Dynamic Modeling and Simulation of Digital Displacement Machine

Sanjib Chakraborty
Fluid and Mechatronic Systems

Master Thesis
Department of Management and Engineering
Division of Fluid and Mechatronic Systems
Linköping University, Sweden
LIU-IEI-TEK-A--12/01489—SE
Master Thesis

Linköping, August, 2012

Department of Management and Engineering
Division of Fluid and Mechatronic Systems

Examiner: Professor Karl-Erik Rydberg
Supervisor: Robert Braun

Top Page Picture:

Yigen C., Control of a Digital Displacement Pump, Department of Energy Technology, Aalborg University, 2012.
Abstract
Improved efficiency, better controllability and low noise are the most demanding features form a displacement machine now-a-days. Most of the conventional displacement machines are basically a reciprocating pumping element controlled by valve plates or with the help of check valve [1]. This kind of hydraulic machines lose efficiency dramatically at partial displacement because all of the pistons remain at high pressure at the cycle time and due to pressure inside the piston leakage and shear losses increases. One approach to improve the efficiency of the displacement machine can be controlling each hydraulic piston by using programmable faster valves called digital valve. As the total displacement will be controlled digitally, the total system is called Digital Displacement Technology.
In digital displacement machine it is possible to disconnect some of the pistons from the load and the piston will connect only with the low pressure side, minimizing losses due to leakage and shear. As the valve will control directly with digital controller it will eliminate the necessity of servo-hydraulic control required by conventional systems. Digital valves can open fully and close again with the input signal within one revaluation of the shaft, so it gives better control to the pumping element results reduction in hysteresis and increase the linearity of the pumping element. In Digital Displacement machines by controlling the valves pistons are connected with the machine when pressure is equal, but in the traditional machines piston connection was pre-determined with the shaft angle. By doing the piston control efficiency of the machine will improve and the sound generates for the decompression flow will be reduced [17]. Also energy storage and recovery can be possible by using accumulator.
Preface

This thesis work has been written at the Division of Fluid and Mechatronic Systems (FluMeS), part of the Department of Management and Engineering (IEI) at Linköping University (LiU).

I would like to convey my gratefulness to LiU for making this project achievable and for giving me the opportunity to complete my studies in an interesting and rewarding way. During this project I have acquired experiences and greatly increased my knowledge in my field of interests.

I would like to thank all the staff at IEI who has helped me out through the project, especially my supervisor Robert Braun for his continuous help and my examiner Professor Karl-Erik Rydberg.

August 2012
Linköping, Sweden
Sanjib Chakraborty
**Contents**

Abstract................................................................................................................................. i
Preface ....................................................................................................................................... ii
List of Figures .......................................................................................................................... v
List of Tables ............................................................................................................................. vi
Nomenclature .......................................................................................................................... vii

1. Introduction ................................................................................................................................. 1
   1.1 Objective ................................................................................................................................. 1
   1.2 Purpose .................................................................................................................................. 1
   1.3 Limitation ............................................................................................................................... 1

2. Background Study ...................................................................................................................... 2
   2.1 Background of Digital Displacement ..................................................................................... 2
   2.2 Background of Hopsan .......................................................................................................... 10

3. System Sketch ............................................................................................................................ 15

4. Methodology .............................................................................................................................. 17
   4.1 Overview ............................................................................................................................... 17
   4.2 Research questions ............................................................................................................... 17
   4.3 Research design .................................................................................................................... 17
   4.4 Components ........................................................................................................................ 17
   4.5 Cam ..................................................................................................................................... 18
       4.5.1 Definition and Classification .......................................................................................... 18
       4.5.2 Cam nomenclature ....................................................................................................... 19
       4.5.3 Cam Curves .................................................................................................................. 20
       4.5.4 Constant velocity curve ............................................................................................... 20
       4.5.5 Simple harmonic motion curve .................................................................................. 21
       4.5.6 Double harmonic motion curve .................................................................................. 22
       4.5.7 Force analysis ............................................................................................................... 23
       4.5.8 Force Calculations ......................................................................................................... 23
       4.5.9 Parallel force, P .............................................................................................................. 24
       4.5.10 Normal force, P_n ......................................................................................................... 24
       4.5.11 Inertia effect ................................................................................................................ 25
       4.5.12 Torque For simple harmonic motion ......................................................................... 26
       4.5.13 Spring effect ............................................................................................................... 26
       4.5.14 Friction effect .............................................................................................................. 28

4.6 Hydraulic Cylinder ................................................................................................................. 29

4.7 Digital Valve ........................................................................................................................... 29
List of Figures

Figure 1: 6 cylinder Digital Displacement pump/motor by Artemis IP limited ........................................ 2
Figure 2: Pumping and idling mode of Digital Displacement pump/motor by Artemis IP limited ........ 2
Figure 3: Motoring mode of a Digital Displacement machine ................................................................. 3
Figure 4: Valve operation for one Motor cycle of a Digital Displacement machine ................................. 3
Figure 5: Layout of three piston digital pump motor .................................................................................. 4
Figure 6: Working cycle of a Single Piston of digital pump motor acting as a pump and motor ............ 5
Figure 7: Layout of three piston digital pump-motor with two independent outlet digital pump motor 5
Figure 8: Efficient hydraulic system ....................................................................................................... 6
Figure 9: Comparison between high efficient system and load sensing system ...................................... 6
Figure 10: Possible Control Sequence of three-piston pump-motor ......................................................... 8
Figure 11: Flow rates with different control sequence and different number of piston ............................. 8
Figure 12: Two ways on/off valves and assembly of valves .................................................................... 9
Figure 13: Systems with Digital Displacement Machine and different valve system .......................... 9
Figure 14: Graphical User Interface of Hopsan ....................................................................................... 10
Figure 15: Block Diagram of lossless Transmission Line ........................................................................ 11
Figure 16: C-type and Q-type component in Hopsan ............................................................................. 12
Figure 17: Reduction of simulation time by using Multi-core .................................................................. 13
Figure 18: Test model consisting of Load-sensing system ..................................................................... 13
Figure 19: Result of using multi-threaded simulation ............................................................................. 14
Figure 20: Graphical User Interface of creating component by using Modelica code .......................... 14
Figure 21: System Sketch using Standard Symbol for six Piston Digital Displacement Pumps ........ 16
Figure 22: Different types of translating followers .................................................................................. 18
Figure 23: Graphical layout of cam-follower system showing maximum displacement of the follower after 90 degrees of cam rotation ................................................................................. 18
Figure 24: Cam nomenclature ............................................................................................................... 19
Figure 25: Constant velocity curve ........................................................................................................ 21
Figure 26: Simple harmonic motion curve construction .......................................................................... 22
Figure 27: Simple Harmonic curve ........................................................................................................ 22
Figure 28: Double harmonic curve ........................................................................................................ 23
Figure 29: Forces acting on a cam .......................................................................................................... 23
Figure 30: Illustration of inertial force and inertial torque ...................................................................... 26
Figure 31: Illustration of inertial force and spring force ......................................................................... 27
Figure 32: Inertial force and spring force for different curves ................................................................. 27
Figure 33: ISO 1219 symbol of a Hydraulic Cylinder .............................................................................. 29
Figure 34: Normally Closed and Open 2/2 on-off valve ........................................................................ 30
Figure 35: Valve operation time ............................................................................................................. 31
Figure 36: Adding ports and parameters on Hopsan component generator ........................................ 33
Figure 37: Hopsan component generator showing the list of parameters .......................................... 34
Figure 38: Hopsan component generator showing the component equations ..................................... 34
Figure 39: Generated Components in the Hopsan External Library ..................................................... 35
Figure 40: Simple model for testing the Generated Cam component with 3 liner ports .................... 35
Figure 41: Position and velocity from one piston .................................................................................... 36
Figure 42: Angular velocity of the Cam ................................................................................................. 36
Figure 43: Pump model with three piston and simple controller ........................................................... 37
Figure 44: Angular velocity of the model shown in figure 28 ............................................................... 38
Figure 45: Simultaneous opening and closing signal from the controller .............................................. 38
List of Tables

Table 1: Selection of Mode based on Valve Controlling ................................................................. 4
Table 2: Parameter for the Model shown in figure 28 ..................................................................... 37
Table 3: Parameter for the Model shown in figure 34 ..................................................................... 40
Nomenclature

\(a = \) follower acceleration, \(m/sec^2\)
\(c = \) length of cam rod guide, \(m\)
\(d = \) cam rod diameter, \(m\)
\(f = \) friction factor
\(g = \) gravity constant, 9.81 \(m/sec^2\)
\(h = \) total rise, \(m\)
\(L = \) external force, \(N\)
\(m = \) ratio of follower overhang to guide length
\(n = \) any number
\(P = \) force parallel to cam rod, \(N\)
\(P_n = \) force normal to cam profile, \(N\)
\(Q_1, Q_2 = \) forces normal to cam rod, \(N\)
\(r = \) radius to the reference point, \(m\)
\(S = \) spring force, \(N\)
\(T = \) torque, \(N-m\)
\(W = \) weight of the accelerated elements, \(N\)
\(y = \) follower displacement, \(m\)
\(y' = \frac{dy}{d\theta} = \) follower velocity, \(m/rad\)
\(y'' = \frac{d^2y}{d\theta^2} = \) follower acceleration, \(m/rad^2\)
\(y''' = \frac{d^3y}{d\theta^3} = \) follower pulse, \(m/rad^3\)
\(\dot{y} = \frac{dy}{dt} = \omega y' = \) follower velocity, \(m/sec\)
\(\ddot{y} = \frac{d^2y}{dt^2} = \omega^2 y'' = \) follower acceleration, \(m/sec^2\)
\(\dddot{y} = \frac{d^3y}{dt^3} = \omega^3 y''' = \) follower pulse, \(m/sec^3\)
\(\beta = \) cam angle for rise \(h, \) rad
\(\omega = \) cam speed, rad/sec
\(\theta = \) cam angle of rotation, rad
\(\gamma = \) cam pressure angle, rad
\(\mu = \) coefficient of friction
1. Introduction

World is digitalizing day by day and to maintain the trend of digitalization displacement machines need to be changed more toward the digital technology. Digital displacement technology uses high speed computer operated on/off valves as a replacement of traditional port and swash plates for displacement machines. This paper will give an overview of digital displacement technology and also a simulation process of making a generic model of Digital displacement pump by using simulation software Hopsan, which is developed by Department of Fluid and Mechatronic Systems (Flumes), Linköping University, Sweden.

1.1 Objective

The objective of this thesis work are following:

- Background study of digital displacement technology.
- Developing a generic model of a digital displacement pump by using Hopsan.
- Developing simple control strategy for controlling the machine.
- Developing simple controller using Matlab/Simulink for the digital displacement pump.
- Validate the model with some example.

1.2 Purpose

The main purpose of this thesis is to gather detailed knowledge about the Digital Displacement Technology and a background study of so far development in the field of digital displacement technology. Also develop a generic pump model using simulation package Hopsan for investigating the performance of the machine, which can be used for research and further development in the field of fluid power.

1.3 Limitation

Limitations of this thesis work are described below:

- Only two cam profiles were developed in this work, more cam profile can be developed and implemented in the model to check the result.
- Cam component needs to change for each kind of profile and for more pistons to attach with the cam.
- A common controller is not included in the model, only the controller for specific example is included.
- The controller is a simple controller based on the velocity profile of the pistons.
- Pre-compression and de-compression inside the cylinder wasn’t consider for the development of the controller.
- Due to the time limitation it was not possible to make the model work as a motor which needs a robust controller to switch the pressure side of the pistons.
- There are so many hydraulic system can be simulated with the newly developed pump model but only two simple systems were investigated.
2. Background Study

2.1 Background of Digital Displacement

According to Artemis intelligent power limited [3] the main component of a digital displacement machine system are hydraulic piston pump/motor with actively controlled poppet valves to control the flow out and into the each cylinder. An electro-magnetic latch control the opening and closing the valves on a per stroke basis. The solenoid coil in each latch is directly connected to the output of a controller which is activated by a power FET. A 6 cylinder Digital Displacement pump/motor developed by Artemis intelligent power limited shown in figure 1 has radially positioned cylinder with valve around the border of the cylinder. Each cylinder has digitally controlled poppet valve, one is connect to high pressure side and another one is connected to low pressure side [3, 4].

![Figure 1: 6 cylinder Digital Displacement pump/motor by Artemis IP limited [3]](image1)

The pumping and idling mode of this machine is shown in figure 2. At the starting of the pumping mode low pressure valve opens and the piston goes from Top dead centre (TDC) to bottom dead centre (BDC), at that instant of time the high pressure valve remains close. After the piston reach its BDC both the valve remains close, due to that pressure increase inside the cylinder and then the high pressure valve opens, the fluid flow to the high pressure side. In the idling mode only low pressure valve remain open, high pressure valve is close, fluid flows in and out of the low pressure valve and separate the cylinder for the high pressure side which reduces the frictional losses and improve the efficiency.

![Figure 2: Pumping and idling mode of Digital Displacement pump/motor by Artemis IP limited [3]](image2)
By controlling the valve sequence at the end of the stroke it is possible to choose the pumping or motoring mode for each cylinder. To change the mode of the cylinder from pump to motor the response time of the valve should be fast enough. The motoring mode of the digital displacement machine is described below along with figure 3.

At the starting of the motoring mode high pressure valve held open by the valve controller and low pressure valve held close, fluid goes into the cylinder pushes down the piston from TDC to BDC. Just before reaching to BDC high pressure valve close down, but still the piston is moving towards BDC which depressurized the cylinder and then low pressure valve opens, fluid goes through the low pressure manifold. Valve operation sequence for a motoring mode can be understood easily from figure 4.

Number of independent outlet for a digital pump is possible by controlling the valves thus controlling the output and input flow through the piston. In [1] the researcher has developed a concept called
digital pump-motor where three reciprocating piston is used and for each piston there are two actively controlled on/off valve. One valve will control the flow between the piston and tank; other valve will control the flow between the piston and the high pressure side or system. The system layout for a three piston digital pump motor is shown in figure 5.

![Figure 5: Layout of three piston digital pump motor](image)

<table>
<thead>
<tr>
<th>Piston Velocity</th>
<th>Valve A</th>
<th>Valve T</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>+</td>
<td>Close</td>
<td>Close</td>
<td>Pressure Build up in the cylinder, Pre-compression</td>
</tr>
<tr>
<td>+</td>
<td>Open</td>
<td>Close</td>
<td>Pumping to the System</td>
</tr>
<tr>
<td>+</td>
<td>Close</td>
<td>Open</td>
<td>Pumping to the tank</td>
</tr>
<tr>
<td>-</td>
<td>Close</td>
<td>Close</td>
<td>Pressure Release</td>
</tr>
<tr>
<td>-</td>
<td>Open</td>
<td>Close</td>
<td>Motor Mode</td>
</tr>
<tr>
<td>-</td>
<td>Close</td>
<td>Open</td>
<td>Suction</td>
</tr>
<tr>
<td>+/-</td>
<td>Open</td>
<td>Open</td>
<td>Emergency Pressure Relief</td>
</tr>
</tbody>
</table>

*Table 1: Selection of Mode based on Valve Controlling [1]*

In order to run the digital pump-motor in idle mode valve A will be close and valve T will open throughout the cycle, fluid will suck from the tank and pump to the tank only. The total cycle of a digital pump-motor for a single cylinder acting as a pump and motor also with pre-compression and pressure release is shown in figure 6. One important point to notice in the working cycle in figure 6 is that the valve will open only when the pressure differential over the valve is zero [1].
Programmability gives a wide range of advantages in digital displacement technology, as the valves can be controlled and actuated by computer program. Mode of operation can be selected autonomously on each other. Valve delays can be omitted, which is one of the major problems with the passive check valve [1].

Implementing independent outlets can be possible by using one pump-motor for each of the actuator in the system, but in a digital system by increasing the number of control valves independent outlet can be achieved. Figure 7 illustrates an example system with three pistons, two independent outlets, where more control valves are added between the piston and the actuators.
So now each piston has six different action modes. Pumping into outlet A, B or T either intake from A, B and T. As each piston is connected to exactly one outlet or to the tank so a wide range of pressure variation in the outlet can be handled by the machine. Also the flow direction can be controlled by proper valve sequencing [1].

To improve the efficiency of hydraulic system losses should be minimize in all kind of condition. Also according to [1] some pre-condition must have to be in the system to get higher efficiency, such as

1. For both positive and negative power required by all of the actuators in all kind of condition have to be matched with the input hydraulic power.
2. Energy storage system or energy sink is a must to recover and utilize the negative mechanical power for the actuator.
3. To suit the temporary higher power energy source should be in the system.

An efficient hydraulic system can be designed like as showed in figure 8, where all the power flows are two-directional. It can be seen from the figure that utilization of the negative power has a very important role to increase the efficiency of the system [1].

![Image of an efficient hydraulic system](image)

**Figure 8: Efficient hydraulic system [1]**

As negative hydraulic power is an important factor for high efficiency, but the load sensing systems cannot utilize the negative hydraulic power. A comparison between load sensing system and efficient hydraulic system based on loss is showing in figure 9.

![Image of load sensing systems comparison](image)

**Figure 9: Comparison between high efficient system and load sensing system [1]**

So in figure 7 if a third outlet can be connected with an energy storage the 3 piston digital pump-motor with two independent outlet will turned into a high efficient hydraulic system.
The main challenge for the digital displacement machines is that controlling of the valves as there are more controlling option. The control strategy for the system illustrated in figure 7 is as follows.

For the simplification of the controlling one piston is considered as a working unit, also some assumption has to be made for the analysis. The assumption as follows [1]:

1. Compressibility of fluid is neglected.
2. Valve response time is exact without any delay.
3. All the strokes are assumed to be complete
4. Losses are neglected

Every piston has six possible modes, three modes for pumping and three modes for motoring. The selection of the modes is independent to each other, even in case of compressible fluid it is possible by considering the pre-compression and de-compressing for each mode. For the analysis of the controlling M Linjama and K huhtala[1] introduced some initial equations [1] for position, velocity of the piston and flow rates at the output ports.

\[ x_i = \frac{x_{\text{max}}}{2} (1 - \cos (\omega t - 2\pi \left( \frac{i-1}{N} \right))) \]

\[ \dot{x}_i = \frac{x_{\text{max}} \omega}{2} \sin (\omega t - 2\pi \left( \frac{i-1}{N} \right)) \]

\[ Q_A = \sum_{i=1}^{N} u_{Ai} \dot{x}_i A \]

\[ Q_B = \sum_{i=1}^{N} u_{Bi} \dot{x}_i A \]

\[ Q_T = \sum_{i=1}^{N} u_{Ti} \dot{x}_i A \]

Here \( x_i \) is the piston, \( \dot{x}_i \) is velocity, \( Q \) flow, \( A \) area, \( u \) control vector for the valves. Note that the valve opening will change at the TDC or BDC of the piston, when piston will have zero flow rates. As only one valve will open for each stoke so the control vector can be assumed like \([1 \ 0 \ 0], [0 \ 1 \ 0], [0 \ 0 \ 1]\). In figure 8 possible control sequences of three pistons with flow rate of digital pump-motor is described. As the modes are independent from each other so it is possible to run outlet A in a pump mode with all piston and at the same time run outlet B in a motor will all piston, which describes the phenomena of energy transfer form one outlet to other [1].

It is very easy to control and get smoother flow if all of the pistons are connected to one outlet and run as a pump or motor. But in case of partial delivery it is very difficult to control. Some simple sequencing in time can give smooth flow such as: every second, third, fourth piston will pump or piston will not pump in this time instant. Figure 11 represents some result of this time sequencing with increased piston number as well. From figure 11(c) it is clear that with 21 pistons the flow is much smoother [1].
Figure 10: Possible Control Sequence of three-piston pump-motor [1]

<table>
<thead>
<tr>
<th>Piston 1: All pistons pump into outlet A</th>
<th>Outlet A in pump mode, outlet B in motor mode</th>
<th>Outlets A and B in pump mode with 1/3 delivery</th>
</tr>
</thead>
<tbody>
<tr>
<td>[1 0 0] TDC Pump A  [0 0 1] Suck T BDC</td>
<td>[1 0 0] TDC Pump A  [0 1 0] Motor B BDC</td>
<td>[1 0 0] TDC Pump A  [0 0 1] Suck T BDC</td>
</tr>
<tr>
<td>[1 0 0] TDC Pump A  [0 0 1] Suck T BDC</td>
<td>[1 0 0] TDC Pump A  [0 1 0] Motor B BDC</td>
<td>[1 0 0] TDC Pump A  [0 0 1] Suck T BDC</td>
</tr>
<tr>
<td>[1 0 0] TDC Pump A  [0 0 1] Suck T BDC</td>
<td>[1 0 0] TDC Pump A  [0 1 0] Motor B BDC</td>
<td>[1 0 0] TDC Pump A  [0 0 1] Suck T BDC</td>
</tr>
</tbody>
</table>

Figure 11: Flow rates with different control sequence and different number of piston [1]

![Flow rates graphs](image)

The control valves in a digital displacement machines are the vital parts as they are the key of digital displacement machine. Selection of suitable valves and also the suitable valve controlling is a demand for this technology. Several ideas about the digital hydraulic valves and implementation can be found in [6].

There are so many control options in case of digital displacement technology. On/off valves are very easy to control and also programmable so two ways on/off valves can be a good solution for digital displacement.

Switching technology can be used to control the valves by pulse modulation. Figure 12 is showing some implementation of pulse controlled on/off valves and also a parallel connection of the valves.
Controlling of flow area depends on the switching of the frequency. Theoretically average flow area can be unlimited, but due to valve dynamics it has a limited value. Controllability can be improved by using low frequency then pressure pulsation will compensate. In figure 12 (b) an assemble of two way on/off valves is shown, in this case the flow area will be the total flow area of the open valves only. By proper coding the combination of the valves can be controlled. The coding can be binary coding, Fibonacci or pulse number modulation [6].

![Figure 12: Two ways on/off valves and assembly of valves [6]](image)

Control electronics that will be use for controlling the on/off valves in digital displacement machines should have some special features like: For rapid current increase it should have the facility for overvoltage, proper method for reducing energy consumption and to sustain the fast current drop negative voltage should be considered. In parallel connected valve system 8-12 ms response time is enough and direct operated valves, spool valves have this response time. But the digital displacement technology demands faster response from the valves. According to [6] typical switching frequency is 50 hz and the implementation of 10% duty cycle means 2 ms open time for valve. So response time requirement for valve is 1-2 ms but with this kind of response time valves have very little flow rates. That is why improvement of valve switching is an interesting research to most of the fluid power researchers [6].

A system with digital displacement machine is showing in figure 13 with different kind of valve arrangement. Figure 11(a) is showing digital displacement machine with traditional valve controlled actuator. Small volume is used for the smooth flow pulsation. In case of traditional spool valves energy recuperation from the system is not possible, as the valves don’t permit reverse flow. Supply pressure for each valve can optimize easily, which gives an advantage over traditional load sensing system. The system represents in figure 13(b) and 13(c) have the energy recuperation facility as bi-directional valves are used. As two pressures source is available in the system of figure 13(c) so energy can be used in a very efficient way. A direct connection between actuator and digital displacement machine and actuator is shown in figure 13(d). A good controller is a necessary element for this kind of system because flow rates at the outlet must be precisely according to the piston area ratio. This system can be said a loss-less system form the controllability point of view [1].

![Figure 13: Systems with Digital Displacement Machine and different valve system [1]](image)
2.2 Background of Hopsan

Due to strong nonlinearities, rigid differential equation and a high degree of complexity fluid power systems are very difficult to simulate. By Distributed modeling using of transmission line method can be a great solution for simulating complex dynamic systems. To achieve robust numerical properties, components can be numerically isolated from each other, which is possible by distributed modeling using transmission line method [7].

Expertise of transmission line has been implemented in the Hopsan simulation package, developed at Linköping University in the late 1970s. The main goal of the package was to compatibility, execution speed and real time simulation for the system design and optimization of fluid and mechatronic systems. Earlier the package was based on FORTRAN based coding but later on a newer version of HOPSAN package was released and the codes were written in C++. The software package consists of two parts: a core containing all pre-complied component and a graphical user interface, which is shown in figure 14. Core can carried out without graphical interface [7, 8].

Component can be numerically isolated from each other if the system is modeled by using transmission line method. This can be implemented by using a capacitive component (C-type) which physical propagation time relates with one simulation time step. Also introduce a resistive component (Q-type) with a time delay [9].

An example of an element transmission line modeling is described below:

\[ q_1 \quad \text{at} \quad p_1 \quad \text{and} \quad q_2 \quad \text{at} \quad p_2 \]

The set of equation [10] that illustrates the lossless transmission line are:
\[
p_1(t) = p_2(t - T) + Z_{cq1}(t) + Z_{cq2}(t - T)
\]
\[
p_2(t) = p_1(t - T) + Z_{cq2}(t) + Z_{cq1}(t - T)
\]

Here \( p \) and \( q \) are the pressures and flows; \( Z_c \) represents the characteristic impedance of the line; \( T \) is the time delay in the transmission line. In order to solve this equation two new variables and equations are introduced \([10]\):

\[
c_1(t) = p_2(t - T) + Z_{cq2}(t - T)
\]
\[
c_2(t) = p_1(t - T) + Z_{cq1}(t - T)
\]

Information that is transmitted from one node to another node of the transmission line is represented by the new variable \( c_1 \) and \( c_2 \) called the wave variables. Using all the equations finally the boundary equations are found.

\[
p_1(t) = c_1 + Z_{cq1}(t)
\]
\[
p_2(t) = c_2 + Z_{cq2}(t)
\]

These equations can be solved by using boundary equations. This information will be transferred by resistive component and the resistive component will get their boundary conditions from the transmission line component. The block diagram of a transmission line is shown in figure 13.

**Figure 15: Block Diagram of lossless Transmission Line [7]**

The origin of transmission line element method was method of characteristics and it was first used in a simulation package called HYTRAN. The basic of that simulation was PQ-modeling; components calculate pressure from flow or flow from pressure. It was the first concept used in Hopsan but later to get very accurate model of wave propagation of fluid systems the characteristic impedance was introduced. The components in Hopsan are divided into two types: C-type components calculate wave characteristics from flow and pressure by using the impedance. And Q-type components calculate flow and pressure from the wave characteristics, see figure 15. This method gives a scope for parallelized simulation. The components will not simulate simultaneously but they will simulate with a time delay. It gives the thread safety to the multi threaded simulation as same function or variable is not used in the same part of the computer \([8]\).
Most of the commercial simulation package uses the centralized solver but the main power of Hopsan is that it uses the distributed solver. It leads to a benefit from multi-core processing and can easily investigate system with different component. Multi-core processors are now popular as the processed speed is slowed down. Multi-core processors facilitates the execution of several threads (execution of different segment of code) in parallel without forfeiting the speed of execution. The main reason of using parallel simulation is to reduce the total simulation time which can be possible by multi-core processor. The advantage has taken in Hopsan as it is using the multi-core as well as the distributed solver and transmission line method. The components are depends on each other in a centralized solver system so the results are limited when the multi-core simulation is used in simulation packages that are using centralized solver. In Hopsan the components are either C-type or Q-type and never communicate with same type of component due to time delay. Figure 15 is showing the single core and multiple core simulation process. In case of single core running parallel simulation will add extra time to simulation because all the work will done by single core also the time for switching of thread change will add to the simulation time. Simulation time can be reducing by using multiple processing units while the workload will be divided into different cores as shown in figure 15. First the entire C-type component will run in parallel and then when synchronization of all threads finished all Q-type component will run in parallel. Thread creation and synchronization will add some overhead in case of multi-threaded simulation which is one of the limiting factors [7, 8, 11].

**Figure 16: C-type and Q-type component in Hopsan [11]**
To check the performance of multi-threaded simulation several tests were conducted by Braun R, Nordin P, Eriksson B, Krus P [8] and found a very good result in reduction of simulation time. An example model (figure 16) was created with two load-sensing systems with four actuators. The simulation time was 10 seconds with 0.1 milliseconds time step on two quad core machines, with Linux and Windows operating system. [8].

The result is shown in figure 19, where the simulation is 40% faster with quad core. Simulations with two threads have shown better result than with four threads. This may be the overhead cost due to thread creation and synchronization.
Compatibility with other software or other environment is one the most versatile feature of simulation software, which enables co-simulation, multi-domain simulation, data analysis, model generation etc. Automatically or manually exported shared library in DLL format gives the compatibility feature for Hopsan. In the latest version of Hopsan models can be exported to Functional Mock-up Unit and Matlab with suitable interface component that is available in Hopsan component library. [8]

Models can be created by writing C++ files (FORTRAN for old versions) directly or using a Modelica code. In the new versions by using component generator showing in figure 18 user can easily create and compile component (C-type, Q-type, S-type) by using Modelica code or C++ code.

Results can be plot by selecting variable from or list or by clicking of the ports of the component.
3. System Sketch

All the parts of a digital pump are showing below separately.

Pump Component (Hardware and Software):

Cam:

- User Defined Parameter
- Cam Profile (Simple Harmonic, Double harmonic etc.)
- Number of Flanks on Cam (1, 2, 3...)
- Number of Piston that will connect with Cam
- Stoke of the piston

Input

Cam

Output

- Position of the Piston
- Velocity of the Piston
- Force required for the piston

Valve Controller

Digital Pump

- Flow
- Pressure

HP Valve

LP Valve

- Flow/Pressure Source

- Flow/Pressure Source

Pistons

Cam
Piston:
- Output from the cam → Pistions → -Pressure
- Also generate resultant

Valve System:
- Signals from the controller to open and close → - High Pressure Valve
- Low Pressure Valve → - Flow to the system and vice-versa by opening and closing the valve

Valve Controller:
Feedback from pistons → Control Logics → Signal to the Valve system

Total System Sketch:

Figure 21: System Sketch using Standard Symbol for six Piston Digital Displacement Pumps [18]
4. Methodology

4.1 Overview

This research is conducted in order to see how the new technology of digital displacement machine excels in performance as compared to the conventional displacement machines. The advantages and disadvantages as well as the reliability of this machine are also part of the objectives. The research is totally based on simulation study. The aims are thus to model the displacement machine in a simulation tool named Hopsan and simulate it to gather data for validation of the new technology.

In the near past the research on this digital displacement machine technology was focused on simulation study of digital pump/motor with independent outlets with a fixed number of pistons and a simple cam. The current methodological guidelines are designed to facilitate the move towards an evaluation practice focused more on a generic model of the digital displacement machine. The study started from three motivations:

1. To make a generic model of digital displacement machine which will have at least two or three different cam curves and variable number of peaks on it. The existing pump models use general simple harmonic profile for the cam and fixed number of cam peaks on it for simulation. But this digital displacement machine model would be generic to have different cam curves and geometries.
2. To make the number of pistons variable so that user can select it for testing the performance of the machine during simulation.
3. The new pump/motor model should be controllable so that it can run on different controlling systems.

4.2 Research questions

From these general aims, a number of research questions and issues arise. These research issues may be further modified during developing the research design. The questions are as follows:

1. What is the working principle of digital displacement machine?
2. Which mechanical parts can be used to model this machine?
3. How these different parts work?
4. Which simulation tool is convenient to model this machine?
5. How this machine can be controlled?

4.3 Research design

In the course of research design, it required a number of literature reviews and theoretical study of this machine as well as its different parts. The volume contains mainly two parts:

1. Theoretical study of the machine parts such as cam, piston, flywheel, valve etc.
2. Theoretical study of a conventional displacement machine such as radial piston variable displacement pump.
3. Developing equations of the components.
4. Modelling of these parts and the whole machine using Hopsan simulation tool.
5. Simulating the machine with a simple controller system using Simulink simulation tool.

4.4 Components

Before modeling the components and the total displacement machine model in Hopsan, some theoretical study has been done which are given below. The equations used to model the component in Hopsan are also developed at the end of each component description.
4.5 Cam

4.5.1 Definition and Classification
Cam is a rotating or sliding piece (as an eccentric wheel or a cylinder with an irregular shape) in a mechanical linkage used especially in transforming rotary motion into linear motion or vice versa [12]. The motion of cam is followed by a follower which transforms the rotary motion into reciprocating or oscillating motion. There are many types of cams and followers. The follower can be classified according to its movement: Translating follower and Oscillating follower. Again the surface of the follower may be Knife-edged, Flat, Roller or Curved. Some examples of translating followers are given below:

![Different types of translating followers](image)

*Figure 22: Different types of translating followers [13]*

Cams can be classified in various ways. In terms of their shape, cams can be wedge, radial, cylindrical, globoidal, conical, spherical or three-dimensional. Again in terms of the follower motion cams can be dwell-rise-dwell (DRD), dwell-rise-return-dwell (DRRD) or rise-return-rise (RRR).

![Graphical layout of cam-follower system](image)

*Figure 23: Graphical layout of cam-follower system showing maximum displacement of the follower after 90 degrees of cam rotation [13]*
4.5.2 Cam nomenclature

The various cam nomenclatures can be understood by the following illustration:

![Figure 24: Cam nomenclature [13]](image)

The *cam profile* is the actual working surface contour of the cam. It is the surface in contact with the knife-edge, roller surface, or flat-faced follower. The above figure shows a popular cam profile consisting of a single-lobe, external radial cam.

The *base circle* is the smallest circle drawn to the cam profile from the radial cam centre. The cam size is dependent on the size of the base circle. Here the radius of the base circle is \( R_b \).

The *trace point* is the point on the follower located at the knife-edge, roller centre, or spherical-faced centre.

The *pitch curve* is the path of the trace point.

The *prime circle* is the smallest circle drawn to the pitch curve from the cam centre. It is similar to the base circle. Here the radius of the prime circle is \( R_a \).

The *pressure angle* is the angle (at any point) between the normal to the pitch curve and the direction of the follower motion. This is an important parameter in cam design because it represents the steepness of the cam profile. If it is too large, then it can affect the smoothness of the action. Designers often empirically limit the pressure angle to 30 degrees or less for smooth cam-follower action. However, if the follower bearings are strong, the cam-follower is rigid, and the cam-follower overhang is small, the maximum pressure angle may be increased to more than 30 degrees. Here \( \alpha = \) pressure angle and \( \alpha_m = \) maximum pressure angle.

The *pitch point* is the point on the cam pitch having the maximum pressure angle, \( \alpha_m \).

The *pitch circle* is one with its centre at the cam axis passing through the pitch point. The radius of the pitch circle is \( R_p \).
The *transition point* is the position of maximum velocity where the acceleration changes from positive to negative and the inertia force of the follower changes direction accordingly.

### 4.5.3 Cam Curves

In the cam-follower system the follower motion i.e. the displacement, velocity, acceleration and pulse are directed by the geometrical property of the cam. The cam can be designed for any acceptable curve or shape to which the follower responds. The cam curves can be divided into following categories:

1. Basic curves
2. Modified curves
3. Polynomial curves

Basic curves are used for simplicity of mathematical analysis and ease of construction. It is used mainly at low to moderate cam speeds. Basic curves can be divided again into two types:

i. Simple polynomial
ii. Trigonometric

The simple polynomial curve contains

- Straight line or constant velocity
- Parabolic or constant acceleration
- Cubic or constant pulse

And the curves of the trigonometric family are

- Simple harmonic
- Cycloidal
- Double harmonic
- Elliptical

Modified curves are used to improve the performance over basic curves. These basic symmetrical curves are inadequate when high cam speed or special functional motion is required. So modifying i.e. blending different basic curves with each other, the modified curves are produced. There may be many possible modified curves. Some of them are listed below:

- Trapezoidal curve
- Modified trapezoidal curve
- Modified sine curve
- Modified cycloidal curve

Polynomial curves are another choice for cam design. These curves have versatility especially in high speed applications. These polynomials are of higher degrees than the simple one. Some examples of polynomial curves are:

- 2-3 polynomial curve
- 3-4-5 polynomial curve
- 4-5-6-7 polynomial curve and so on.

Finally, in the work, for mathematical analysis and simulation, the following three different curves for the cam profile are chosen:

a. Simple Harmonic Curve
b. Double Harmonic curve

### 4.5.4 Constant velocity curve

This curve has a uniform straight line displacement, constant velocity and zero acceleration diagrams. At the terminals it has impracticable condition of instantaneous change of velocity resulting in
theoretically infinite accelerations. This property makes the curve unacceptable except some modifications.

For constant velocity curve:

\[ y = C \theta \]

When \( \theta = \beta; \ y = h \)

This gives

\[ C = \frac{h}{\beta} \]

Therefore,

Displacement, \( y = \frac{h}{\beta} \theta \)

Velocity, \( y' = \frac{h}{\beta} \)

Acceleration, \( y'' = 0 \)

![Figure 25: Constant velocity curve][1]

### 4.5.5 Simple harmonic motion curve

This is a circular curve having smooth continuous cosine acceleration. This is also known as crank curve. The main disadvantage of this curve is it has a sudden change in acceleration at the dwell ends giving infinite pulses and this is objectionable at high speeds. This curve is popular in combination with other curves.

In Fig. 25 the projection of a radius point P starting at point O moves vertically at point Q along the diameter h of the circle with simple harmonic motion.

Let \( \phi = \) angle of rotations with radius \( \frac{h}{2} \)

The basic harmonic motion displacement function is

\[ y = \frac{h}{2} (1 - \cos \phi) \]

The construction of Fig. 25 uses a circle of radius \( h/2 \). Displacements are taken at angular increments moving through angle \( \phi \) the same increments along the displacement curve. The relationship between angle \( \phi \), the generating circle, and the cam angle \( \theta \) is

\[ \frac{\phi}{\theta} = \frac{\pi}{\beta} \]
Figure 26: Simple harmonic motion curve construction [13]

Substituting the value of $\varphi$:

Displacement, $y = \frac{h}{2}(1 - \cos \frac{x^0}{\beta})$

Velocity, $y' = \frac{hx}{2\beta} \sin \frac{x^0}{\beta}$

Acceleration, $y'' = \frac{hx^2}{2\beta^2} \cos \frac{x^0}{\beta}$

Pulse, $y''' = \frac{hx^3}{2\beta^3} \sin \frac{x^0}{\beta}$

Figure 27: Simple Harmonic curve [14]

4.5.6 Double harmonic motion curve

Double harmonic cam profile have an advantage over simple harmonic motion is that this profile eliminates the high shock and vibration at the beginning of the stroke. The jerk at the beginning of the stroke is small comparable to simple harmonic curve.

The equations for the cam profile are:

Displacement, $y = \frac{h}{2} \left(1 - \cos \frac{x^0}{\beta}\right) - \frac{1}{4} \left(1 - \cos \frac{2x^0}{\beta}\right)$

Velocity, $y' = \frac{hx}{2\beta} \left(\sin \frac{x^0}{\beta} - \sin \frac{2x^0}{\beta}\right)$

Acceleration, $y'' = \frac{hx^2}{2\beta^2} \left(\cos \frac{x^0}{\beta} - \cos \frac{2x^0}{\beta}\right)$
Pulse, $y'' = \frac{hx^3}{2\beta^3} (\sin \frac{2\alpha \theta}{\beta} - \sin \frac{\pi \theta}{\beta})$

**Figure 28: Double harmonic curve [13]**

### 4.5.7 Force analysis

The forces acting on a cam includes inertial force, weight of the elements, external loads, spring forces and frictional forces. Fig. 29 shows an illustration of different forces acting on a cam for both clockwise and counter-clockwise rotation of cam. The friction between the roller follower and the cam profile is neglected here. Forces acting to the left or upward are considered positive and moments are taken positive in counter-clockwise direction according to the custom.

**Figure 29: Forces acting on a cam [14]**

### 4.5.8 Force Calculations

Summation of all horizontal forces must equal to zero to maintain equilibrium.
\[ \sum F_h = 0 = Q_1 - Q_2 - P \tan \gamma \]

Also the sum of all forces in the direction of acceleration equals to the inertial force.

\[ \sum F_v = \pm \frac{W}{g} a = P - W - L - S \mp (\mu Q_1 + \mu Q_2) \]

\[ \therefore P = \pm \frac{W}{g} a + W + L + S \pm (\mu Q_1 + \mu Q_2) \]

Again the sum of the moments with respect to any point in the system should be equal to zero. Here moments are taken about the points “o” and “p”.

\[ \sum M_o = 0 = (P \tan \gamma) mc - Q_2 c \mp \frac{1}{2} \mu Q_1 d \pm \frac{1}{2} \mu Q_2 d \]

\[ \sum M_p = 0 = (P \tan \gamma)(mc + c) - Q_1 c \mp \frac{1}{2} \mu Q_1 d \pm \frac{1}{2} \mu Q_2 d \]

Neglecting the frictional moments we get

\[ (P \tan \gamma) mc - Q_2 c = 0 \]

\[ (P \tan \gamma)(mc + c) - Q_1 c = 0 \]

Therefore,

\[ Q_2 = (P \tan \gamma)m \]

\[ Q_1 = (P \tan \gamma)(m + 1) \]

Now putting these values of \( Q_2 \) and \( Q_1 \) into the equation of \( P \) and solving for \( P \) gives

\[ P = \frac{\pm (W/g)a + W + L + S}{1 \pm \frac{1}{2} \mu (2m + 1) \tan \gamma} \]

The force parallel to cam rod determines the force acting on the piston. Here, in the above equation, the term \( \mu (2m + 1) \tan \gamma \) represents the effect of friction. Denoting it by \( f \), the equation for the force parallel to cam rod becomes

**4.5.9 Parallel force, \( P \)**

\[ P = \frac{\pm (W/g)a + W + L + S}{1 \pm f} \]

The force normal to cam profile determines the contact stress between cam and roller. It is given as follows

**4.5.10 Normal force, \( P_n \)**

\[ P_n = \frac{P}{\cos \gamma} \]
The maximum torque determines the cam shaft load, the power to drive the system and the size of the drive. The torque on the cam shaft is determined by the following formula

\[ T = rP \tan \gamma \]

Again from the definition of pressure angle

\[ \tan \gamma = \frac{V_f}{V_c} \]

where

\[ V_f = \dot{y} = \omega y' \text{ follower velocity, m/sec} \]

\[ V_c = r \omega = \text{cam sliding velocity, m/sec} \]

Substituting in the equation of torque gives

Torque, \( T' \):

\[ \therefore T' = \frac{P \dot{y}}{\omega} \]

Now, the weight of the piston, \( W \), can be discarded from the equation as this is a radial displacement machine having pistons around the cam and cancelling the effect of weight of each other.

A little consideration will show that the forces acting on the cam are the results of several effects. The total average torque acting on the cam is the summation of four different torques resulting from four different phenomena.

So,

\[ \bar{T'} = \bar{T}_i + \bar{T}_s + \bar{T}_f + \bar{T}_p \]

Here,

\( \bar{T}_i = \text{average torque due to piston inertia} \)

\( \bar{T}_s = \text{average torque due to spring force} \)

\( \bar{T}_f = \text{average torque due to viscous friction} \)

\( \bar{T}_p = \text{average torque due to pressure} \)

4.5.11 Inertia effect

The inertial force due to piston mass can be calculated from D’Alembert’s principle which can be stated as:

\[ F_i = -m \ddot{y} \]
The negative sign indicates that the inertial force is acting in the opposite direction of acceleration. Now, the inertial force acting on the cam surface is positive when the piston is accelerating and negative when the piston is accelerating. For simple harmonic cam profile the piston accelerates during the first half of the stroke and then retards during the latter half of the stroke. As a result the direction of inertial force also changes with the change in direction of piston motion. The total average torque applied on the cam surface due to piston inertia over a cycle can be calculated by integrating the torque over one full rotation of cam.

\[ T_i = \int F_i \frac{\dot{y}}{\omega} d\theta \]

\[ T_i = \int_0^{2\pi} m\ddot{y} \frac{\dot{y}}{\omega} d\theta \]

4.5.12 Torque For simple harmonic motion

Linear velocity, \( \dot{y} = \omega y = \omega \left( \frac{hx}{2\beta} \sin \frac{\pi \theta}{\beta} \right) \) and linear acceleration, \( \ddot{y} = \omega^2 \dddot{y} = \omega^2 \left( \frac{h^2 \pi}{2\beta^2} \cos \frac{\pi \theta}{\beta} \right) \)

\[ T_i = \int_0^{2\pi} m\omega^2 \left( \frac{h^2 \pi^2}{2\beta^2} \cos \frac{\pi \theta}{\beta} \right) \left( \frac{h \pi^2}{2\beta^2} \sin \frac{\pi \theta}{\beta} \right) d\theta \]

After integrating the above equation from 0 to \( 2\pi \) we get the average inertial torque due to piston is zero. i.e.

\[ T_i = 0 \]

So, this inertial force is only creating some pulsations in the flow but the net effect of this force over the cam load is zero at the end of the cycle.

4.5.13 Spring effect

In a cam driven pump/motor, especially with multiple strokes in one cam, the pistons must be held in contact with the cam all time. This is because when the piston retards after the half stroke, if the piston is not held by a compression spring, it will jump due to inertial force while retarding. In a closed cam system the spring not needed as the pistons are held back by cam itself while retarding. But in a cam having multiple numbers of peaks, it is not possible to construct a closed cam system. So, a spring with each piston is needed to prevent them from jumping. Usually, with some preload, helical compression springs are used for this purpose.

The spring force is directly proportional to the piston displacement. If the spring force is too small, it will not prevent the piston jumping. Again if the force is too big, it will cause wear of the parts and a stronger design of the system is required. The disadvantage of spring loaded system is that the spring
force adds some extra load on the system. The following figure shows a cam-piston mechanism with its displacement and inertia force curves. As the inertia force is proportional to the acceleration, the acceleration curve can be used to represent the inertial force of the piston.

![Diagram of cam-piston mechanism with displacement and inertia force curves](image)

*Figure 31: Illustration of inertial force and spring force [13]*

The inertia force tends to remove the piston from cam at maximum negative acceleration point as at this point the piston has the maximum negative inertial force. The critical point is defined at the point where the inertia and the spring forces are nearest to each other. This point occurs near the maximum negative acceleration. Generally, the spring force should be higher than the inertial load by 30 to 50 percent due to the strength loss of the spring over a period use. Moreover, at high speeds, forced vibration waves are produced advancing and reflecting throughout the length of the spring. These continuous vibrations further reduce the effective spring force. The spring forces for different cam curves are shown below:

![Diagram of spring forces for different curves](image)

*Figure 32: Inertial force and spring force for different curves [13]*

From the curves, it is seen that lower spring force is required to overcome the inertial load in harmonic curve as compared to the other curves. The spring force can be calculated from Hooke’s law which states:

\[ F_s = -ky \]
Here, \( k \) is the spring rate and \( y \) is the displacement. The negative sign indicates that the force is acting in the opposite direction of the displacement. With some spring preload of \( C \), the spring force becomes

\[
F_s = ky + C
\]

For calculating the spring rate \( k \), the minimum spring force has to be equal to the maximum negative inertial force. So we get,

\[
F_s = \max (F_I)
\]

\[
ky_c + C = |\max (m\dot{y})|
\]

\[
k = \frac{|\max (m\dot{y})| - C}{y_c}
\]

Here, \( y_c \) is the displacement at the critical point i.e. at maximum negative acceleration point.

Now, for a cam driven machine, the piston as well as the spring displacement is changing according to the cam rotation angle. So the torque due to this spring force is also varying with the displacement. The torque due to spring can be expressed by

\[
T_s = \int F_s \frac{\dot{y}}{\omega} \, d\theta
\]

\[
T_s = \int_{0}^{2\pi} (ky + C) \frac{\dot{y}}{\omega} \, d\theta
\]

\[
\bar{T}_s = \int_{0}^{2\pi} \left[ k \left( \frac{h}{2} \left( 1 - \cos \frac{\pi \theta}{\beta} \right) + C \left( \frac{h \pi}{2 \beta} \sin \frac{\pi \theta}{\beta} \right) \right) \right] \, d\theta
\]

After integrating the above equation we get the average spring torque on cam over a whole cycle,

\[
T_s = \frac{h}{16} \left[ \left( \cos \frac{4\pi^2}{\beta} - 1 \right) - \left( \frac{kh}{2} + C \right) \left( \cos \frac{2\pi^2}{\beta} - 1 \right) \right]
\]

4.5.14 Friction effect

In a hydraulic pump/motor the piston is lubricated by the hydraulic oil used in the system which causes viscous friction between the piston and cylinder walls. The flow of liquid lubricants between piston and cylinder clearance causes viscous damping of the piston speed. Assuming laminar lubricant flow through the clearance, this viscous damping force is directly proportional to the piston speed which can be represented by

\[
F_f = -Bv
\]

Here, \( B \) is the viscous damping coefficient and \( v \) is the velocity. The negative sign indicates that the damping force acts in the opposite direction of piston movement. This viscous force is always opposing the piston force and causing some energy losses independent of the direction of piston motion.

So, the average frictional torque can be written as

\[
\bar{T}_f = \int F_f \frac{\dot{y}}{\omega} \, d\theta
\]

\[
T_f = \int_{0}^{2\pi} Bv \frac{\dot{y}}{\omega} \, d\theta
\]
\[ T_f = \int_0^{2\pi} B \frac{\dot{\gamma}^2}{\omega} \, d\theta \]

\[ T_f = \int_0^{2\pi} B \frac{\dot{\gamma}^2}{\omega} \, d\theta \]

\[ \bar{T}_f = \int_0^{2\pi} B \left( \frac{\pi \theta}{2\beta} \sin \frac{\pi \theta}{\beta} \right)^2 \, d\theta \]

Integrating above equation the average frictional torque over the cam becomes

\[ \bar{T}_f = \frac{B h^2 \pi}{16\beta} \left( \frac{4\pi^2}{\beta} - \sin \frac{4\pi^2}{\beta} \right) \]

Implementation of the Cam profile will be shown in the simulation parts later on this report.

### 4.6 Hydraulic Cylinder

A hydraulic cylinder is the working chamber of the pump. It mainly consists of pistons and by the continuous movement of the pistons flow goes in/out of the chamber which affects the pressure.

![Figure 33: ISO 1219 symbol of a Hydraulic Cylinder](image)

Force that will generate form the cam will affect pressure according to the area of the piston. For this work single acting cylinder is chosen. So only the area of the piston will affect the pressure marked red on figure 33.

The pressure from the cylinder can be shown by the simple relation between the force and the pressure

\[ P = \frac{F}{A} \]

Here,

- \( P \) is the pressure from the cylinder
- \( F \) is the force acting of the piston rod
- \( A \) is the area of the piston

For the simulation the hydraulic cylinder is used directly for the inbuilt library of the Hopsan NG.

### 4.7 Digital Valve

In this work 2/2 on off valves form the Hopsan default library will be used to develop the pump model. But the valves will be controlled by a controller.

The reason behind using 2/2 on-off valve is that it is direct acting solenoid actuated valve and response is fast. As these valves are direct operated they respond faster to the signals than poppet vales. This
kind of valves has only two options open or close, which is easy to control. The normally open and close 2/2 on-off valves is showing in figure 29 [15]. As there is so many controls option for the digital displacement machine 2/2 on off valve is a suitable solution.

Figure 34: Normally Closed and Open 2/2 on-off valve [15]

Valve response time is a major factor for digital displacement machines as the valves should open and close immediately with the signals. One of the basic requirements for the valves is that the response time should be fast enough. Also the frequency of the valve should be higher if the pump speed will increase. A simple calculation for the valve frequency related to this work is shown below:

4.7.1 Requirement for the valve

1 Cam rotation requires 2p number of valve operations; \( p = \text{number of flanks on cam profile} \)

Assuming the pump speed is 5000 rpm i.e 83.33 rps

So 83.33 rotations takes place in 1 sec

Therefore 1 rotation takes place in \( \frac{1}{83.33} = 0.012 \) sec = 12 ms

Now 2p number of valve operation has to be done in 12 ms

Therefore 1 valve operation has to be done in \( \frac{12}{2p} = \frac{6}{p} \) ms

If \( p = 1 \) then the valve has to open or close within 6 ms so the valve operation time is 6 ms.

Again if we increase the number peaks in cam profile suppose 6 peaks then the valve operation time should be 1 ms.

The general Equation for the valve operation time:

\[
 t_v = \frac{60000}{2\phi p} \text{ ms}
\]

Here,

\( t_v = \text{valve operation time in milisec} \)

\( \phi = \text{pump speed in rpm} \)

\( p = \text{number of flanks on cam profile} \)

Figure 35 shows the relation between valve operation time, pump speed and the number of flanks on the cam profile.

From the figure if the number of flanks on the cam is 1, pump speed is 3170 rpm, the operation time the valve will be 9.46 millisecond. If the number of flanks on the cam profile increase valve response
time will decrease accordingly. Also for an increased pump speed a faster response from the valve is required.

Figure 35: Valve operation time

Valve frequency: 

\[ f_v = \frac{1000}{\tau_p} \text{ Hz} \]

So if the number of peaks on the cam profile is 6 the valve frequency will be 1000 Hz.
5. Work Progress

5.1 Modeling Cam in Hopsan

The new updated version of Hopsan supports creating components from the equations of that component dynamics. It compiles the equations along with the TLM equations and generates a component in the external library. There are mainly two types of components in Hopsan: C-type component and Q-type component. C-type components calculate wave characteristics from flow and pressure or force and velocity by using the impedance. On the other hand, Q-type components calculate flow and pressure or force and velocity from the wave characteristics. So, the C-type components have to be connected to the Q-type components for running the TLM boundary equations.

Here, the cam is made as a Q-type component as a C-type torque source will be used to drive it and in return it will compute the force and velocity which will be used by the C-type piston connected to it to calculate the wave characteristics. For this Q-type cam, a set of equations relating to the position, velocity and force balance is required to solve TLM boundary equations and compute the force and velocity.

Now the driving equations of the cam component with a harmonic curve can be recalled.

Displacement,

\[ y = \frac{h}{2} \left( 1 - \cos \frac{\pi \theta}{\beta} \right) \]

Torque,

\[ T' = P \frac{\dot{y}}{\omega} \]

Here, the term \( \frac{\dot{y}}{\omega} \) represents the instantaneous torque arm i.e. perpendicular distance from the centre of the cam to the contact point of cam and piston and \( P \) is the total force acting on the cam.

Torque balance:

\[ T - T' = J\omega + \beta \omega \]

Here,

\( T \) -- Input torque, Nm
\( T' \) -- Total load torque, Nm
\( \beta \) -- Dynamic viscosity coefficient, Nms/rad
\( J \) -- Moment of inertia, kgm\(^2\)
\( \omega \) -- Angular speed, rad/s
\( \dot{\omega} \) -- Angular acceleration, rad/s\(^2\)

Now, the load torque is calculated from the total load multiplying it by the instantaneous radius of cam. Again the total load \( P \) is the summation of four different forces.

\[ P = F_i + F_s + F_f + F_p \]

In Hopsan, the C-type cylinder calculates the wave characteristics considering both and at a time and passes the information to the connected Q-type cam. So this cylinder considers the viscous friction force and the pressure force only. The other two forces are added in the equation separately in order to have a proper torque balance. From the torque balance equation, angular speed of cam is
computed. Again, for the cam curve equation, the variable angle is computed by integrating the angular speed of cam. It should be noted that, in transmission line method, the speed is calculated by itself by using the wave characteristics from other components when input is given in the form of force or torque. If the speed is an input, then the required force or torque to achieve that speed is calculated by using the same TLM boundary equations. At last the linear speed is calculated by differentiating the cam curve equation which in turn is used to calculate the linear force by using the TLM boundary equations. So, there are total six variables included namely, torque (T1), angular speed (w1), angle (th1), force (F2), linear speed (v2) and displacement (x2). Total six equations are needed to calculate these six variables. Four of them are basic equations of torque balance, cam curve, angle and linear speed. The other two equations are TLM boundary conditions. These equations are stated below as it was given in the Hopsan component generator:

Basic equations:

\[ T1 - F2 \frac{(v2/w1)}{w1} = J \cdot \text{der}(w1) + B \cdot w1 \]

\[ x2 = \frac{(h/2)}{2} \cdot (1 - \cos(p \cdot th1)) \]

\[ w1 = \text{der}(th1) \]

\[ v2 = \text{der}(x2) \]

\[ m_e 2 = 1 \quad \text{(as connected piston will be a C-type it need an equivalent mass)} \]

TLM boundary equations:

\[ T1 = c1 + Zc1 \cdot w1 \]

\[ F2 = c2 + Zc2 \cdot v2 \]

The above equations are only for a cam that will connect with one piston only.

In the component generator power ports (Linear, rotational etc.) and parameters can be easily added by clicking the add button as show in figure 36.

![Figure 36: Adding ports and parameters on Hopsan component generator](image-url)
But to connect more pistons with the cam phase angle should be added with angle (th1). For example to create a cam that can connect with three pistons 15 equations needs to be solved. In these 15 equations four equations will be TLM equations.

Below figures are showing the procedure of adding ports and parameters and also the Modelica equations for generating the cam component that can connect with three pistons.

![Figure 37: Hopsan component generator showing the list of parameters](image1)

![Figure 38: Hopsan component generator showing the component equations.](image2)

Here again me2, me3, me4 are the equivalent mass needed for three C-type pistons individually. After writing all the equations by clicking compile the component can be compiled, if there is no error the
component will be appeared in the external library of the Hopsan. By changing the phase angle in the equations the number of ports on the liner side where the C-type pistons will be connected can be increased. All generated components will be shown in the Hopsan External library as shown in figure below:

![Component Library](image)

*Figure 39: Generated Components in the Hopsan External Library*

5.1.1 Testing of Developed Cam profile

The generated cam model can be tested by developing a very simple model in Hopsan. The model is shown in figure 40, with a C-type torque source as input, three piston and pressure sources. Also the results for position, velocity are shown below:

![Simple model](image)

*Figure 40: Simple model for testing the Generated Cam component with 3 liner ports*
From figure 41 it can be illustrated that the position and the velocity of the piston is same as expected, as the cam profile is a harmonic profile so the velocity will be a sine curve and the position will be cosine curve. Also the angular velocity of the cam is constant few times after the simulation starts. Constant angular velocity is an important factor for the model because the angle is the integration of angular velocity. Velocity and position depends on the angle as well. So if the angular velocity is constant, angle will increase with time.

But if the number of the flanks on the cam profile increased the angular velocity starts to oscillate also the results of the velocity and position changed. Because as the input is torque now when the number of flanks on the cam profile increased resultant torque is changed and that creates the change in the angular velocity.

After selecting the valve the model form figure 40 changed and after pistons valves are added also with a simple valve controller.
The valve controller will open and close the high pressure valve and the low pressure valve by using the velocity profile of the pistons. The control logic used for the simple controller can be expressed as below:

- If the velocity is greater than zero i.e. the piston will pump to the high pressure side, so the high pressure valve need to be opened and low pressure valve needs to be closed.
- If the velocity is less than zero i.e. the piston will suck from the tank or low pressure side, so the low pressure valve need to be opened and high pressure valve needs to be closed.

There will be a simultaneous opening and closing of the high pressure valve and the low pressure valve. This is an initial idea to test how the model works with the valve opening and close. But the valves should open and close with respect to inside pressure and system pressure. This aspect of valve controlling will be discussed later.

The model with simple valve controller and results are showing below:

![Figure 43: Pump model with three piston and simple controller](image)

**Model Parameters:**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Torque</td>
<td>3000 Nm</td>
</tr>
<tr>
<td>Number of flanks on Cam profile</td>
<td>1</td>
</tr>
<tr>
<td>Stroke</td>
<td>0.1 m</td>
</tr>
<tr>
<td>Dynamic Viscosity</td>
<td>10 Nms/rad</td>
</tr>
<tr>
<td>Moment of inertia</td>
<td>0.02 kg-m^2</td>
</tr>
<tr>
<td>High pressure Source</td>
<td>10 Mpa</td>
</tr>
<tr>
<td>Low pressure Source</td>
<td>1 Mpa</td>
</tr>
<tr>
<td>Resonance frequency of the valve</td>
<td>2500 rad/sec</td>
</tr>
</tbody>
</table>

*Table 2: Parameter for the Model shown in Figure 28*

The result for the angular velocity generated by the torque source is shown below:
The angular velocity is approximately 160 rad/sec or 1526 rpm. But the angular velocity is fluctuating as the input to the pump is a torque source which is not so practical. So the torque source should be replaced with the speed input by adding a feed forward which will be shown in a new model after the result of the present model.

The result from controller related to one high pressure valve and low pressure valve shown below:

The result for the total output flow from the pump is shown in figure 43.
The total output flow from the pump is approximately 420 liter/minute.

To change the source to the pump from torque input to speed input a feed forward need to be added here. The feed forward will measure how much torque will require generating the desired speed and the information will go through the cam component. So the final angular speed from the cam will be the desired speed.

To develop the feed forward the basic torque balance equation of the cam component can be recalled:

\[ T - T' = J\dot{\omega} + \beta\omega \]

Here \( T' \) represents the resultant torque form the pistons. In the above equation all the terms are known except the term \( T \). So rearranging the equation will be:

\[ T = \left( \sum F \ast V \right)/\omega + J\dot{\omega} + \beta\omega \]

The feed forward is described with the figure below:

![Figure 47: Feed forward for the torque source](image)

The same model as figure 43 will be implemented with feed forward and also to test the function ability of the pump a position servo model is developed. The results from the position servo then compare with the result using a pump from Hopsan existing library.

![Figure 48: Position Servo system](image)
Inside System of the pump:

![Diagram of pump system and feed forward](image)

**Figure 49: Inside the Pump system and feed forward**

The model was developed with the following parameter:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Speed</td>
<td>200 rad/s</td>
</tr>
<tr>
<td>Number of flanks on Cam profile</td>
<td>1</td>
</tr>
<tr>
<td>Stroke</td>
<td>0.01 m</td>
</tr>
<tr>
<td>Dynamic Viscosity</td>
<td>10 Nms/rad</td>
</tr>
<tr>
<td>Moment of inertia</td>
<td>0.02 kg-m²</td>
</tr>
<tr>
<td>Low pressure Source</td>
<td>1 Mpa</td>
</tr>
<tr>
<td>Resonance frequency of the valve</td>
<td>2500 rad/sec</td>
</tr>
<tr>
<td>Displacement</td>
<td>$3 \times 10^{-5}$ m³/rev</td>
</tr>
</tbody>
</table>

*Table 3: Parameter for the Model shown in figure 34*

The results are below:

![Graph of angular velocity](image)

**Figure 50: Angular velocity of the model shown in figure 35**

The angular velocity is exactly the same after a certain period. So the feed forward is suitable for the model. Also angular velocity is linear without any fluctuation.
Total output flow from the pump:

![Graph showing total output flow](image1)

*Figure 51: Total output flow of the model shown in figure 48*

The total flow out from the pump is now approximately 60 litre/min. There is also some leakage flow through the valve as the controller is very simple and the signal only follow the velocity profile, so the valve response to the input signal is not fast. To get the better result a good controller is necessary which will measure the pressure difference between the piston and the system and give the signal to the valve accordingly. Also the valve response should be very small with the increment of the input speed to the pump.

The result form the position of the mass is shown below:

![Graph showing position of the mass](image2)

*Figure 52: Position of the mass*

From figure 52 it is clear that the position of the mass is following the input step signal with some fraction of difference. So it can be illustrated that the pump model can work with a physical system. To compare the developed pump model, the example model of position servo (given with the Hopsan
Package) was simulated. The displacement and the speed was the same as the previous model. The example model and the results are illustrated in figure 53 and 54.

By comparing figure 52 and 54 it can be illustrated that the position of the mass has the same result for the both model. But in the developed model there was fraction difference in the position due to the flow fluctuating from the pump. In the example model flow is constant as the pump is developed based on the basic flow equation.

5.2 Generic Model:
In this work the word generic means user will have option to vary the model according to following parameters:

- Cam profile can be choose by the user, but till now there is only one cam profile is developed, later on may be more cam profile will be available.
- Number of flanks on the cam profile can be varied.
- Number of pistons can be varied and also with number of piston high pressure valve and low pressure valve will be vary accordingly.
• For the simulation by varying the stroke parameter the displacement of the pump can be changed, but in practice it is not realistic. To change the displacement in practice the piston should be switched between high pressure side and low pressure side by controlling the valves.

5.2.1 Limitation of the generic model

• Main limitation of the varying model will be that, the cam profile should be available in the external library, so before run the model or testing the generic model user should check what kind of cam profile is available in the external library.
• Variable displace is done only changing the stroke length.

5.2.2 Procedure for making the model generic

Python 2.7 software is used to make the model generic in Hopsan. A python script is attached in the Appendix for the generic model and the procedure is described below:

➢ Defining the function for adding component.
➢ Defining the function for adding component stack.
➢ Defining the function for connecting component to component.
➢ Defining the function for connecting the stack to stack.
➢ Defining the function for connecting stack to component.
➢ To set parameter value using the command form Hopsan user guide.
➢ To set the system parameter using the command form Hopsan user guide.

Some parts of the scripts are shown below for better understanding.

```python
def connectStackToComponent(stackname, port1, component, port2):
    for i in stacks[stackname]:
        hopsan.connect(i, port1, component, port2)
```

*Figure 55: Function to connect stack to component*

Form figure 56 connectStackToComponent is the function name to connect a stack with a component.

```python
hopsan.addComponent("HP", "HydraulicMultiTankC", 2825, 2630, 0)
hopsan.connectStackToComponent("Valvstack1", "PI", "HP", "MP")
hopsan.addComponent("LP", "HydraulicMultiTankC", 2825, 2700, 0)
hopsan.connectStackToComponent("Valvstack2", "PI", "LP", "MP")
```

*Figure 56: Use of Function to connect stack to component*

Figure 56 illustrates the use of the function that will connect stack to the component. Here a valve stack is connected with the tank.

To set the parameter value and also the system parameter the following command is used:

```python
hopsan.setSystemParameter("Stroke", 0.1)
hopsan.setParameter("Cam", "h", "Stroke")
```

*Figure 57: Command for setting system parameter*

After running the python script in the Hopsan python console with the command of the main file user can vary the model. Some examples of model with command are given below:
To develop a simple controller in Matlab/Simulink the model need to be export to Matlab. To export the model and to find the ports in Matlab file for controller development interfaces needs to be connected with the model. For the pump model the interfaces will be connected with the velocity sensors as an output and signals to the valves as an input. As there are systems parameter in Hopsan Model this parameter will appear as a mask in the Matlab/Simulink model.
If the model is developed by using only the existing component of Hopsan library, export function works directly and the model can be compiled in Matlab. But when the model developed with external library some procedure need to be followed to export the model to Matlab.

5.3 Procedures for exporting model from Hopsan to Matlab

- After developing the model in Hopsan with interfaces connected, go to simulation toolbar – Export-Export to Simulink S-function source file-Choose Compiler-Ok. Select one folder for the exported files to be stored.
- But as there is an external library component, when this model will run in Matlab/Simulink, it can’t recognize the external library component. To make the external library component familiar with the Matlab/Simulink the source file of the external library component needs to be compiled with Qt-compiler and the .dll file needs to be added in the same folder with the exported file.
- Also the name of the compiled .dll file needs to be added in the directory.
- Now the model is ready to run in Matlab/Simulink. In the command window of the Matlab HopsanSimulinkCompile.m file needs to be run as a command.
- Open Simulink -New Model-User Defined Function-S-function.
- The S-finction name will be HopsanSimulink.

After doing the above procedure results will be:

Now the model is ready for controlling with the controller.

5.4 Alternative Approach for Developing Pump Model

Without using the component generator the cam component can be developed by using the signal components from the Hopsan existing library. But one of the limitations is that as the signals cannot flow backward so this model can only use as a pump.
5.5 Variable Displacement

As it is mentioned earlier that variable displacement can be done by changing the stroke length of the cylinder. But it is not so realistic. In practice variable displacement can be done by switching the pistons, controlling the high pressure valve and low pressure valve.

In this work variable displacement is done in Matlab/Simulink as it is a part of controller. The logics behind variable displacement illustrated below (Controller logic based on ten piston pump):

- When the pump runs in full displacement setting all pistons will pump to the high pressure side means all high pressure valves will open when the velocity is positive and low pressure valve will open when velocity is negative.
- If the displacement is reduced to 10% of the total displacement, as there is ten pistons so one piston will pump to the high pressure side and rest of the piston will suck from and pump to the low pressure side or tank. So for one piston high pressure valve will open when the velocity is positive and low pressure valve will open when the velocity is negative. For all other pistons high pressure valve will remains close, low pressure valve will remain open for entire cycle.
- With the same logic the pump can run for ten different displacement setting i.e: 10% (one piston will active), 20%(two piston will active), 30%(three piston will active), 40%(four piston...
piston will active), 50%( five piston will active), 60%( six piston will active), 70%( seven piston will active), 80%( eight piston will active), 90%( nine piston will active), 100%( ten piston will active) of the total displacement.

- More partial displacement like 15% is not possible because a piston can’t switch the pressure side instantly. If the pressure side changes instantly in the middle of the stroke vibration and noise will produce from the pump.

Results from the partial displacement will be shown in the next chapter of this report.
6. Results & Analysis

After finishing the procedure of simulation, to analyze the model a pump model with ten pistons is selected and results are taken for analysis. First the results from the Hopsan will be show and later the model will be exported to Matlab and test the variable displacement function ability. To check that the developed pump model can work with hydraulic systems, two example systems was developed and the results from the example system will also described and analyzed in this part of the report.

6.1 Pump Model for analysis

The model shown in figure 62 is the selected pump model with ten pistons and a simple controller for valves. To reduce the flow pulsation an orifice is also introduce in the high pressure side which will damp the fluctuation of the flow and also pressure.

6.2 Results from the Pump model

Figure 63 is showing the flow output from the pump at a pressure difference of 32 bars and input speed is almost 2000 rpm. The flow output is 168 liter/min.

Figure 64: Flow at different speed
Figure 64 and 65 is showing actual flow output from the pump at different speed and pressure difference at different speed. As the speed is increasing flow is increasing as expected.

But at high speed overall efficiency of the pump is reduced and also the leakage flow is increased which is clear from figure 66 and figure 67. As splash losses and pressure losses are proportional to speed, so at higher pressure pump starts to lose efficiency and also the leakage loss increased. At the lower speed actual and ideal flow is almost similar but at the higher speed there is much difference between the two values. For example at 5000 rpm the leakage flow is almost 25% of the ideal flow and the overall efficiency is also less at 5000 rpm.
Valve frequency should be higher also at the higher speed because the pistons are sucking and pumping more at higher speed. So the valves need to be open and close very fast. For 5000 rpm the valve frequency was set 450 Hz. But if the number of peaks on the cam profile increased the valve frequency needs to be increased according to the equation stated in section 4.7.1.

To check that the controller with the variable displacement (from section 5.5) is working fine with this model, the model was transferred to Matlab/Simulink and checks the controller and model with different displacement. The results are below:
To check the adaptability of developed pump model in the Hopsan two system in simulated and examined. Two systems are:

- Open Centre System
- Movement of a mass

### 6.3 Open Centre system and result

![Open Centre System](image1)

**Figure 70: Open Centre System**

![Angular speed output from the transmission](image2)

**Figure 71: Angular speed output from the transmission**

The results shown above are the output speed form an open centre system with the newly developed system. In the motor side the system have different torque loading like 200Nm, 100Nm and 0Nm. The shaft with 0Nm has the highest angular velocity and the power in the output shaft and then the 100Nm, 200 Nm torque loaded shaft. Signal to the valves are given in different time to open the valve and let the flow to the motor. The system works at a maximum pressure difference of 44 MPa.
The reduction of the output speed regarding to the input speed of the pump can be the size of the driven motor and also the rotational inertia with the motor and also the torque from the torque source to the inertia. So power can be switched between the different actuator.

So the pump can work with the system simulation in Hopsan. To check the functionality of the variable displacement the model needs to be export to Matlab/Simulink and check at different variable displacement setting.

![Graph 100% Pump Displacement](image1)

![Graph 10% Pump Displacement](image2)

**Figure 72: Angular speed output from the transmission with variable displacement**

From figure 72 the effect of displacement setting on the output of the transmission shaft is shown. With a full displacement setting of the pump the output speed is higher at different load. But with a 10% displacement the output speed is lower as expected. As for 10% displacement the flow and the pressure will be less to the motor side. Also at less displacement it takes time follow the signal of the valve cause the volume is still the same for the less displacement, so it takes a little time to fill up the volume and pump to the high pressure side. It can be reduced by decreasing the volume with the partial displacement.

### 6.4 System for Moving a Mass and Results

With the newly developed pump model a system for moving mass of 100 Kg (figure 70) was modeled and results for different input speed to the pump was analyzed. 20 MPa was the Maximum pressure difference for the system. The pressure difference was set by a pressure relief valve. Also a 1 liter volume was introduced to damp the oscillation. A step signal was the input to the valve that will permit the flow to the cylinder that will move the mass.
From figure 74 (a) and 74(b) mass moves faster when the input speed to the pump increased as it
increases flow through from the pump. After the mass reaches to its highest position pump is still
pumping to the system but as the mass is in the highest position pressure starts to build up in the
system. When the pressure rises to the maximum pressure difference o the system, pressure relief
valve opens and rest of the flow drains through the pressure relief valve. Form figure 74(c) and 74(d)
pressure relief valve open fast for the increased pump speed as the pressure increases fast at the higher speed.

To check the effect of variable displacement on the mass system, the model was exported to Matlab/Simulink and tested with different displacement settings.

![Figure 75: Mass position and Pressure at different displacement for 1800 rpm](image)

When the pump runs at full displacement it takes very less time for the mass to reach its highest position and after reaching the maximum position of the mass pressure starts to build up in the system and when the pressure reaches to the maximum pressure difference of the system, pressure relief opens and flow drains through the relief valve. At full displacement it takes approximately 2.5 sec to gain the maximum pressure different as the flow is bigger from the pump. But at less displacement like 50% of full displacement to gain the maximum pressure difference it takes almost 3.5 sec and at 10% displacement there is no flow through the relief valve as the pressure cannot goes up to highest pressure difference. Also from figure 75 (a,c,e) after the mass reaches to its maximum position then pressure starts to build up and flow is almost zero from the pump and when the pressure is constant at maximum pressure flow from the pump starts again and it drains through relief valve.
If the input speeds of the pump increase then the flow will increase and also the response of the mass will be faster. Also it will take less time to gain the maximum pressure difference. For same pressure difference but for an increased speed of 3000 rpm the results from the same system are shown in figure 76.

![Figure 76](image_url)

*Figure 76: Mass position and Pressure at different displacement for 3000 rpm*

For the same mass it can be seen that the time for the mass reached to its maximum position is less and also with 100% displacement to achieve the maximum pressure difference for the system is also less compared with 1800 rpm. Also flow through the relief valve is much more as expected. By comparing figure 76 (f) and 75 (f), there is also flow through the relief valve even at 10% displacement for the speed of 3000 rpm.
7. Conclusion

The simulation and analysis of this paper shows that it is possible to develop a generic model of digital displacement pump with the simulation software Hopsan and also co-simulation with Matlab/Simulink is possible. And the developed model can be used for system simulation. Also by using the controller all the on/off valves can be control individually which is one of the key features of a digital displacement machine. But the main hindrance to implement the digital displacement pump in practical may be the lack of faster on/off valves. Also the response time of the valve should be fast enough as the controlling is real time controlling. From some recent research stated in [1] shows that now there are some valves are available like Sturman SI-1500 and “Comb” valve which can satisfy almost the requirement for the digital displacement technology. In this paper more emphasis is given to develop the model and make the model the generic, so the response time of the on/off valve was overlooked.

As all the dynamics of the components considered in the simulation so the results found from the simulation is close to real results. But the hydraulic cylinder used for simulation has a large end stroke damping for that the flow out from the pump is little less from actual. For a further recommendation for the Hopsan may be the correction of the end stroke damping length.

Hopsan is robust software for simulation complex hydraulics system as it uses the transmission line method and also its uses the multicore for simulation. It consumed less time to simulate complex hydraulic system. Also plotting is very user friendly as all the desired data can be plotted in same plot area with different axis, which is comprehensive for comparing different results.

While working with the simulation model results might be affected by some errors. One the major is that, the models are always an approximation of real model. So the precision of the results depends on the accuracy of the approximations. As there was no previous model of digital displacement pump to compare the result of this newly developed digital displacement pump by using Hopsan, so the performance of the machine was compared with the performance of the traditional machine which might lead to a source of error. For simulation there is necessity for solving system of equations so numerical error might be another source of error.

This is a very first simulation approach for digital displacement machine by using simulation software Hopsan and shows a promising future for robust modeling.

I have gained detail knowledge about digital displacement technology, implementation of new components and simulation in Hopsan. Also I experienced the basic of co-simulation between Matlab/Simulink and Hopsan. I think the knowledge that I have achieved during the project will be helpful for my future career.
7.1 Future Work

As digital displacement technology is a very immersing technology in the field of fluid power so there are much more option to work with in near future. So along with the following future work the main goal of a digital displacement can be fulfilled.

- The simulation was done with a simple controller but it would be interesting to examine the possibilities of using advanced control algorithm such as by using the pressure difference before and after the valve, using pre-compression.
- Build the controller in Hopsan by using the C++ coding to avoid the co-simulation and to examine the performance only in Hopsan.
- Develop a generic controller which will vary accordingly with the model.
- More cam profile can be developed to check the function ability of the cam based machines. Also optimization of the cam profile can be done.
- Developing a C-type cam component so that it can be directly connected with the speed source without any feed forward. Also add a flywheel in the input shaft.
- The digital displacement pump has shown promising results in the simulation it would be interesting to build prototype of proper configuration and experiment it in the laboratory.
References


[16] Hopsan User Guide


[18] Yigen C., Control of a Digital Displacement Pump, Department of Energy Technology, Aalborg University, 2012.
Appendix

Python Script for Pump with external library:

from math import
stacks = {}
def addVerticalStack( stackname, basename, type, x, y, rot, dist, n):
    stack = []
    for i in range(0, n):
        print(basename+str(i))
        print(hopsan.addComponent(basename+str(i), type, x-(n/2.0-0.5-i)*dist,rot))
        stack.append(basename+str(i))
    global stacks
    stacks[stackname]=stack
def addHorizontalStack( stackname, basename, type, x, y, rot, dist, n):
    stack = []
    for i in range(0, n):
        print(basename+str(i))
        print(hopsan.addComponent(basename+str(i), type, x-(n/2.0-0.5-i)*dist,y,rot))
        stack.append(basename+str(i))
    global stacks
    stacks[stackname]=stack
def connectStackToComponent( stackname, port1, component, port2):
    for i in stacks[stackname]:
        hopsan.connect(i, port1, component, port2)
def connectStackToStack( stackname1, port1, stackname2, port2):
    for i in range(len(stacks[stackname1])):
        hopsan.connect(stacks[stackname1][i], port1, stacks[stackname2][i], port2)
def connectStackToCam( stackname, port1, cam, port2):
    x=1
    for i in stacks[stackname]:
        hopsan.connect(i, port1, cam, port2+str(x))
    x=x+1
def pumpwithexlib(n):
    global stacks
    stacks = {}

    hopsan.addComponent("TorqueSource", "MechanicTorqueTransformer", 2750, 2175, 180)
    hopsan.addComponent("Cam", "Cam"+str(n), 2625, 2150, 180)
    hopsan.connect("TorqueSource", "P1", "Cam", "PR")
    addHorizontalStack("PistonStack", "C-type Piston", "HydraulicCylinderC", 2600, 2290, 0, 150, n)
    addHorizontalStack("LeakageStack", "Q-type Pressure Source", "HydraulicPressureSourceQ", 2625, 2330, 0, 150, n)
    connectStackToStack("PistonStack", "P2", "LeakageStack", "P1")
    addHorizontalStack("VelocitySensorStack", "Velocity Sensor", "MechanicSpeedSensor", 2600, 2240, 90, 150, n)
    connectStackToStack("PistonStack", "P3", "VelocitySensorStack", "P1")
    addHorizontalStack("VelocityOutputStack", "Velocity", "SignalOutputInterface", 2640, 2240, 0, 150, n)
    connectStackToStack("VelocityOutputStack", "in", "VelocitySensorStack", "out")
    addHorizontalStack("ValveStack1", "HP Valve", "Hydraulic22DirectionalValve", 2625, 2450, 0, 170, n)
    connectStackToStack("PistonStack", "P1", "ValveStack1", "P2")
    addHorizontalStack("ValveStack2", "LP Valve", "Hydraulic22DirectionalValve", 2675, 2550, 0, 170, n)
    connectStackToStack("PistonStack", "P1", "ValveStack2", "P2")
    hopsan.addComponent("HP", "HydraulicMultiTankC", 2625, 2630, 0)
    connectStackToComponent("ValveStack1", "P1", "HP", "MP")
    hopsan.addComponent("LP", "HydraulicMultiTankC", 2625, 2700, 0)
    connectStackToComponent("ValveStack2", "P1", "LP", "MP")
    addHorizontalStack("SignalInputStack1", "HP Valve Signal", "SignalInputInterface", 2553, 2500, 0, 170, n)
    connectStackToStack("SignalInputStack1", "out", "ValveStack1", "in")
    addHorizontalStack("SignalInputStack2", "LP Valve Signal", "SignalInputInterface", 2603, 2600, 0, 170, n)
    connectStackToStack("SignalInputStack2", "out", "ValveStack2", "in")
    hopsan.addComponent("Sum", "SignalSum", 2625, 2770, 0)
    connectStackToComponent("FlowSensorStack1", "out", "Sum", "in")
    hopsan.addComponent("Total Flow", "SignalSink", 2680, 2770, 0)
    hopsan.connect("Sum", "out", "Total Flow", "in")
connectStackToCam("PistonStack", "P3", "Cam", "P")

#System Parameter
hopsan.setSystemParameter("Stroke", 0.1)
hopsan.setParameter("Cam", "h", "Stroke")
hopsan.setSystemParameter("NumberOfPeaks", 1)
hopsan.setParameter("Cam", "p", "NumberOfPeaks")
hopsan.setSystemParameter("DynamicViscosity", 10)
hopsan.setParameter("Cam", "B", "DynamicViscosity")
hopsan.setSystemParameter("MomentofInertia", 0.02)
hopsan.setParameter("Cam", "J", "MomentofInertia")
hopsan.setSystemParameter("HighPressure", 100000)
hopsan.setParameter("HP", "p", "HighPressure")
hopsan.setSystemParameter("LowPressure", 100000)
hopsan.setParameter("LP", "p", "LowPressure")

# feed forward
hopsan.addComponent("w", "SignalStep", 2850, 2120, 0)
hopsan.setSystemParameter("Speed", 200)
hopsan.setParameter("w", "y", "Speed")
hopsan.addComponent("delay", "SignalTimeDelay", 2890, 2065, 0)
timeStep = hopsan.getTimeStep()
hopsan.setParameter("delay", "deltat", timeStep)
hopsan.connect("w", "out", "delay", "in")
hopsan.addComponent("subtract", "SignalSubtract", 2940, 2120, 0)
hopsan.connect("w", "out", "subtract", "in1")
hopsan.connect("delay", "out", "subtract", "in2")

hopsan.addComponent("B", "SignalGain", 2940, 2170, 0)
hopsan.setParameter("B", "k", "DynamicViscosity")
hopsan.connect("w", "out", "B", "in")
hopsan.addComponent("T_s", "SignalSource", 3020, 2065, 0)
ts = hopsan.getTimeStep()
hopsan.setParameter("T_s", "y", ts)
hopsan.addComponent("divide", "SignalDivide", 3020, 2120, 0)
hopsan.connect("subtract", "out", "divide", "in1")
hopsan.connect("T_s", "out", "divide", "in2")
hopsan.addComponent("J", "SignalSource", 3090, 2065, 0)
hopsan.setParameter("J", "y", "MomentofInertia")
hopsan.addComponent("multiply", "SignalMultiply", 3090, 2120, 0)
hopsan.connect("divide", "out", "multiply", "in1")
hopsan.connect("J", "out", "multiply", "in2")
addHorizontalStack("FourceSensorStack", "Force Sensor", "MechanicForceSensor", 2680, 2240, 0, 150, n)
connectStackToStack("PistonStack", "P3", "FourceSensorStack", "P1")
addHorizontalStack("multiplyStack", "forcexvelocity", "SignalMultiply", 2640, 2200, 270, 150, n)
connectStackToStack("FourceSensorStack", "out", "multiplyStack", "in2")
connectStackToStack("VelocitySensorStack", "out", "multiplyStack", "in1")
hopsan.addComponent("sum_Fxv", "SignalSum", 2525, 2170, 270)
connectStackToComponent("multiplyStack", "out", "sum_Fxv", "in")
hopsan.addComponent("w_system", "MechanicAngularVelocitySensor", 2750, 2090, 0)
hopsan.connect("w_system", "P1", "TorqueSource", "P1")
hopsan.addComponent("divide1", "SignalDivide", 2750, 2040, 0)
hopsan.connect("w_system", "out", "divide1", "in2")
hopsan.connect("sum_Fxv", "out", "divide1", "in1")
hopsan.addComponent("T", "SignalSum", 3150, 2140, 0)
hopsan.connect("multiply", "out", "T", "in")
hopsan.connect("B", "out", "T", "in")
hopsan.connect("divide1", "out", "T", "in")
hopsan.connect("T", "out", "TorqueSource", "in")
hopsan.addComponent("T", "SignalGreaterThan", 3150, 2140, 0)
Python Script for alternative approach (Using inbuilt library):

from math import *

stacks = {}

def addVerticalStack( stackname, basename, type, x, y, rot, dist, n ):
    stack = []
    for i in range(0, n):
        print(basename+str(i))
        print(hopsan.addComponent(basename+str(i), type, x,y-(n/2.0-0.5-i)*dist,rot))
        stack.append(basename+str(i))
    global stacks
    stacks[stackname]=stack

def addHorizontalStack( stackname, basename, type, x, y, rot, dist, n ):
    stack = []
    for i in range(0, n):
        print(basename+str(i))
        print(hopsan.addComponent(basename+str(i), type, x-(n/2.0-0.5-i)*dist,y,rot))
        stack.append(basename+str(i))
    global stacks
    stacks[stackname]=stack

def connectStackToComponent( stackname, port1, component, port2):
    for i in stacks[stackname]:
        hopsan.connect(i, port1, component, port2)

def connectStackToStack( stackname1, port1, stackname2, port2):
    for i in range(len(stacks[stackname1])):
        hopsan.connect(stacks[stackname1][i], port1, stacks[stackname2][i], port2)

def connectStackToCam( stackname, port1, cam, port2):
    x=1
    for i in stacks[stackname]:
        hopsan.connect(i, port1, cam, port2+str(x))
    x=x+1
def connectStackToCamSystem(stackname, port1, cam, port2):
    x=0
    for i in stacks[stackname]:
        hopsan.connect(i, port1, cam, port2+str(x))
    x=x+1

def connectStackToPorts(stackname1, port1, stackname2, port2):
    x=0
    for i in range(len(stacks[stackname1])):  
        hopsan.connect(stacks[stackname1][i], port1, stacks[stackname2][i], port2+str(x))
    x=x+1

def speedpumpsytem(n):
    global stacks
    stacks = {}
    hopsan.addComponent("PumpSystem", "Subsystem", 2400, 2400, 0)
    hopsan.enterSystem("PumpSystem")
    hopsan.addComponent("CamSystem", "Subsystem", 2600, 2170, 0)
    hopsan.enterSystem("CamSystem")
    addHorizontalStack("CamStack", "Cam", "Subsystem", 2600, 2400, 0, 150, n)
    for i in range(0, n):
        ## Cam (1st subsystem)
        hopsan.enterSystem("Cam"+str(i))
        hopsan.addComponent("w", "HopsanGUIContainerPort", 2430, 2500, 0)
        hopsan.addComponent("Integrator1", "SignalIntegrator2", 2500, 2500, 0)
        hopsan.connect("w", "w", "Integrator1", "in")
        hopsan.addComponent("Add1", "SignalAdd", 2550, 2500, 0)
        hopsan.connect("Add1", "in1", "Integrator1", "out")
        hopsan.addComponent("phi", "HopsanGUIContainerPort", 2550, 2400, 0)
        hopsan.connect("Add1", "in2", "phi", "phi")
        hopsan.addComponent("Multiply2", "SignalMultiply", 2600, 2500, 0)
        hopsan.connect("Add1", "out", "Multiply2", "in1")
hopsan.addComponent("p", "HopsanGUIContainerPort", 2600, 2400, 0)
hopsan.connect("p", "p", "Multiply2", "in2")
hopsan.addComponent("Cos", "SignalCos", 2650, 2500, 0)
hopsan.connect("Cos", "in", "Multiply2", "out")
hopsan.addComponent("Subtract2", "SignalSubtract", 2700, 2550, 0)
hopsan.connect("Cos", "out", "Subtract2", "in2")
hopsan.addComponent("1", "SignalSource", 2600, 2550, 0)
hopsan.connect("1", "out", "Subtract2", "in1")
hopsan.addComponent("Multiply3", "SignalMultiply", 2750, 2550, 0)
hopsan.connect("Multiply3", "in1", "Subtract2", "out")
hopsan.addComponent("Half", "SignalGain", 2750, 2500, 90)
hopsan.connect("Multiply3", "in2", "Half", "out")
hopsan.setParameter("Half", "k", 0.5)
hopsan.addComponent("h", "HopsanGUIContainerPort", 2750, 2400, 0)
hopsan.connect("h", "h", "Half", "in")
hopsan.addComponent("Multiply4", "SignalMultiply", 2720, 2600, 0)
hopsan.connect("Multiply4", "in2", "h", "h")
hopsan.connect("Multiply4", "in1", "p", "p")
hopsan.addComponent("Multiply5", "SignalMultiply", 2750, 2650, 0)
hopsan.connect("Multiply4", "out", "Multiply5", "in2")
hopsan.addComponent("Sin", "SignalSin", 2710, 2650, 0)
hopsan.connect("Sin", "in", "Multiply2", "out")

hopsan.addComponent("Position & Velocity Source", "MechanicVelocityTransformer", 2800, 2600, 0)
hopsan.connect("Position & Velocity Source", "xin", "Multiply3", "out")
hopsan.connect("Position & Velocity Source", "vin", "Multiply5", "out")
hopsan.addComponent("xv", "HopsanGUIContainerPort", 2870, 2600, 0)
hopsan.connect("Position & Velocity Source", "Pm1", "xv", "xv")

hopsan.exitSystem()
## Taking out the parameters outside of the Cam (1st subsystem)

```python
hopsan.addComponent("h", "HopsanGUIContainerPort", (2300-50*n), 2300, 0)
connectStackToComponent("CamStack", "h", "h", "h")
hopsan.addComponent("p", "HopsanGUIContainerPort", (2300-50*n), 2400, 0)
connectStackToComponent("CamStack", "p", "p", "p")
hopsan.addComponent("w", "HopsanGUIContainerPort", (2300-50*n), 2500, 0)
connectStackToComponent("CamStack", "w", "w", "w")
addHorizontalStack("xvStack", "xv", "HopsanGUIContainerPort", 2645, 2500, 0, 150, n)
connectStackToPorts("CamStack", "xv", "xvStack", "xv")
addHorizontalStack("serialStack", "serial", "SignalSource", 2530, 2300, 0, 150, n)
addHorizontalStack("phiStack", "phi", "SignalGain", 2575, 2300, 0, 150, n)
p=0
for j in range(0, n):
    hopsan.setParameter("serial"+str(j), "y", p)
    p=p+1
    hopsan.setParameter("phi"+str(j), "k", 6.283185/n)
connectStackToStack("serialStack", "out", "phiStack","in")
connectStackToStack("phiStack", "out", "CamStack", "phi")
hopsan.exitSystem()
```

## Adding all the ports

```python
hopsan.addComponent("h", "HopsanGUIContainerPort", 2500, 2100, 0)
hopsan.connect("h", "h", "CamSystem", "h")
hopsan.addComponent("p", "HopsanGUIContainerPort", 2600, 2100, 0)
hopsan.connect("p", "p", "CamSystem", "p")
hopsan.addComponent("w", "HopsanGUIContainerPort", 2700, 2100, 0)
hopsan.connect("w", "w", "CamSystem", "w")
```

# Bringing & connecting pistons, valves, tanks etc

```python
addHorizontalStack("PistonStack", "Piston", "HydraulicCylinderC", 2600, 2290, 0, 150, n)
addHorizontalStack("LeakageStack", "Leakage", "HydraulicPressureSourceQ", 2625, 2330, 0, 150, n)
connectStackToStack("PistonStack", "P2", "LeakageStack", "P1")
```
addHorizontalStack("ValveStack1", "HP Valve", "Hydraulic22DirectionalValve", 2625, 2430, 0, 170, n)
connectStackToStack("PistonStack", "P1", "ValveStack1", "P2")
addHorizontalStack("ValveStack2", "LP Valve", "Hydraulic22DirectionalValve", 2675, 2550, 0, 170, n)
connectStackToStack("PistonStack", "P1", "ValveStack2", "P2")
hopsan.addComponent("HP", "HydraulicMultiTankC", 2625, 2630, 0)
connectStackToComponent("ValveStack1", "P1", "HP", "MP")
hopsan.addComponent("LP", "HydraulicMultiTankC", 2625, 2700, 0)
connectStackToComponent("ValveStack2", "P1", "LP", "MP")
addHorizontalStack("SignalInputStack1", "HP Valve Signal", "HopsanGUIContainerPort", 2553, 2500, 0, 170, n)
connectStackToPorts("ValveStack1", "in", "SignalInputStack1", "HP Valve Signal")
addHorizontalStack("SignalInputStack2", "LP Valve Signal", "HopsanGUIContainerPort", 2603, 2600, 0, 170, n)
connectStackToPorts("ValveStack2", "in", "SignalInputStack2", "LP Valve Signal")
connectStackToStack("FlowSensorStack1", "P1", "ValveStack1", "P2")
connectStackToStack("FlowSensorStack2", "P1", "ValveStack2", "P2")
hopsan.addComponent("Sum", "SignalSum", 2625, 2770, 0)
connectStackToComponent("FlowSensorStack1", "out", "Sum", "in")
connectStackToComponent("FlowSensorStack2", "out", "Sum", "in")
hopsan.addComponent("Total Flow", "HopsanGUIContainerPort", 2680, 2770, 0)
hopsan.connect("Sum", "out", "Total Flow", "Total Flow")
connectStackToCamSystem("PistonStack", "P3", "CamSystem", "xv")
addHorizontalStack("VelocitySensorStack", "Velocity Sensor", "MechanicSpeedSensor", 2600, 2240, 90, 150, n)
connectStackToStack("PistonStack", "P3", "VelocitySensorStack", "P1")
addHorizontalStack("VelocityOutputStack", "Velocity", "HopsanGUIContainerPort", 2670, 2230, 0, 150, n)
connectStackToPorts("VelocitySensorStack", "out", "VelocityOutputStack", "Velocity")
hopsan.exitSystem()
hopsan.addComponent("h", "SignalSource", 2350, 2300, 0)
hopsan.setSystemParameter("Stroke", 0.1)
hopsan.setParameter("h", "y", "Stroke")
hopsan.connect("h", "out", "PumpSystem", "h")
hopsan.addComponent("p", "SignalSource", 2380, 2300, 0)
hopsan.setSystemParameter("NumberOfPeaks", 1)
hopsan.setParameter("p", "y", "NumberOfPeaks")
hopsan.connect("p", "out", "PumpSystem", "p")
hopsan.addComponent("w", "SignalStep", 2410, 2300, 0)
hopsan.setSystemParameter("Speed", 200)
hopsan.setParameter("w", "y_A", "Speed")
hopsan.setParameter("w", "t_step", "0")
hopsan.connect("w", "out", "PumpSystem", "w")
addVerticalStack("InputStack1", "HP Valve Signal", "SignalInputInterface", 2150, 2400, 0, 50, n)
connectStackToCamSystem("InputStack1", "out", "PumpSystem", "HP Valve Signal")
addVerticalStack("InputStack2", "LP Valve Signal", "SignalInputInterface", 2250, 2500, 0, 50, n)
connectStackToCamSystem("InputStack2", "out", "PumpSystem", "LP Valve Signal")
addVerticalStack("VelocityStack", "Velocity", "SignalOutputInterface", 2575, 2400, 0, 50, n)
connectStackToCamSystem("VelocityStack", "in", "PumpSystem", "Velocity")
hopsan.addComponent("Total Flow", "SignalOutputInterface", 2650, 2400, 0)
hopsan.connect("Total Flow", "in", "PumpSystem", "Total Flow")
Code for Simple variable controller:

```matlab
function [x0,y0] = fcn(u0,e)
    if e == 1.0 && u0 >= 0
        x0 = 1;
        y0 = 0;
    elseif e == 1.0 && u0 < 0
        x0 = 0;
        y0 = 1;
    elseif e == 0.9 && u0 >= 0
        x0 = 1;
        y0 = 0;
    elseif e == 0.9 && u0 < 0
        x0 = 0;
        y0 = 1;
    elseif e == 0.8 && u0 >= 0
        x0 = 1;
        y0 = 0;
    elseif e == 0.8 && u0 < 0
        x0 = 0;
        y0 = 1;
    elseif e == 0.7 && u0 >= 0
        x0 = 1;
        y0 = 0;
    elseif e == 0.7 && u0 < 0
        x0 = 0;
        y0 = 1;
    elseif e == 0.6 && u0 >= 0
        x0 = 1;
        y0 = 0;
    elseif e == 0.6 && u0 < 0
        x0 = 0;
        y0 = 1;
    else
        x0 = 0;
        y0 = 0;
end
```
y0 = 1;
elseif e == 0.5 && u0 >= 0
    x0 = 1;
    y0 = 0;
elseif e == 0.5 && u0 < 0
    x0 = 0;
    y0 = 1;
elseif e == 0.4 && u0 >= 0
    x0 = 1;
    y0 = 0;
elseif e == 0.4 && u0 < 0
    x0 = 0;
    y0 = 1;
elseif e == 0.3 && u0 >= 0
    x0 = 1;
    y0 = 0;
elseif e == 0.3 && u0 < 0
    x0 = 0;
    y0 = 1;
elseif e == 0.2 && u0 >= 0
    x0 = 1;
    y0 = 0;
elseif e == 0.2 && u0 < 0
    x0 = 0;
    y0 = 1;
elseif e == 0.1 && u0 >= 0
    x0 = 1;
    y0 = 0;
elseif e == 0.1 && u0 < 0
    x0 = 0;
    y0 = 1;
else
    x0 = 0;
    y0 = 0;
end