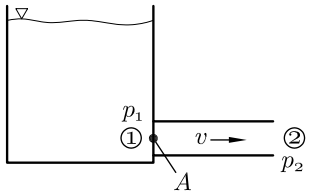

Fig. 2 Basic hydraulic system: Fluid flowing out of a tank with velocity v through a pipe with constant cross section A. 1 and 2: Points where Bernoulli's principle is applied.

We apply Bernoulli's principle in the form (cf. [7])

$$p_1 + \rho g h_1 + \frac{\rho v_1^2}{2} = p_2 + \rho g h_2 + \frac{\rho v_2^2}{2}, \qquad (1)$$

with the gravity acceleration g, the fluid density ρ, the pressure p_i, the fluid flow speed v_i and the elevation above a reference plane h_i of the observed point, $i \in \{1, 2\}$. With the assumptions $v_1 = 0$ and $h_1 = h_2$, we obtain

$$p_1 = p_2 + \frac{\rho v_2^2}{2}, \qquad (2)$$

and by using the relation

$$v = \frac{q}{A} \qquad (3)$$

with the cross section A, the final equation results in

$$\Delta p = p_1 - p_2 = \frac{\rho}{2} \left(\frac{q}{A} \right)^2. \qquad (4)$$

Since Simscape is employing acausal modelling, the Simscape solver can also use Eq. (4) in the form

$$q = A \sqrt{\frac{2\,\Delta p}{\rho}}, \qquad (5)$$

to calculate the pressure drop Δp or the volume flow q, depending on which variable is predefined from the remaining components in the network. Given the fact that these equations are non-linear, the Simscape solver produced errors during our simulation runs when we tried to use either Eq. (4) or Eq. (5) for the `flow element`. These errors could only be eliminated by explicitly implementing both equations and letting the user decide during the modelling process which equation is to be used, forcing a causal implementation of the equations describing the `flow element`. This was realized by defining a boolean parameter `c`, which has to be set by the user and whose two different values correspond to the two equations, i.e.:

```
if c == 0
     q == A*sqrt(2*p/rho);
else
     p == rho/2*(q/A)^2;
end
```

The value of cross section A can be set via a block parameter and because the cross section is also a variable in the hydraulic domain, the parameter furthermore defines the cross sections of the components connected directly to a `flow element`, so that linking elements with different values of cross section is prohibited.

3.2 Diffusor

To connect components with different cross sections at the ports, an element has to be used, which allows change in cross section. Such an element is represented by the `diffusor`. The values of cross sections at the left and right connection port are determined by the neighbouring elements (e.g. a `flow element`).

We can obtain the equations describing the `diffusor` element by considering Fig. 3.

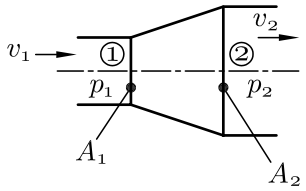

Fig. 3 General situation at a `diffusor`: Volume flow q is constant, the cross section changes from A_1 to A_2, pressure and velocity also change from p_1 to p_2 and v_1 to v_2, respectively. 1 and 2: Points where Bernoulli's principle is applied.

Fluid of volume flow q is streaming in with the velocity v_1 and pressure p_1. As the cross section continuously changes from A_1 to A_2, the values of pressure and flow velocity also change from p_1 to p_2 and v_1 to v_2, respectively. Since this element is not able to store fluid, the volume flow has to remain the same.

We apply once again Bernoulli's principle (see Eq. (1)) and assume $h_1 = h_2$. Since the volume flow q through the `diffusor` has to be constant, we can express the velocities v_1 and v_2 according to

$$v_1 = \frac{q}{A_1} \qquad \text{and} \qquad v_2 = \frac{q}{A_2} \qquad (6)$$

by using the cross sections A_1 and A_2, and obtain the final equation

$$\Delta p = p_2 - p_1 = \frac{\rho}{2}\left(v_1^2 - v_2^2\right) = \\ = \frac{\rho q^2}{2}\left(\frac{1}{A_1^2} - \frac{1}{A_2^2}\right). \quad (7)$$

This equation shows that the pressure is increasing ($\Delta p > 0$) if the cross section is growing too ($A_2 > A_1$), due to the declension of the velocity v and the conservation of energy according to Bernoulli's principle.

3.3 Reservoir

Another important component in the hydraulic library is the `reservoir`. It can store fluid and has connection ports for in- and outlet. For calculation of the pressure at the ports, two different cases have to be considered. If the fluid level in the tank is lower than the height at which the in-/outlet is placed, then the pressure is zero and the fluid can stream in without back pressure. If, on the other hand, the fluid level is higher, the back pressure corresponds to

$$p = \rho g\left(h - h_i\right) \quad (8)$$

with the fluid density ρ, the earth acceleration g, the fluid level h and the in-/outlet height h_i.

The fluid level itself is resulting from the change in fluid amount and the cross section of the `reservoir`:

$$\dot{h} = \frac{1}{A}\left(q_{\text{in}} - q_{\text{out}}\right). \quad (9)$$

The initial condition h_{init} for the fluid level can be set via a block parameter.

The Simscape hydraulic library contains two different `reservoir` blocks (see Fig. 1), `reservoir2` with two in-/outlet ports and `reservoir4` with additional ports for an inlet at the top and an outlet at the bottom. Both elements provide the values of their fluid level at an additional signal output port.

3.4 Valve

Three types of valve elements have been created for the Simscape hydraulic library. The first element, simply called `valve`, represents a simple hand-operated valve, which is similar to the `flow element`, but has a variable cross section, which is adjusted via a block parameter. If this parameter is zero, the `valve` is completely closed, and the volume flow vanishes, i.e. `v=0`.

The `controlled valve` behaves like the normal valve, with the difference that the cross section is regulated via a signal input port instead of a block parameter.

The third type of valve is a `check valve` which allows flow only in one direction. In direction of flow, this component is also described with the same equations as the `flow element`. A negative value of pressure drop across the element would lead to a volume flow in the opposite direction, which, in this case, is prevented by usage of the equation `v=0`.

3.5 Set Area

Like in section 2 described, we use the cross section as a further `across` variable in the hydraulic domain. One possibility to define this cross section is by using a `flow element`. However, with different elements, one can also use the `set area` element, whose only purpose is to define the value of cross section at both ports. The additional equation `p=0` eliminates any other influence of this element on the pressure.

It is important to note that it is not allowed to directly connect a `set area` element with a `flow element`, since both components provide equations for defining the cross section and the Simscape solver would raise an error because the cross section would be over-determined.

3.6 Drain

Every physical domain needs a reference element in order to indicate a reference point in the physical network, with respect to which all absolute values of `across` variables are determined. For our hydraulic domain, this was realized with the `drain` element, which defines the absolute reference of the pressure variable as ideal outflow without back pressure (`p=0`).

3.7 Hydraulic Sources

In addition to classical elements of a physical domain, which define dependencies between the domain's `through` and `across` variables, modelling of physical systems also needs sources of energy, where specific values of these variables are set. For the Simscape hydraulic library, we implemented two types of hydraulic sources, an ideal flow source and an ideal pressure source. The ideal `flow source` provides a desired volume flow independently from the pressure drop. The ideal `pressure source` on the other hand provides a desired pressure drop across its ports independently from the volume flow through the element.

To stipulate the desired values of pressure or volume flow, one can also use the controlled sources available in the Simscape hydraulic library. Unlike the basic sources, where the values of pressure and volume flow are adjusted via a block parameter, the controlled sources use additional signal input ports, from where they receive information about the desired values.

3.8 Sensors

To measure the domain variables pressure and volume flow, hydraulic sensors were created. The `flow sensor` is an ideal gauge measuring the volume flow through its ports and therefore has to be connected in series to an element (see also section 2). Since this element is considered ideal, it does not produce a pressure drop across the ports, which would affect the measurement result. The actual value is provided at a signal output port for further processing.

The `pressure sensor` on the other hand is connected in parallel to an element and measures the pressure difference between its ports without needing any

volume flow (ideal element). It also provides the actual pressure value at a signal output port.

Furthermore, the Simscape hydraulic library contains a `measuring flow element` which behaves like a normal `flow element`, but also measures pressure and volume flow.

4 Verification and Results

To verify the implemented components, an example of a hydraulic system, which is shown in Fig. 4 (cf. [8]), was modeled and simulated with Simscape using the components of the created hydraulic library. It is also tested, if the created Simscape blocks are compatible with existing Simscape and Simulink elements.

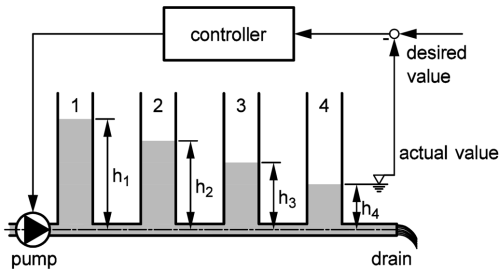

Fig. 4 Example for use of the Simscape hydraulic library

The figure shows four identical connected tanks which can store fluid. A pump brings fluid in the first tank on the left. From there, the fluid moves through pipes between the tanks and drains away after the last tank. The assignment is to simulate a control loop where the fluid level of the last tank is to be controlled by actuating the pump in front of the first tank. Therefore, the fluid level in each tank is measured, the value of the last tank is compared to the desired value and supplied to a controller, which actuates the pump.

In the Simscape model shown in Fig. 5, the pump is represented by a `controlled flow source`, and for the tanks and pipes we used the elements `reservoir2` and `flow element`, respectively.

In order to compare the outcomes of our simulation with known results presented in [8], we used the same parameters for cross sections of the pipes and tanks:

$$\text{pipe 1: } 13.44\,\text{mm}^2, \quad \text{pipe 2: } 18.54\,\text{mm}^2,$$
$$\text{pipe 3: } 17.5\,\text{mm}^2, \quad \text{pipe 4: } 18.75\,\text{mm}^2,$$
$$\text{pipe 5: } 13.44\,\text{mm}^2, \quad \text{tanks: } 27.8\,\text{cm}^2.$$

For the controller, we used a typical PID structure, described by

$$u(t) = K\left(e + \frac{1}{T_N}\int_0^t e\,d\tau + T_V\,\dot{e}\right) \qquad (10)$$

with the control deviation e (i.e. desired value subtracted by the actual value) and the control parameters T_N, T_V and K. These parameters were also chosen like in [8] as follows:

$$K = 5,\ T_V = 12.7\,\text{s},\ T_N = 127\,\text{s}.$$

The resulting implementation of the PID controller in Simscape can be seen in Fig. 6. The controller consists of three parallel branches representing the three parts of the PID controller. Unlike the integrator block, there does not exist a derivative block for Simscape signals, which is why we had to use the existing Simulink block in connection with two converter blocks, which convert the physical signals from Simscape to Simulink and vice versa. A saturation element specifies a range for the ouput (i.e. the actuating variable for the pump) between 0 and 10.

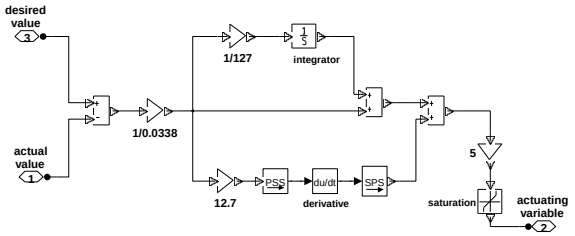

Fig. 6 Structure of the PID controller as subsystem in Fig. 5

As a simulation result of the given task, the progression of all fluid levels in the tanks during the simulation time can be seen in Fig. 7. Thereby, a desired fluid level in the fourth tank of 0.05 m during the first 500 seconds, then 0.08 m until 1000 seconds and at last an alteration to 0.03 m was stipulated.

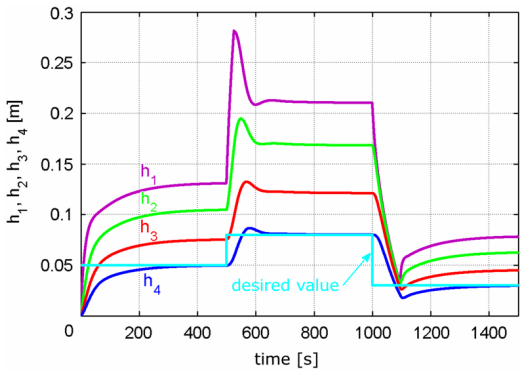

Fig. 7 Progression of the fluid levels and desired value during simulation. The fluid levels h_1 to h_4 correspond to the values labeled in Fig. 4.

We observe conformity with the results given in [8]. To verify continuity of the volume flow in our simulation, we also observed the total amount of fluid volume provided by the pump and compared it to the fluid volume, which has drained away, plus the fluid amounts in the tanks. Since there are no other elements in the model that can store fluid, these two values have to be equal in order to exclude mistakes in calculation of the volume flow.

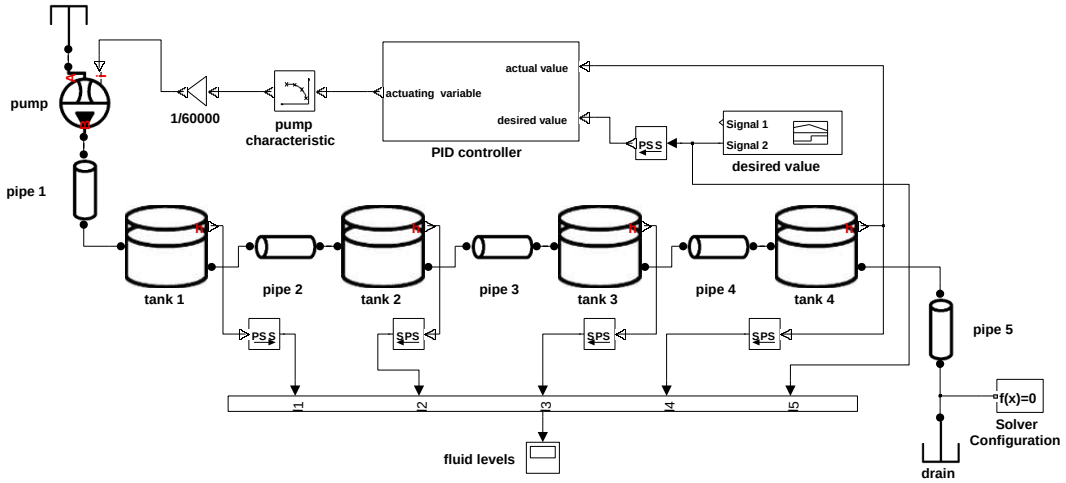

Fig. 5 Simscape model of the example shown in Fig. 4 with blocks from Simulink and the created Simscape hydraulic library. The PID controller subsystem is shown in Fig. 6.

5 Conclusion

Simscape turned out to be a suitable tool for creating hydraulic components and simulating hydraulic systems. We also found that the new components could be used in connection with existing Simscape and Simulink blocks without any problems. However, there were a few unexpected complications with the Simscape solver during the implementation, which had to be solved in order to create an applicable library. These problems appeared for one thing due to the non-linearity of the equations describing the `flow element` and similar components. Another important matter was defining the cross section as further `across` variable in the hydraulic domain in order to avoid calculation errors when connecting blocks with different cross sections.

The concept of acausal modelling with physical networks works very well for mechanical and electrical systems and has proven its worth in these domains by being employed in various modelling languages, such as Modelica [9] or 20-sim [10]. However, these domains use linear balance equations, whereas the balance equations shown in this paper are non-linear. This demonstrates not only a fundamental distinction between these domains, it can also cause difficulties when solving the equations for the model.

The physical network approach simplifies the process of modelling physical systems with Simscape, since the modeler can focus on the physical structure rather than the mathematical equations. However, this approach makes higher demands on the simulation software and especially the algorithms for network construction and automatic conversion of acausal defined DAEs in calculable state space form.

While it requires a lot of effort to create such a Simscape library, the components can be used afterwards very easily and without any restrictions.

6 References

[1] Mathworks. *Simscape 3 - Getting started Guide*. The Mathworks Inc., March 2010. `http://www.mathworks.com/access/helpdesk/help/pdf_doc/physmod/simscape/simscape_gs.pdf`.

[2] Mathworks. *Simscape 3 - User's Guide*. The Mathworks Inc., March 2010. `http://www.mathworks.com/access/helpdesk/help/pdf_doc/physmod/simscape/simscape_ug.pdf`.

[3] F. E. Cellier. *Continuous System Modeling*. Springer-Verlag, New York, 1991.

[4] Mathworks. *Simscape 3 - Reference*. The Mathworks Inc., March 2010. `http://www.mathworks.com/access/helpdesk/help/pdf_doc/physmod/simscape/simscape_ref.pdf`.

[5] Mathworks. *Simscape 3 - Language Guide*. The Mathworks Inc., March 2010. `http://www.mathworks.com/access/helpdesk/help/pdf_doc/physmod/simscape/simscape_lang.pdf`.

[6] G. Zauner B. Zupančič and F. Breitenecker. Modelling and simulation in process technology with modelica. In *Proceedings of the 2005 Summer Computer Simulation Conference*, San Diego, USA, July 24 - July 28 2005. SCS Society for Computer Simulation.

[7] D. Surek and S. Stempin. *Angewandte Strömungsmechanik für Praxis und Studium, p. 51.ff.* B.G. Teubner Verlag, Wiesbaden, 2007.

[8] H. E. Scherf. *Modellbildung und Simulation dynamischer Systeme. Eine Sammlung von Simulink-Beispielen, p. 65.ff.* Oldenbourg Wissenschaftsverlag, Munich, 2007.

[9] S. E. Mattsson H. Elmqvist and M. Otter. Modelica - the new object-oriented modeling language. In *The 12th European Simulation Multiconference*, Manchester, UK, 1998.

[10] Ir. C. Kleijn. *Getting Started with 20-sim 4.0*. Controllab Products B.V., the Netherlands, 2008. ISBN 978-90-79499-01-4.