Using Modelica as a Design Tool for an Ejector Test Bench

Christoph Richter, Christian Tischendorf, Ricardo Fiorenzano, Peterson Cavalcante,
Wilhelm Tegethoff, Jürgen Köhler
Technical University Braunschweig, Institute for Thermodynamics
Hans-Sommer-Straße 5, 38106 Braunschweig, Germany
ch.richter@tu-bs.de

Abstract

As a result of the Kyoto Protocol, the use of R134a in air conditioning system of new cars is going to be forbidden in the EU due to the high Global Warming Potential (GWP) of this substance. Carbon dioxide (CO_2) is one of the possible alternatives as a cooling agent in mobile air conditioning applications and is desirable since it is a natural refrigerant. Cooling cycles using CO_2 are currently achieving COP similar to those of R134a cycles but there are promising options to further improve COP. One possibility is the application of an ejector instead of the valve. A Modelica library was developed that allows computation of ejector cooling cycles in steady state with simplified component models that can be used as a design tool for the construction of an ejector prototype. The library uses a new object-oriented library that serves as an interface to external medium libraries to compute thermodynamic and transport properties for the refrigerant.

Keywords: Fluid properties, cooling cycle, ejector

1 Introduction

As a result of the discussion about reducing the worldwide emission of greenhouse gases, carbon dioxide (CO_2) was re-discovered as a natural refrigerant with promising thermodynamic properties. Among a few others, Lorentzen [1] pointed out early that trans-critical CO_2 refrigeration cycles encounter significant throttling losses reducing their coefficient of performance (COP). One way to overcome this problem is to use an ejector instead of the throttling valve. A lot of research is currently carried out in this field [2], [3] and the obtained results are very promising [4].

This paper describes the development of a Modelica library that was used in the design process of a test bench for an ejector cooling cycle and that will be extended to allow the deeper analysis of the ejector refrigeration process. The developed library is kept as simple as possible. The fluid properties for CO_2 are computed using a new fluid property library that provides a convenient interface to an external library written in C.

2 Problem description

To be able to simulate an ejector cooling cycle one needs models for the components of the cooling cycle as well as for the fluid properties of the chosen refrigerant. Although it would be possible to have one library for both requirements it usually is a good idea to have separate libraries for the components and for fluid properties. Well-known Modelica libraries such as Modelica.Media/Modelica_Fluid and AirConditioning Library/ThermoFluidPro follow the same basic concept.

Before looking at libraries that are capable of modeling the given problem the most important requirements are compiled in the following list:

- The component models should be as simple as possible for the first rough analysis during the design process. The component models should only require the definition of a minimal set of parameters and efficiencies because exact geometries are usually not known in the first rough analysis.
- There are a lot of engineers that can only spend a certain amount of their work time on understanding Modelica libraries. The developed library should enable those users as well as students with no or little background in modeling to get started with developing new models as easily as possible.
- The simple models used in the first steps of the design process should be designed in such way, that they can be replaced by more detailed models in future steps when an in-depth analysis of experimental results is required.

- The fluid property library should be as flexible as possible. It should be possible to use the same fluid properties in different applications (i.e. Modelica, Matlab, LabVIEW, CFX) and the use of the library should be as simple and straightforward as possible.
- The fluid property library should not contain any compiler specific elements and should be as flexible to use as possible.

The following subsections give a brief overview over existing libraries and point out their advantages and drawbacks within the scope of simulating a CO_2 ejector cooling cycle.

2.1 Modelica_Fluid/Modelica.Media

The Modelica_Fluid library is a free library for describing 1-dimensional thermo-fluid flow and is developed by the Modelica Association. The library is still under development and a new beta version is going to be presented at this conference [5]. The Modelica_Fluid library provides models for standard fluid components such as pipes, control valves and pumps. There are currently no models for compressors or simple heat exchangers available. Modelica_Fluid uses fluid models from Modelica.Media which is part of the Modelica Standard Library since version 2.2. Modelica.Media provides medium models for ideal gases and water. Refrigerants are currently not implemented within Modelica.Media. Modelica.Media can be extended to allow the use of external fluid property libraries implemented in FORTRAN or C.

2.2 AirConditioning Library/ThermoFluidPro

The AirConditioning Library is a commercial library developed and maintained by Modelon AB [6]. This library allows transient and steady-state simulations of air conditioning systems at a very high model complexity level. It is used as a development tool by several large automotive companies. The AirConditioning Library uses ThermoFluidPro as fluid property library that is also developed and maintained by Modelon AB. ThermoFluidPro offers high-precision equations for the fluid properties of CO_2 using the Helmholtz equation given by Span and Wagner [7].

2.3 TIL/TILFluids

TIL (TLK-IfT-Library) is a library for simulating refrigeration systems developed and maintained by the Institute for Thermodynamics (IfT) at TU Braunschweig in close cooperation with TLK (TLK-Thermo GmbH). TIL contains models with very different levels of complexity. TIL.HVAC contains very simple models that are used for educational and training purposes and that are especially designed to allow an easy access to this library for new users. More detailed models that are used in R&D projects are also available within TIL. TIL uses TILFluids to compute fluid properties. TILFluids does not provide fluid models implemented in Modelica but offers general interfaces to employ existing external libraries written in C and FORTRAN.

2.4 Summary and Conclusion

All three libraries could be extended to allow the computation of CO_2 ejector refrigeration cycles. Fluid properties for CO_2 are readily available in the ThermoFluidPro library and in TILFluids.

The main drawback of Modelica_Fluid was its development status when starting this project. Another drawback is the lack of simple standard components (i.e. compressors, simple gas coolers) that are needed to build up a cooling cycle. Many of the models are already too complex for the given application. Modelica_Fluid is furthermore taking into account advanced flow situations such flow reversal which are not required within the scope of this work.

The models from the AirConditioning Library are very detailed due to the intended field of application for that library. They seem to be too detailed for a rough analysis during the design process of a test bench. ThermoFluidPro is a very advanced and powerful fluid property library completely implemented in Modelica. It is an extension of the free Modelica.Media library and offers a variety of different models including models for pure refrigerants, binary mixtures, pseudo-pure fluids, and water.

TIL offers very simple as well as advanced models for components used in air conditioning systems. The simple models in TIL.HVAC are especially suitable for projects that involve students having limited or no previous experiences with Modelica.

It was therefore decided to use TIL and to implement a simple ejector model within the framework TIL.HVAC, the part of TIL that is used for educational purposes. The fluid property library TILFluids was considerably improved during this project and offers a very user-friendly object-oriented approach for fluid properties.

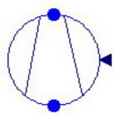
Developing libraries that are easy to understand and simple in their basic concept seems to be a very important task. A lot of the available libraries seem to be unnecessary complicated which makes it hard for new users to get started with Modelica. We experienced this problem with students that were asked to work with Modelica.Media and Modelica_Fluid. They had quite some difficulties to get started with those libraries. The same thing seems to happen with commercial partners where an engineer is asked to start modeling using one of these libraries. It is very common that engineers can only dedicate a small percentage of the work time to this task which makes it very hard to get started.

Concerning the Fluid-Properties it seems to be important that companies can reuse their code that they are already employing in other problems. These are usually FORTRAN or C libraries that they might have developed over decades. Now start looking at Modelica.Media to figure out how to implement something like an external interface for a two-phase medium. You would probably start with inheriting from PartialTwoPhaseMedium in which case one is focusing an inheritance structure that is 4 levels deep. This is not really human-readable and far from being easy to understand. One would also probably have a closer look at the implemented water model implementing the IF97 standard. The problem with this model is that it actually incorporates late inlining and a hidden property record to improve performance significantly when using Dymola. The structure of this two-phase library is not easy to understand for new users.

3 TIL.HVAC

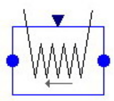
TIL.HVAC is the part of TIL that contains simple models for each component of a refrigeration cycle. The following section gives a brief overview over the most important components.

3.1 Compressor



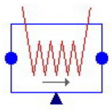
The simple compressor model uses constant volumetric, isentropic and energetic efficiencies to describe the change of state, the mass-flow rate and the energy consumption. It also allows setting the rotational speed using a standard RealInput.

3.2 Gas cooler



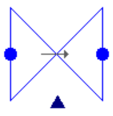
The gas cooler model assumes that the pressure drop within the gas cooler is negligible. It uses a static mass balance and a static momentum balance and allows the user to set an adiabatic efficiency. This model takes the ambient temperature as an input.

3.3 Evaporator



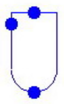
The evaporator model is quite similar to the gas cooler model. It also assumes negligible pressure drop and static mass and momentum balances.

3.4 Valve



The valve model assumes an isenthalpic throttling process. The effective flow area can be specified using the RealInput interface.

3.5 Separator

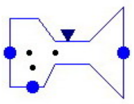


The separator model assumes that the separation process works ideal yielding saturated liquid at the liquid outlet and saturated vapor at the vapor outlet. The pressure drop is neglected which might not be realistic.

3.6 phDiagram

The phDiagram is a sensor that can be used to record pressure and specific enthalpy anywhere in the cycle. It allows for simple post processing using phMonitor, a program developed at the IfT in cooperation with TLK, to visualize the results in a ph-diagram.

3.7 Ejector



The ejector model is based on a very simple model initially developed by Kornhauser [8]. It is composed of two nozzles, a mixing section and a diffuser. The main purpose of the ejector is to recover some of the kinetic energy otherwise converted into friction and lost during the throttling process. The refrigerant is entering the primary (motive) nozzle at high pressure and is accelerated. The high velocity stream leaving the primary nozzle is used to entrain refrigerant from the secondary (suction) nozzle. Both streams are mixed in the mixing section. The diffuser is used to convert parts of the remaining kinetic energy of the mixed stream into a pressure increase that helps to improve the overall performance of the cycle.

The following assumptions were made for the ejector model. The analysis is one-dimensional and the refrigerant is in thermodynamic-equilibrium at all times. Those first two assumptions correspond to what is called homogeneous equilibrium model in two-phase flow. It is furthermore assumed that the deviations from adiabatic reversible processes that occur in the nozzles and the diffuser can be expressed in terms of efficiencies. Any shock effects that might occur are included in these efficiencies. The kinetic energy was considered to be significant only within the ejector not within the rest of the cycle.

Some of these assumptions will have to be reconsidered for a more detailed analysis. One of the main problems of the implemented model is the missing relationship between mass flow rate and pressure drop for the nozzles which requires the explicit specification of at least one pressure level (i.e. the mixing pressure). Other models such as the one developed by Groll [2] suffer from the same problems. Experimental analysis of the ejector refrigeration cycle which is currently performed at the IfT might overcome this drawback.

Figure 2 shows the internal structure of the developed ejector model.

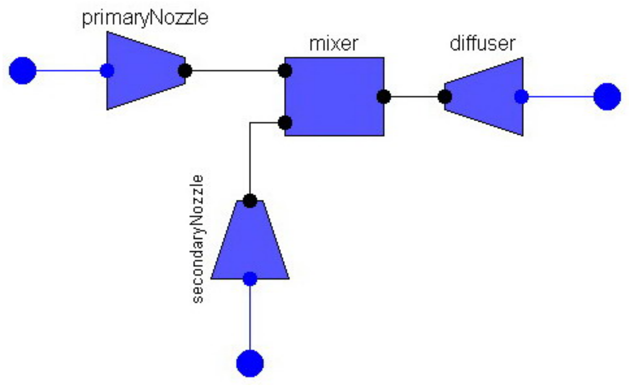


Figure 2: Internal structure of the ejector model.

4 TILFluids

TILFluids is an object-oriented fluid property library. The main design goal when developing this library was to create a simple interface that can easily be extended by users to fit their individual needs. Figure 1 shows the general structure of the library. The package Common contains common elements such as types and records that are used throughout the library. The package Icons contains the library icon. The package Internal is the core of the library and contains all functions that are needed to access the external library. The contained functions are not used directly in the component model but rather accessed using a fluid object such as Gas, Liquid or Refrigerant.

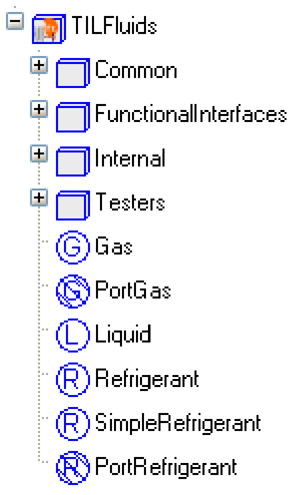
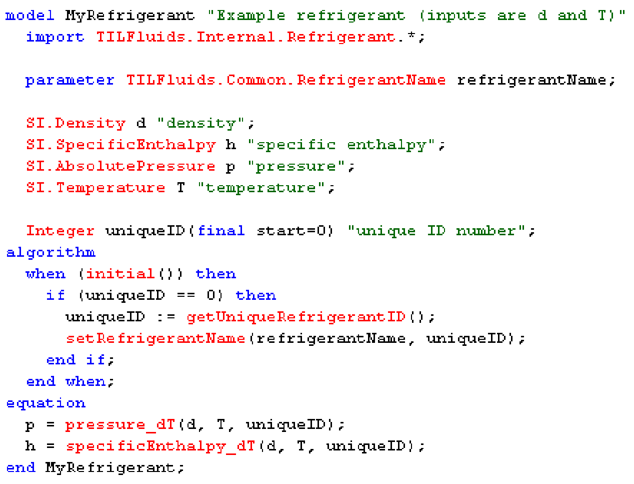


Figure 1: Structure of TILFluids

The provided fluid objects such as “Refrigerant” and “Gas” are ready-to-use implementations of medium models that can be used in a wide range of applications. They can also be changed or extended by developers to best meet their own requirements.

Figure 3 shows an example for a simple refrigerant model. The unqualified import statement ensures that the internal functions required for computing the fluid properties (i.e. “pressure_dT()”) are available within the model “MyRefrigerant”. The “refrigerantName” is a character string with a couple of predefined choices specified in TILFluids.Common. The “uniqueID” is an integer that uniquely identifies each instance of “MyRefrigerant”. All functions in the external library require the “uniqueID” as an input. The external library creates a map of refrigerants and stores information about the refrigerant and the results from the last computation within this map. This allow for an optimization regarding the computational time requirements. The library can be designed in such a way that it returns the values computed at the last call for each refrigerant if the inputs did not change. This is very important especially in the case when an inverse iteration is required for the computation.



```
model MyRefrigerant "Example refrigerant (inputs are d and T)"
  import TILFluids.Internal.Refrigerant.*;

  parameter TILFluids.Common.RefrigerantName refrigerantName;

  SI.Density d "density";
  SI.SpecificEnthalpy h "specific enthalpy";
  SI.AbsolutePressure p "pressure";
  SI.Temperature T "temperature";

  Integer uniqueID(final start=0) "unique ID number";
algorithm
  when (initial()) then
    if (uniqueID == 0) then
      uniqueID := getUniqueRefrigerantID();
      setRefrigerantName(refrigerantName, uniqueID);
    end if;
  end when;
equation
  p = pressure_dT(d, T, uniqueID);
  h = specificEnthalpy_dT(d, T, uniqueID);
end MyRefrigerant;
```

Figure 3: Code example for a simple refrigerant model that uses function calls to an external library to compute fluid properties

The above example is instantiated in the component model and contains all required fluid properties. Actual refrigerant models will contain more thermodynamic and transport properties than the simple “MyRefrigerant” from example above. Figure 4: Code example for a simple component model to demonstrate the use of the fluid object.Figure 4 shows a simple component model with an instantiated refrigerant model. Density and temperature are specified for “myRefrigerant” and the pressure and specific enthalpy are computed within “MyRefrigerant” and are attributes of “myRefrigerant”. The object-oriented approach used in TILFluids was found to be very user-friendly and allows for a clear separation of component and fluid model. The TILFluids Users’ Guide [11] offers more detailed information on using and extending TILFluids.

There are situations where a functional approach might be handier than the object-oriented approach used in TILFluids. TILFluids therefore provides functional interfaces that can be called without instantiating a medium. The functional interfaces are contained in TILFluids.FunctionalInterfaces. However the user should be aware of certain limitations that apply to the functional interfaces (see [11] for more details).

```
model MyComponent "Example component"
  MyRefrigerant myRefrigerant(refrigerantName="CO2")
    "refrigerant object";

  SI.Density d "density";
  SI.Temperature T "temperature";

  SI.AbsolutePressure p "pressure";
equation
  myRefrigerant.d = d;
  myRefrigerant.T = T;

  d = 10 + 990*time;
  T = 273.15;

  p = myRefrigerant.p;
end MyComponent;
```

Figure 4: Code example for a simple component model to demonstrate the use of the fluid object.

The current external library that is used with TILFluids provides interfaces to two different libraries. The first interface uses the fluid properties used in a platform for steady-state computation of cooling cycles that was developed at IfT in the past [9]. The second interface calls the current beta version of Refprop (for details see [10]) to obtain fluid properties. An additional interface to a library that implements the IF97 standard for water is currently under development. The TILFluids users’ manual offers more detailed information on how to use TILFluids [11].

The interface to a C library of fluid property data developed at IfT is available for free, the Refprop interface upon request. Additional interfaces can be implemented if required.

One of the advantages of having medium models implemented in Modelica is that the simulation software can take full advantage of the entire equation. It can use this information to compute analytical Jacobians or to perform index reduction. Using an external library might require the explicit definition of derivative functions using annotations. This is currently not implemented in TILFluids but can be added at a later time. It is however very often possible and desirable to formulate the problem in such a way that no index reduction is required to solve it. The Bridgman tables [12] can be used in many thermodynamic problems to accomplish this task.

5 Example

The main goal when developing the presented libraries was to perform preliminary computations for an ejector cooling cycle. A comparison of a conventional cooling cycle and an ejector refrigeration cooling cycle was carried out in a first step to demonstrate the theoretically achievable COP improvements. In a second step a certain design point was chosen for the test bench and all required geometrical parameters were computed (see [13] for more details).

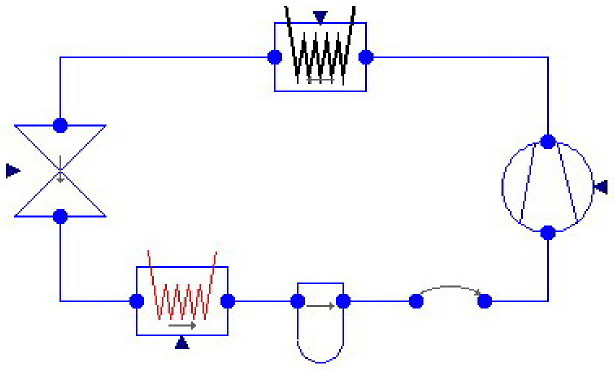


Figure 5: Schematic diagram of a standard cooling cycle (the extra component behind the receiver breaks the algebraic loop for the mass flow rate)

An ejector refrigeration cycle is compared to a standard trans-critical CO_2 cooling cycle to determine the possible gain in COP for the ejector cycle. The COP is defined as

$$COP = \frac{Q_{evap}}{W_{p,el}}$$

where Q_{evap} is the evaporator refrigeration capacity and $W_{p,\,el}$ is the electrical power consumption of the compressor. Figure 5 shows the standard cooling cycle and Figure 6 the ejector cooling cycle.

The same conditions were applied to both cycles. The simple compressor model assumes a volumetric efficiency of 70% and an isentropic efficiency of 70%. The displacement was set to 30 cm^3 and the speed to 20 Hz. The evaporation temperature was set to 0°C and the gas cooler was assumed to work with 100% efficiency, this means that the CO_2 outlet can be cooled to ambient temperature.

For the one stage CO_2 cycle, the throttling valve was assumed to operate isenthalpic. The opening area of the valve was varied to achieve maximal COP.

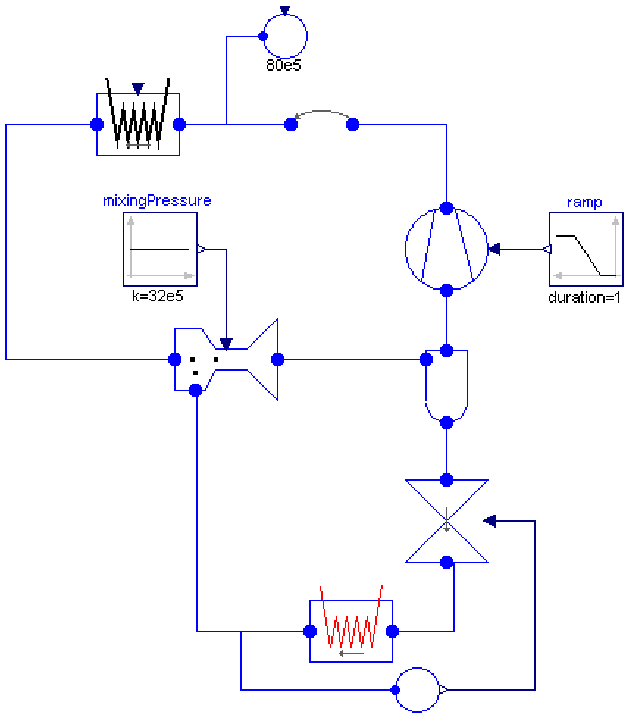


Figure 6: Schematic diagram of ejector cooling cycle

In the ejector cycle, the ejector efficiencies were set to 100% for the first cycle computations to estimate the maximum COP improvements for the ejector cycle compared with a conventional cycle. That means, that all three nozzles are working isentropic and that the mixing occurs without any losses. However, this will not be the case for the real ejector. Experimental results from [2], [3] show, that there are significant losses especially in the suction nozzle and during the mixing process. The mixing pressure for the ejector cycle was set to 32 bar. Computations to determine the effects of the mixing pressure on cycle efficiency and to determine the optimal mixing pressure are currently carried out at the IfT.

COP computations were carried out for ambient temperatures from 26°C to 40°C and compressor outlet pressures from 86 bar to 120 bar. The COP for each ambient temperature that yielded the best COP was chosen for the comparison of the two cycles.

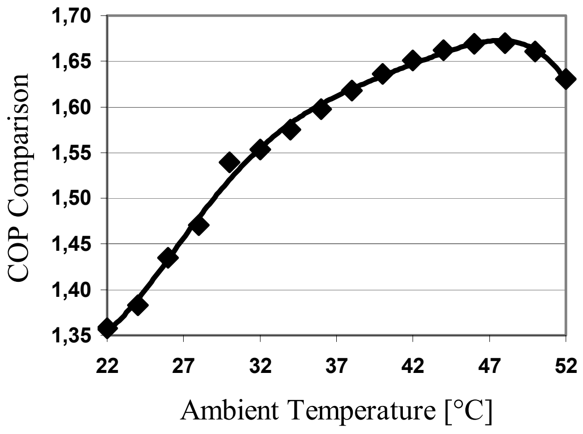


Figure 7: Comparison of COP of the conventional and the ejector cooling cycle.

Figure 7 shows a comparison of the COP of the two cycles. The performance of the ideal ejector cycle is clearly better than the performance of the conventional cycle for all operating conditions. However, one should keep in mind that ideal working conditions were assumed and that Figure 7 does not represent the expected improvement in COP for a real cycle. However, it shows the potential of the ejector cooling cycle.

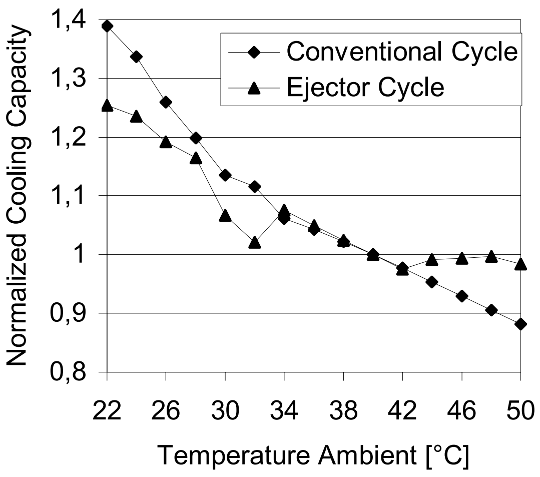


Figure 8: Comparison of normalized cooling capacity of the conventional and the ejector cooling cycle

Figure 8 shows a comparison of the normalized cooling power of a conventional and an ejector cooling cycle for different ambient temperatures.

The cooling power of the conventional refrigeration cycle is declining drastically for high ambient temperatures due to the raising losses in the throttling process. The ejector cooling cycle can regain some of this otherwise dissipated energy.

The analyzed ejector cycle does not use an internal heat exchanger which might actually improve the performance in some regions. An analysis of the effect of using an internal heat exchanger will be investigated in detail in the future.

Using the same analysis it could be shown that the ejector efficiencies have to be better than 70% to achieve a better overall performance than for the conventional cooling cycle.

One design point for a heat pump application was chosen for the test bench and all geometric parameters have been computed for that design point. The gas cooler outlet temperature was assumed to be 20 – 25°C with a gas cooler outlet pressure of 95 bar. The gas cooler power was set to 4 – 5 kW and the evaporator pressure to 37 – 40 bar. A driving mass flow rate of approximately 0.02 kg/s is required to obtain the designated heating power. The ejector geometry was determined using the results from the cycle analysis for this operating point.

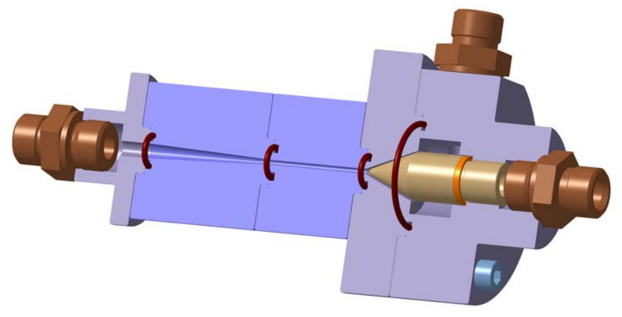


Figure 9: Cross-sectional view of the designed ejector

Figure 9 shows the designed ejector in a cross-sectional view. The prototype is currently tested at IfT. The experimental results will be used to further improve the ejector model.

6 Conclusion and future perspective

An existing simple library with standard elements to compute a cooling cycle has been extended to allow the steady-state computation of ejector cooling cycles. A new fluid property library for Modelica has been developed that is easy to use and that offers the possibility to use existing external libraries to compute fluid properties. The results from the first computations were used to set up a test bench for ejector cooling cycles at the IfT.

Transient models for the cooling cycle components and for the ejector are currently developed to allow a detailed analysis of the cooling cycles. The library is also extended to allow an exergy analysis based on the second law of thermodynamics. The results from the designed test bench will be used to further improve the existing models.

References

[1] Lorentzen G., 1983, Throttling – the Internal Haemorrhage of the Refrigeration Process, Proc. Inst. Refrig., Vol. 80:39-47.

[2] Li D. and Groll E.A., 2006, Analysis of an Ejector Expansion Device in a transcritical CO_2 Air Conditioning System, Proc. 7th IIR Gustav Lorentzen Conference on Natural Working Fluids, Trondheim, Norway.

[3] Elbel S. and Hrnjak P., 2006, Experimental Validation and Design Study of a Transcritical CO_2 Prototype Ejector System, Proc. 7th IIR Gustav Lorentzen Conference on Natural Working Fluids, Trondheim, Norway.

[4] Ozaki Y., Takeuchi H. and Hirata, 2004, Regeneration of Expansion Energy by Ejector in CO_2 Cycle, Proc. 6th IIR Gustav Lorentzen Conference on Natural Working Fluids, Glasgow, UK.

[5] Casella F., Otter M., Proelß K., Richter C., Tummescheit H., 2006, The Modelica Fluid and Media library for modeling of incompressible and compressible thermo-fluid pipe networks, Proc. 5th International Modelica Conference, Vienna, Austria.

[6] Modelon AB, http://www.modelon.se, Sweden

[7] Span, R., Wagner, W. 1996, A New Equation of State for Carbon Dioxide Covering the Fluid Region from the Triple-Point Temperature to 1100 K at Pressures up to 800 MPa, Journal of Physical and Chemical Reference Data, 25/6: 1509-1596.

[8] Kornhauser A., 1990, The Use of an Ejector as an Refrigerant Expander, USNC/IIR Purdue Refrigeration Conference, Purdue, Indiana

[9] Tegethoff W., 1999, Eine objektorientierte Simulationsplattform für Kälte-, Klima- und Wärmepumpensysteme, Diss. TU Braunschweig, Germany

[10] Lemmon E.W., McLinden M.O., Huber M.L., 2002, NIST Reference Fluid Thermodynamic and Transport Properties – REFPROP, Version 7.0, Users' Guide

[11] Richter C.C., Cavalcante P., Tegethoff W., 2006, TILFluids – Users' Manual, Version 0.9

[12] Bejan A., 1988, Advanced Engineering Thermodynamics, Wiley-Interscience

[13] Tischendorf, C., 2006, Aufbau eines Prüfstandes für Ejektoren und Ejektor-Kältekreisläufe, TU Braunschweig, Diplomarbeit