THESIS FOR THE DEGREE OF LICENTIATE OF ENGINEERING

An Interactive Traction Motors Design and Selection Software

by

Hasan El Hinaoui-Hamze

Department of Energy and Environment
Division of Electric Power Engineering
Electrical Machines and Drive Systems Group
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden, 2005

An Interactive Traction Motors Design and Selection Software

Hasan El Hinaoui-Hamze

Technical report No. 4
Department of Energy and Environment
Division of Electric Power Engineering
Electrical Machine and Drive Systems Group
Chalmers University of Technology
SE-412 96 Göteborg
Sweden

Telephone: +46 (0) 31 772 1654
Fax: +46 (0) 31 772 1224
www.elteknik.chalmers.se/~EMDS/EMDS/index.htm

Abstract

Use of electric motors in vehicle traction, in fact, predates the internal combustion engines (ICEs). Electric vehicles powered by dc motors were known as far back as 1890. Compared with vehicles powered by ICEs, the main drawback of the early electric vehicles was, and still is, their limited range and long recharging time. To overcome this, in recent years, developments have been focused on hybrid electrical vehicles (HEVs) combining two sources of energy: an ICE of a conventional vehicle and an electric motor (or motors). Such a hybrid vehicle enables the driver (or the vehicle computerised energy management system) to decide which source of power is appropriate for a particular journey.

Drive Trains for HEVs are of course more complex than both conventional ICEs and totally electric vehicles. Design of HEVs, therefore, encompasses several technologies and the vehicle designer cannot be expected to be an expert in all related fields. In particular, it is recognised that vehicle designers may not be familiar with electric drive systems. The work reported here aims to provide a design tool to enable decisions to be made on type and feasibility of traction motors to meet specific traction requirements.

In the first part of the software described in this thesis normalized curves, giving theoretically available force density values for different cooling arrangements, are utilised, after adopting piecewise linear approximations, to provide the vehicle designer with a quick answer to whether or not drive requirements can be met within a specified space envelope. Once the requirements are deemed feasible, the user can progress to the second part of the programme in which detailed design work for a selected drive topology is carried out. Thereafter, the user can export drive details into a number of commercially available CAE packages to perform further investigations (e.g. dynamic performance).

Due to time constraints, only induction motor systems are fully developed in the reported work. However, the design of the software allows for modules relating to conventional dc, reluctance and brushless dc drives to be added at a later stage.

Keywords

- Electric Motors (EMs)
- Electric Drives
- Traction Motors
- Internal Combustion Engines (ICEs)
- Hybrid Electric Vehicles (HEVs)
- CAD
- SQL
- Visual Basic
- Access
- MagNet
- MATLAB

Dedication

This thesis is dedicated to my beloved family: my father, my mother, my sisters, and my brother, I just want to thank them for their endless love and unlimited support.

Acknowledgements

The author wishes to express his gratitude to the Chalmers Foundation for providing financial support during the last year of his study.

He is grateful to his supervisor and examiner, Professor Essam Hamdi for providing a relaxed, but efficient, work environment and for his guidance and support throughout his studies.

The work was undertaken as a part of the *GreenCar* project in collaboration with Lund Technical University. Thanks are due to Professor Mats Alaküla and Raissa Öttl for useful discussions and constructive criticism throughout the work. Thanks are also due to Joachim Lindström, of Volvo, and Lars Hoffmann, of SAAB, for their help during the early part of the project.

The author wishes to express his heartfelt thanks to the members of the Electrical Machines and Drive Systems Group, with whom he had numerous fruitful discussions over different issues (not to mention the random turns of the launch).

He also would like to thank his colleagues in the Department (now Division) of Electrical Power Engineering, especially those regulars at the wine parties. Their company was really enjoyable.

Table of Contents

Chapter 1: Electric and Hybrid Electric Vehicles

Chapter 2: General Overview of the Developed Software

Chapter 3: Package Analysis

Chapter 4: Conclusions and Future Work

Appendix I

Appendix II

Appendix III

CHAPTER I

Electric and Hybrid Electric Vehicles

1.1. INTRODUCTION

Electric traction in vehicles can be accomplished with a number of different topologies. The electrical machines can be mounted on the same shaft as the Internal Combustion Engine (ICE), in or after the transmission, directly at the wheels, or in combinations of these locations. They can be several different types, like synchronous AC (Permanently or electrically magnetized), induction, reluctance or even conventional DC though this type is less relevant today. They can be made for high torque at low speed, preferably for direct drive at the wheels, or for high speed and low torque and operate via a transmission system. They machine topology can be of radial flux (of the interior or external rotor type), axial flux, transverse flux type etc. The choice of course also defines the type and size of power electronic converter.

The choice of topology for a given application will inevitably be a compromise where such properties as torque density, power density, efficiency, cost, life length and safety are among the important inputs.

The traction motors in commercial and concept vehicles used today represent most of the types and configurations mentioned above. It is not a straightforward task to compare the different solutions. How much lighter, smaller, cheaper, quieter, more efficient, and more reliable would a particular choice be depend not only on the traction motor type but also on the overall vehicle system.

Once a specific motor type and location is selected, there is a need to represent the properties of that particular type in a simulation program for vehicle simulation. The control method used for e.g. torque control and limitations related to the power electronics, temperature etc, must be taken into account in the model. The work presented here focuses on the traction motor selection and design. It is the intention that this work, when modules for most drive topologies are added, will be integrated in future into an overall simulation package of the vehicle electrical system.

1.2. Electric Vehicles

Electric vehicle means a vehicle, which is powered by an electric motor drawing current from rechargeable storage batteries instead of gasoline or diesel fuel. In general, electric vehicles are powered by batteries and are driven by one or more electric motors. The first electric vehicle dates back to 1890 when William Morrison built one that could travel for 13 hours at a speed of 22.5 km/h [4]. Other developments are as follows:

- As early as 1899, electric taxis, trams, and omnibuses were commonly seen in major cities.
- Between 1900 and 1915 more than 60 American companies were building electric vehicles.
- Between 1896 and 1901 the Andrew Ricker Company offered a wide range of styles and models.

In (1899-1914) the Walker Baker Company created an electric-powered race car that could go over 120 km/h and it was the first car to have passenger seat belts [4].

Electric vehicles of course are environment friendly. They are clean, quiet and simple to operate. They consist of batteries, traction motor, motor speed controller, an interface between the motor and wheels (a transmission and/or differential), as is illustrated in Fig. 1.1.

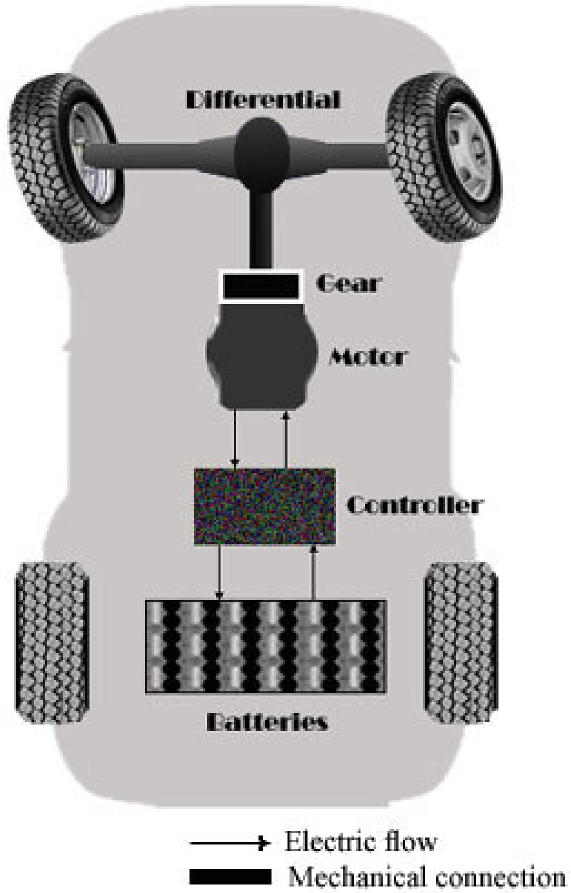


Figure 1.1: Electric vehicle drive train

Batteries store the electric energy to operate an electric motor, which, in turn, changes electrical energy to mechanical energy (drive force), which reaches the wheels through a reduction gear, there is no need for a conventional transmission, and an exhaust system is not needed as well. Electric Vehicles, however, have certain limitations and shortcomings. The two major drawbacks are:

1. Limited range and long charging time: electric car can be plugged into a regular outlet to charge the batteries, a process that would take several hours. In addition, conventional batteries do not last long and they need to be recharged after a short trip. This would limit the use of such cars in urban areas.

   Some electric cars run on fuel cells, which use hydrogen and oxygen to generate electricity to power the traction motor [5]. Instead of batteries, a tank is filled with hydrogen which when reacting with oxygen produces electric current. This clean source of power is not without practical and technical problems. Those kinds of fuel cells are not readily available; just a few stations provide them. In addition, they need a big tank to be stored. Another technological problem lies in fuels, which need about five minutes reaction time before it can be used.

2. The batteries that power the motor are heavy and take up a lot of space; in addition, their life cycle is relatively short.

1.3. Hybrid Electric Vehicles (HEVs)

The drawbacks of electric cars have prevented them from being widely used. As concerns in recent years have grown about global warming cased by carbon-dioxide emissions, in addition, realizing that the earth resources are limited have renewed interest in electrically powered vehicles. In addressing one of the main obstacles to wide acceptance of electric cars, namely their limited range, recent developments have focused on the hybrid vehicle concept.

In a hybrid electric vehicle there are of two sources of traction:

- Electric motor powered by batteries or fuel cells, and
- Internal combustion engine as in of a conventional vehicle.

This combination offers several benefits:

- Rapid refuelling that consumers expect from a conventional vehicle.
- Environmental benefits: a diesel vehicle emits amount of nitrogen oxides and black smoke when the engine is run at a low speed under high load. Hybrid electric vehicles can provide approximately 50 to 70% lower black smoke emission and approximately 20 to 30% lower nitrogen oxides emission than a diesel vehicle [6]. In general hybrids will never be true zero-emission vehicles, however, because of their internal combustion engine, but at least will cut emissions of global-warming pollutants by a third to a half and even more, but it could be zero-emission inside cities [7].
- Hybrid power systems are used to compensate for the shortfall in battery technology. Because batteries could supply only enough energy

for short trips, an onboard generator, powered by an internal combustion engine, could be installed and used for longer trips. In addition, when a hybrid electric vehicle is stopped in traffic, it does not use fuel just to keep the motor running as the conventional vehicles, on the contrary it recharges the battery (regenerative braking), that means that the hybrid uses its electric motor as a generator to produce electricity, and of course this will result in energy savings and reduction in fuel consumption, which in turn reduces the emission.

- Engines can be sized to accommodate average load, not peak load, which reduces the engine's weight.
- Hybrid electric vehicles reduce dependency on fossil fuels because they can run on alternative fuels.
- Conventional internal combustion engines convert the liquid fuel energy into shaft energy. All energy from the combustion process centres around the crankshaft with the exception of that lost in the form of heat. An internal combustion engine vehicle uses approximately 16% of the liquid fuel energy to move the vehicle [7]. The heat emitted in the combustion process wastes the majority of energy while frictional losses from the hundreds of moving parts in the engine; transmissions and the mechanical connections to the drive wheels consume the rest. Such a mechanical arrangement is illustrated in Fig. 1.2 and Fig. 1.3 illustrates the construction of the internal combustion engine itself. Reducing the dependency on internal combustion engine or at least use it just when it is needed will result in saving lost energy in a heat forms. Furthermore, the main source of energy used in hybrid electric vehicles are the batteries. A battery contains no moving parts and this result in decreasing the wasted energy in heat forms, which is produced from the moving parts.

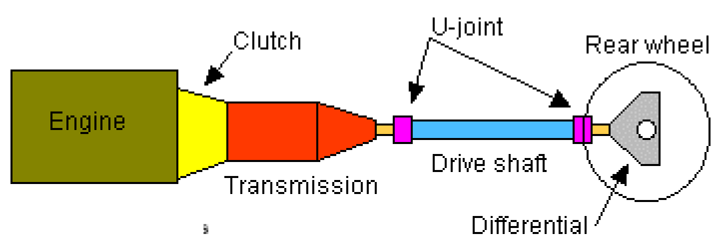


Figure 1.2: Mechanical arrangements between engine and wheel.

- Energy storage: as is mentioned earlier, fuel cells (an electrochemical device in which a fuel reacts with oxygen to release electrons, producing electricity) eliminate much of the drawbacks associated with conventional batter packs (size, weight, short life cycle, and of course the time of charging). For completeness, another device that can store energy is of course the flywheel (which store energy mechanically). Flywheels store kinetic energy within rapidly spinning wheel like rotor or disk. Ultimately, flywheels could store amounts of energy comparable to batteries [10,11]. They contain no acids or other potentially hazardous materials. They are not affected by temperature extremes, as most batteries are. While all of today's internal combustion engines use flywheels to store energy and deliver a smooth flow of power from abrupt power pulses of the engine, flywheels are still very complex and heavy. In addition, there are some concerns regarding the safety of that spins mass at high speed.

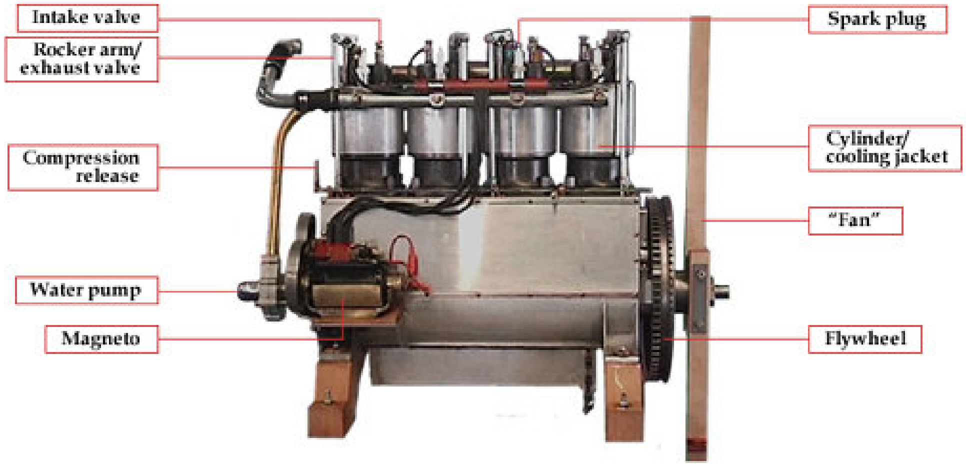


Figure 1.3: Conventional internal combustion engine

1.3.1. Hybrid Vehicle - Series Type

In a series hybrid vehicle, the electric motor/s is/are only connected directly to wheels, engine-generator set is only operated to generate electric power, which is stored in a battery, which, in turn, feeds the electric motor. The vehicle is driven by an electric motor run by electricity from the battery, as shown in Fig. 1.4.

The engine provides the battery with electricity until the battery becomes fully recharged (upper limit), then the engine will shut off. Similarly when the battery reaches to the lower limit the engine starts again to feed it, and so forth. Intelligent power electronics (controller) decide when to use the motor and engine and when to store electricity in advance in the batteries for future use. Fig. 1.5 illustrates the energy flow in a series hybrid electric vehicle.

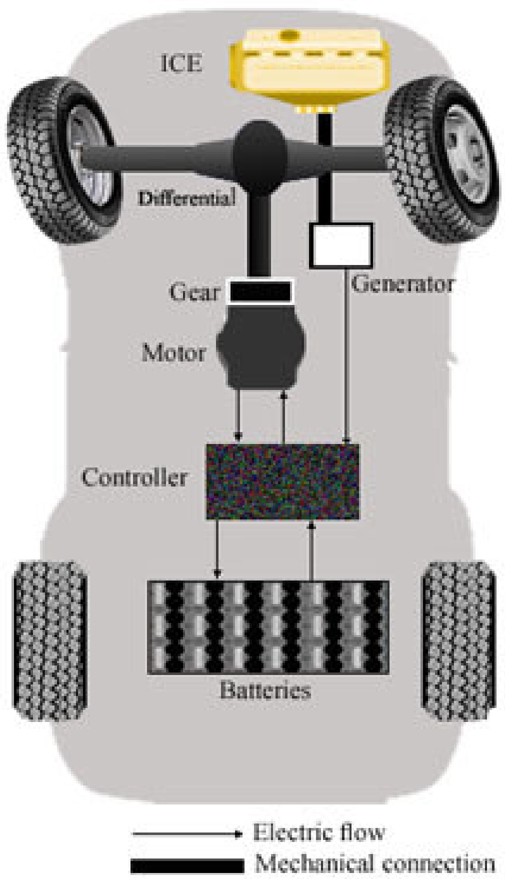


Figure 1.4: Series HEV

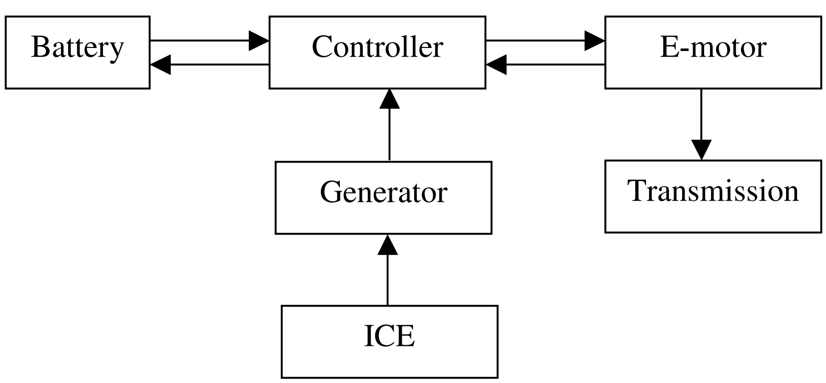


Figure. 1.5: Energy flow in a series HEV

In some series hybrid electric vehicles more than one electric machine are used. Fig. 1.6 illustrates a hybrid electric vehicle with two electric machines, one can be used to run the vehicle (motoring), while the other is charging a battery (generator). Both of them can operate as motors to provide traction when it necessary depending on the driving conditions.

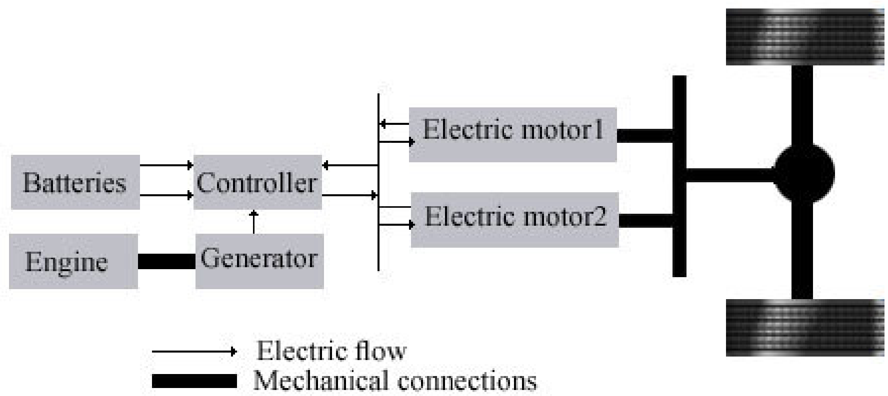


Figure 1.6: A series HEV employing two electric motors

Electric motors revert to generating mode and are used to recharge batteries as well, when a hybrid electric vehicle is stopped in traffic. In other words, breaking energy is converted to electrical energy and stored in the batteries.

In some series hybrid electric vehicles, electric motor obtains electric power from both the battery and the internal combustion engine; this will help the vehicles to get more power, for overtaking or climbing a hill.

Since only the electric motor provide traction at the wheels, the engine can run at optimum performance greatly reducing emission. This enables the internal combustion engine that drives the alternator to run at an almost constant speed and so can be optimized for fuel consumption and exhaust emission.

1.3.2. Hybrid Vehicle - Parallel type

In a parallel hybrid vehicle, the wheels are driven directly by two motive power sources; electric motor(s) and an internal combustion engine. The electric motor is suitable in town and the engine for longer journeys. Both of them can be used simultaneously to get extra acceleration when required. Parallel vehicles do not need a dedicated generator since the electric motor is used as generator to recharge the batteries, as shown in Fig. 1.7.

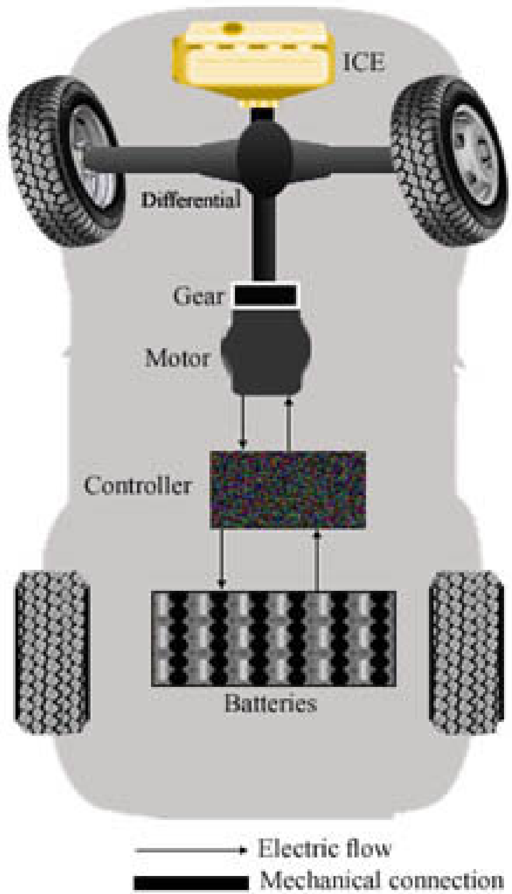


Figure 1.7: HEV of the parallel type

They are two types of parallel hybrid vehicles:

- One of motive power source is connected to the front wheels and the other is connected to the rear wheels.
- The two motive power sources are connected to the same wheels.

Both types work in the same way, as it shown in Fig. 1.8.

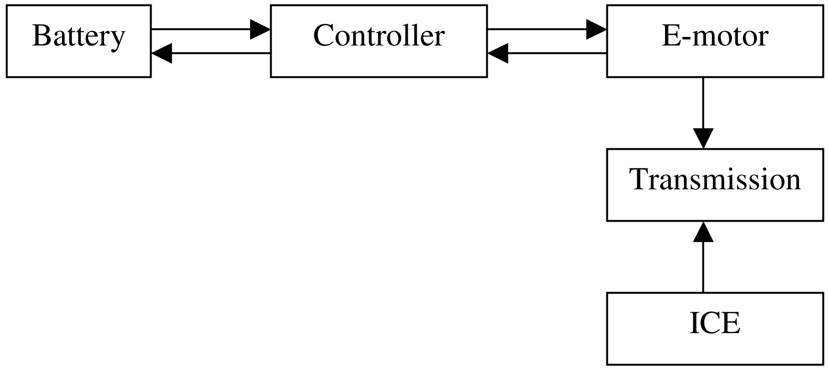


Figure 1.8: Energy flow in a parallel HEV

1.3.3. Hybrid Vehicle – Dual Mode

As the name implies, mixed hybrids are a blend of the two configurations described above; this type combines the best aspects of both (series and parallel). This arrangement and associated energy flow diagram are illustrated in Figs. 1.9 and 1.10; respectively.

Dual-mode hybrid vehicles can be driven directly by an electric motor or internal combustion engine, and from the both in some conditions to provide the vehicle with more power. Internal combustion engine is used to charge batteries through a generator, as the electric motor can be used as generator to recharge batteries (regenerative braking).

The energy flow diagram is illustrated in Fig. 1.10.

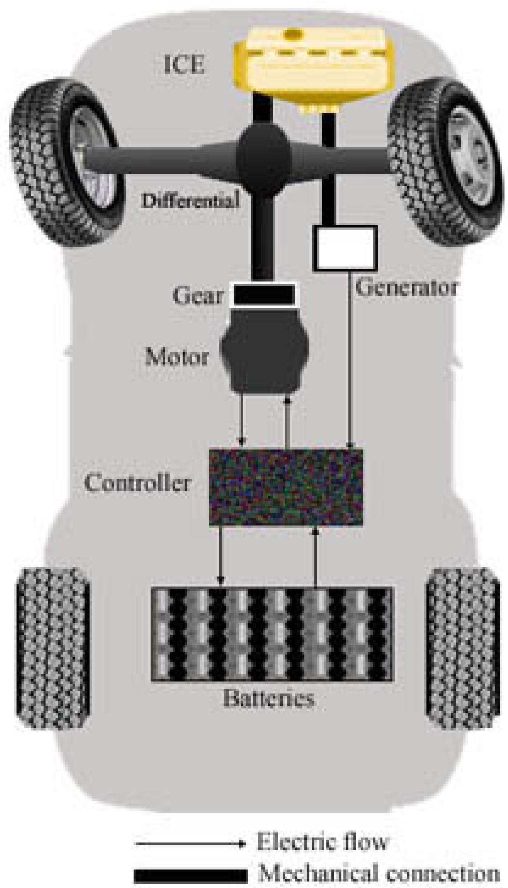


Figure 1.9: HEV of the dual-mode

For completeness, a general comparison between conventional and hybrid vehicles is summarised in Table 1.1 [12,13].

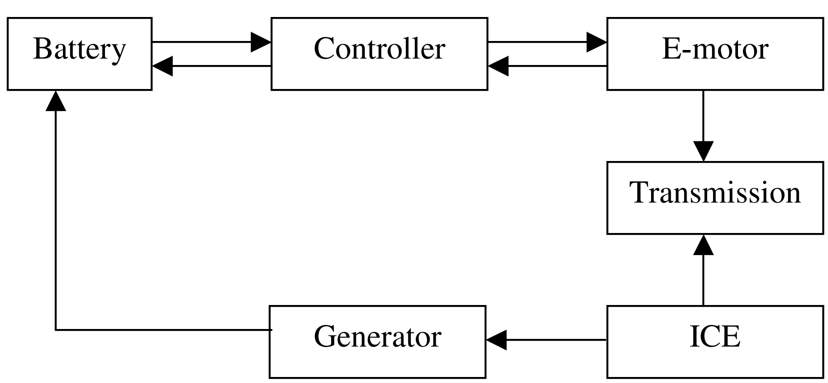


Figure 1.10: Energy flow in a dual mode HEV

	Conventional Vehicles	Hybrid Vehicles
Power resource	Engine	Engine & electric motor
Fuel used	100%	50%
Engine losses	60 to 70 %	25 to 30%
Idling losses	10%	0%
Regenerative Braking	0%	+4% and more
Engine size	Big	Relatively small
Engine weight	Heavy	More lighter
Nitrogen oxides	High	20 t0 30% lower
Black smoke emission	High	70% lower
Environmental harm	High	Less (more friendly)
Transportation cost	High	Less
System complicated	Normal	More complicated

Table 1.1: Comparison between conventional and HEVs.

1.4. Scope of the Thesis

The main aim of the work is to design a searchable engineering tool to assist selection and design of electric traction drives in HEV. The search input to the database should be such properties like torque, power, efficiency and speed range. The output of the database should, for a specific motor type be size, number of poles, field weakening range, efficiency etc. The purpose of this database is to support the vehicle designer in the selection of a suitable traction motor type, not to give the entire traction motor design details, which takes much more work.

This thesis consists of four chapters. Following the introduction above, chapter II provides a general view of the developed software. It defines the input data that need to be provided by the user, describes the main procedures followed at each stage of the program and the output information provided. Also included is the design know-how that has been programmed in the user-friendly package developed.

Chapter III focuses on IT aspects of the developed software. It describes programming language (user interfaces), the storage media and the data search technique employed. Also briefly described are various possible interfaces that would allow exporting results to other design and analysis tools. The conclusions and recommendations for further work are given in chapter IV.

CHAPTER II

General Overview of the Developed Software

2.1. INTRODUCTION

The work is concerned with an engineering tool to assist selection and design of traction drives in HEVs. The search input to the CAD package would be drive requirements such as torque, power, efficiency and speed range. The output should be a recommendation for a specific motor type, size, number of poles, and field weakening range. The developed software therefore consists of three main parts as illustrated in Fig. 2.1.

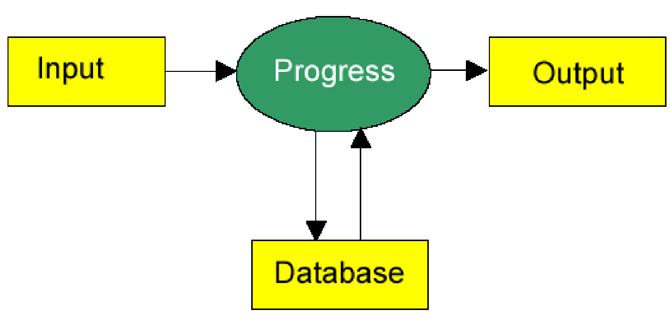


Figure 2.1: Illustration of the developed software

The flowchart of the Fig. 2.2 illustrates the functional organisation of the interactive CAD facility.

The first step in the development is to decide on the programming language and data storage and search tools. Visual Basic is a programming language concept in which a program is viewed as collection of discrete objects that are self-contained collections of data structures and routines that interact with other objects. It is an Object-Oriented Programming (OOP) Language in which a variable comprising both routines and data is treated as a discrete entity. As Visual Basic emulates the human way of seeing things, it is therefore adopted in programming the user-friendly interface as well as to perform computational steps within the software.

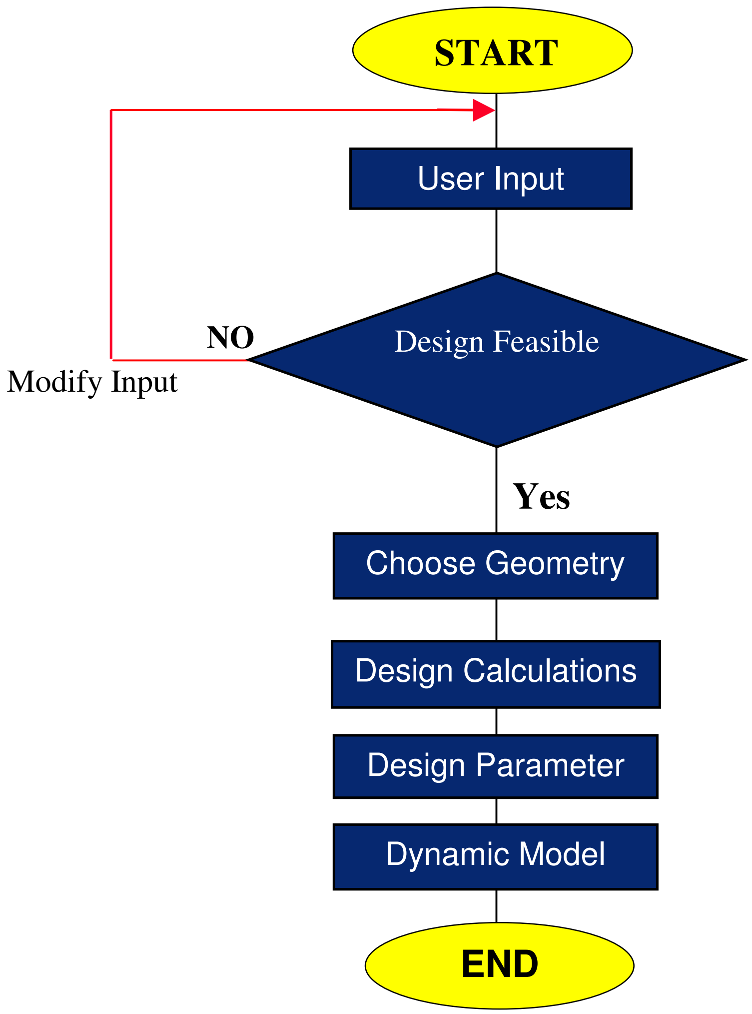


Figure 2.2: Organisation of CAD package

Structured Query Language (SQL) can perform the search operation for specific information within a huge database in a fraction of a second, regardless the database type. SQL can also be embedded within a Visual Basic medium. As the developed software makes use of expert drive system design knowledge, which is stored in the form of a database, SQL is utilised to perform the data search. The data itself is stored using Access, which contains details of hundreds of standard laminations as well as decision making criteria and detailed design characteristics and equations.

The software developed here can be divided into three main parts, as shown in Fig. 2.3 If we take into account that VB, SQL, and Access, all of these products are fully supported by Microsoft, as indeed is the target operating system (Windows), and this should eliminates any incompatibility problems. The various layers of the software and stages of computation are now described in detail. Also given are the engineering design methods that are programmed in the various modules.

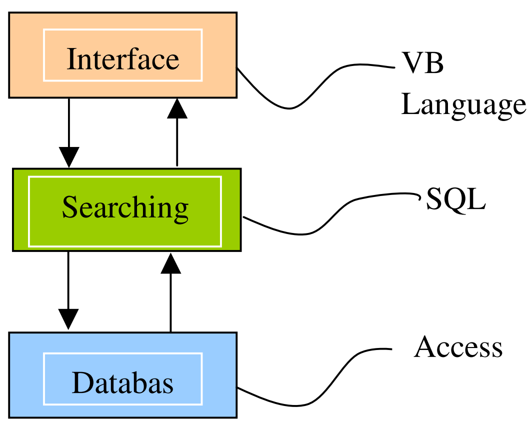


Figure 2.3: Integration of VB, SQL and Access in the developed software

2.2. First Stage of the Computation - Design Feasibility

The first stage of computation aims to give the user a quick answer to whether or not the drive requirement can be met within a specified space envelope.

2.2.1. Requirements

The start up menu (user defined parameters) requires the inputs shown in Fig. 2.4 to be provided by the user. Not all the specified parameters are used in every stage of computation. For example, in the initial feasibility study, only the space envelope, base-speed, continuous torque and method of cooling are used.

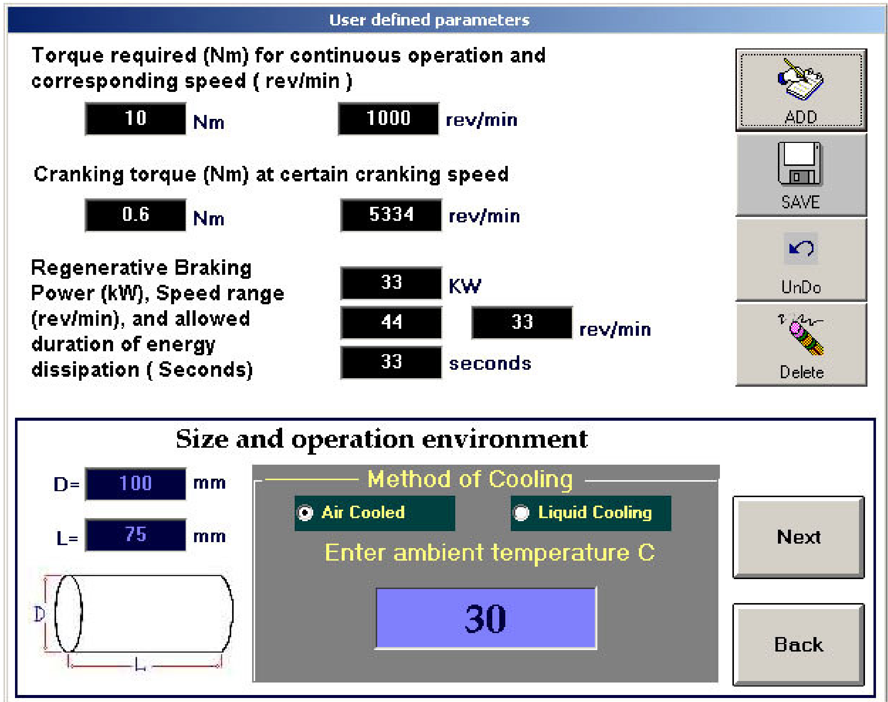


Figure 2.4: Inputting of user defined parameters

The computation utilizes the specified drive base-speed (N, rev/min) to determine a preliminary number of poles, which is, together with the diameter of envelope available (D, mm) are used to calculate the pole-pitch (τ, mm) as follows:

$$\tau = \frac{0.56\pi D}{p} \tag{1}$$

As is seen in the equation (1), the air-gap diameter (Fig. 2.5) has been assumed to be 0.56 of the user defined envelope diameter (to allow for stator slot or magnet and core depths and thickness of housing frame). This value is based on experience and can easily be verified by examining laminations catalogues [23].

The number of poles (p), in equation (1) is estimated from the speed, as shown in the table 2.1.

Speed (N, rev/min)	Poles (P)
250 ≤N< 500	8
500 ≤N< 1000	6
1000 ≤N< 1500	4
1500 ≤N	2

Table 2.1: Synchronous speeds and poles based for 50 Hz frequencies.

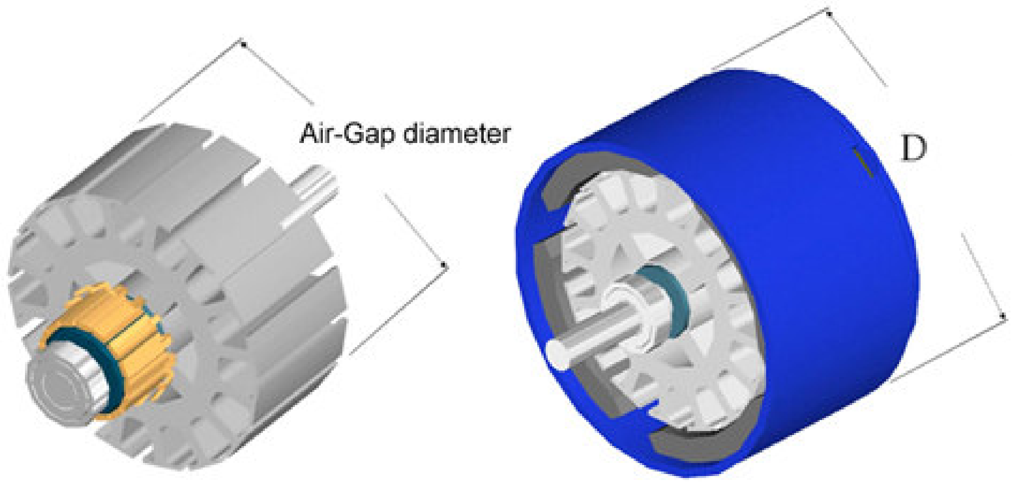


Figure 2.5: Air-gap diameter and outer diameter (*D*)

In the database, it is stored sets of normalised available force density curves for different pole-pitch values (details are given in Appendix III), obtained according to the work of Kasinathan [3]. The calculated pole-pitch (τ) points to the appropriate data set in the database, as shown in Figs 2.6 and 2.7.

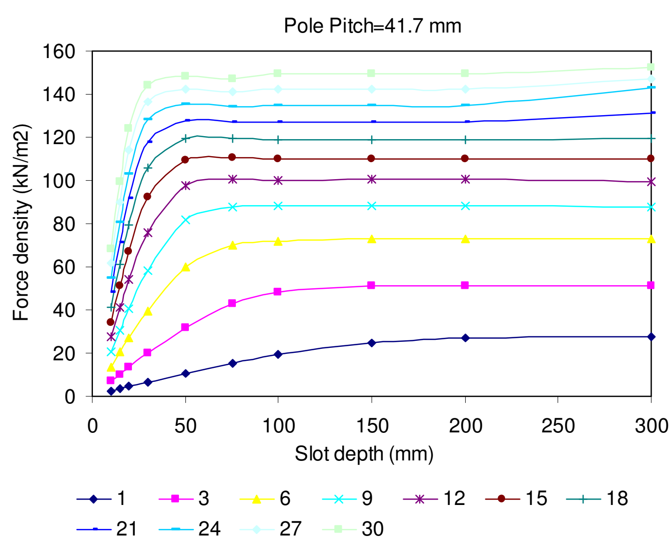


Figure 2.6: Typical force values for different winding current densities

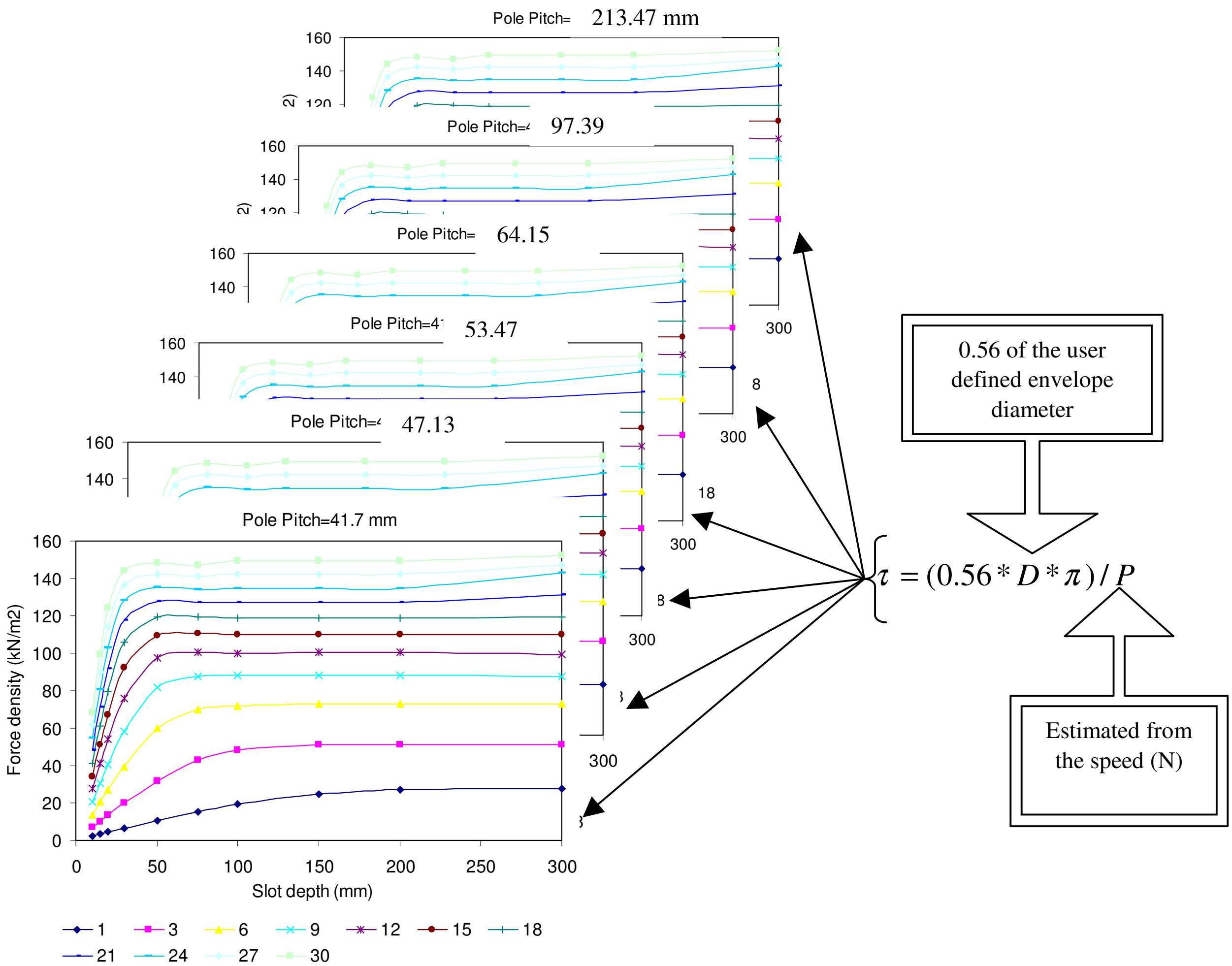


Figure 2.7: Calculated pole-pitch (τ) points to the appropriate data set in the database.

2.2.2. Method of Cooling

The specified method of cooling is used to decide winding current density. This value is, in turn, used to determine which particular curve in the chosen family is to be used to estimate the force density limits. Table 2.2 shows the used relationships between method of cooling, ambient or coolant temperature, and winding current density.

For programming purpose, each force density curves (Fig. 2.6) is divided into two main regions. The first shows an increase of obtainable force density as the slot depth (and machine size) is increased (and is approximated by a straight-line equation on the form $y = mx$).

Method Of Cooling	Temperature (T) C	Winding Current Density A/mm
Air Cooled	T < 10	12
	10 ≤ T < 25	9
	25 ≤ T < 40	6
	40 ≤ T	3
Liquid Cooling		21

Table 2.2: Choice of current density according to method of cooling [1].

The second Region (saturated part of the curve) indicates no farther increase of force density is obtainable. This region is approximated by a straight-line equation on the form $y = C$. This is illustrated in Fig. 2.8.

2.2.3. Required Force Density

The theoretical maximum value of the force density available is determined by the intersection of the two regions. This value is denoted theoretical limit (TL). The practical limit (PL) and balanced practical design (BPD) points are determined from design experience as half and 25% of the theoretical limit [1]. The second Region (saturated part of the curve) indicates no farther increase of force density is obtainable, as shown in Fig. 2.8.

Computation utilises the continuous torque (T, Nm) as specified by the user, the assumed initial air-gap diameter and available length of the space envelope (L, mm) to estimate a value of the force density required (kN/m^2).
As it shown in the Fig. 2.8, the part of the winding current density curve has been approximated to linear, due to programming purpose, from this point the force density can be calculated as follows:

$$y = mx \Rightarrow x = y / m \tag{2}$$

where: $m = y_{max} / x_{max}$: $y_{\max}, x_{\max}$ are stored in the database (for each individual curve and for a range of pole-pitch values).

$$y = T / (D^2 * L) \tag{3}$$

where D and L are given by the user.

After that x will be calculated and, projecting its value the linear approximation of the force density curve yields the force density obtainable.

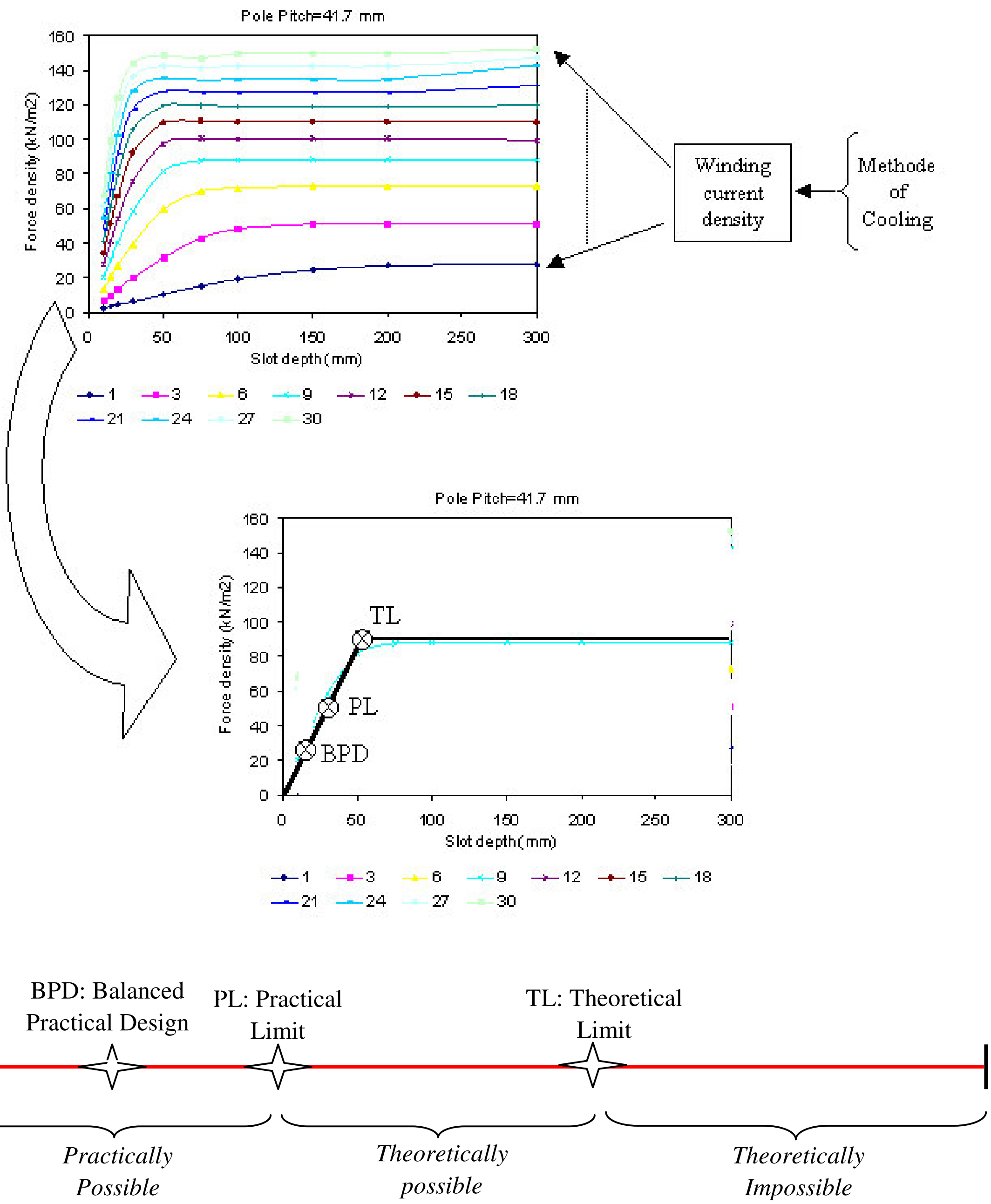


Figure 2.8: Illustration of the computer implementation of force density computation.

2.2.4. Feasibility of the Design

After calculating the force density, the user will be provided with an indication regarding the theoretical and/or practical feasibility of specified requirements, as shown in Fig. 2.9. In cases when the required force density exceeds the theoretical limits, the user is prompted to significantly reduce the requirements or increase the space envelope specified. An indication as to how far away the requirements are from the theoretical limit is also provided.

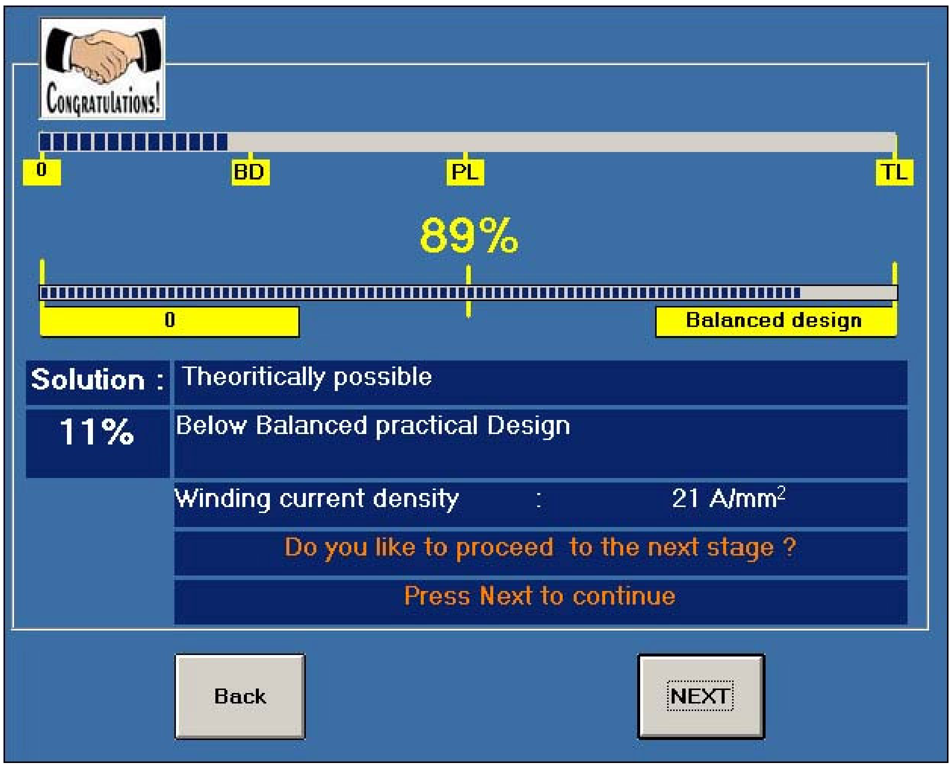


Figure 2.9: Screen shot of outcome of the feasibility study

Requirements above the practical limit (but below the theoretical limit) would indicate that the design may be possible, but it would be either too expensive or having inferior dynamic performance. A balanced practical design in one that is, from experience, is deemed to possible to manufacture economically, and is expected to have acceptable performance.

When the requirements can be accommodated in a smaller space envelope, a message to that effect appears on the screen. This would enable the vehicle designer to either increase the electrical traction requirements or to utilise the spare space in accommodating other equipment.

2.3. SECOND STAGE OF THE COMPUTATION STEADY STATE PERFORMANCE

When the traction requirements are deemed to be compatible with the available space envelope and specified cooling method, the user can proceed to select a particular drive topology, and to perform detailed design and dimensioning calculations that are unique to the chosen topology. In the current software, four virtual slots are provided into which different topology design method modules can be added, as illustrated in Fig. 2.10.

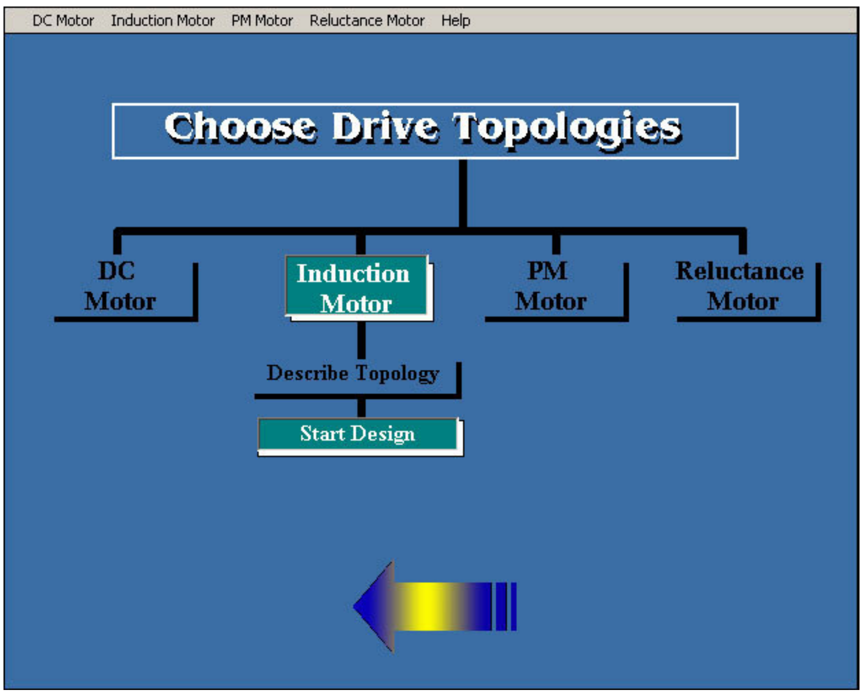


Figure 2.10: Screen shot of the topology selection menu

At present, only the module dealing with the design of induction motors is fully developed. The methodology, which is programmed in the developed software, is described below. First, a brief description of this type of machines is give.

2.3.1. Induction Motors

Induction machines are usually operated as motors and they provide the most common form of drive for industrial applications. The popularity of the induction motor stems from its simple construction which makes it less expensive than other drive topologies of the same rating. Also, induction motors require practically no maintenance. Traditionally, induction motors were used in applications requiring constant speed operation. Recently, with advances in the area of electronic power conversion, induction motors have become widely used in variable-speed drives for various industrial and processing applications.

The main disadvantages of induction motors are their high starting current and their low power factor at light loads. As their use in HEV would be in conjunction with power conditioning units, these drawbacks can be treated in the system design.

Construction:

The operation of an induction motor requires two windings, a primary, which is excited by an alternating current source, and a secondary, which is short-circuited. In the conventional design, the primary winding is housed on the stator while the rotor carries the secondary winding, but this not essential to the concept.

The primary always utilises a distributed winding, wound to give the required number of poles and phases. There are two types of secondary windings giving rise to two types of induction motors; the slip-ring (or the wound-rotor) and the squirrel-cage.

The rotor of the slip-ring motor carries an insulated poly-phase winding, wound to give the same number of poles as the stator winding. The winding leads are connected to slip-rings mounted on the shaft. Carbon brushes riding on the slip-rings allow insertion of external resistances into the rotor circuit to improve starting characteristics. During normal operation the slip-rings are shorted together. Generally, wound-rotors are only used in large motors and, due to their maintenance requirements, they are not considered here to be suitable candidates for HEV traction.

The squirrel-cage winding consists of solid bars of copper or aluminium joined at each end by an end-ring of the same material. Rotors of small machines usually utilise die-cast aluminium bars and end-rings. Fig. 2.11 shows a cutaway view of a three-phase squirrel cage induction motor. This construction is practically maintenance-free. Therefore, the squirrel-cage design is the one that is considered here to be suitable for HEV applications.

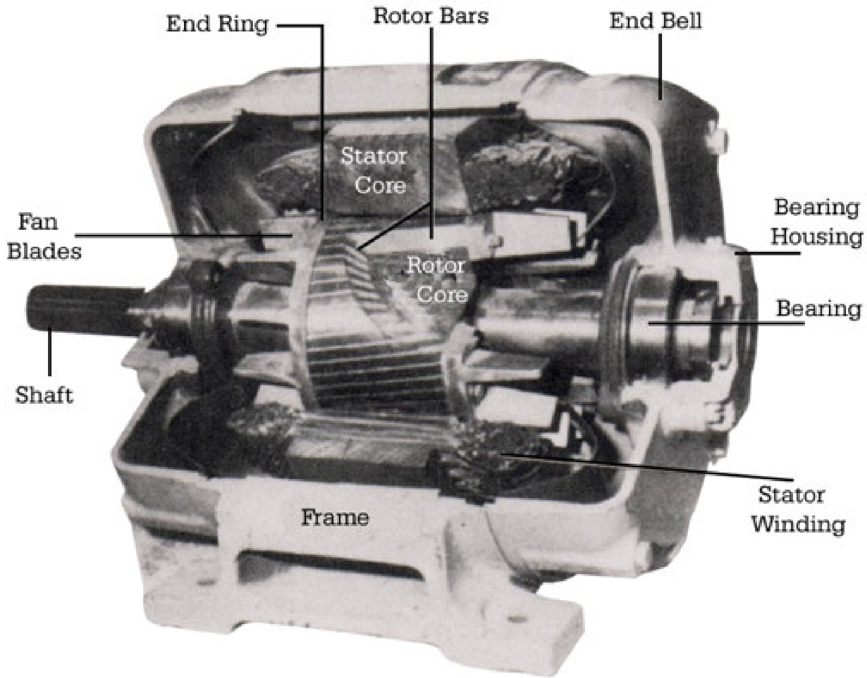


Figure 2.11: Cutaway view of a squirrel-cage induction motor.

Both the stator and the rotor of an induction motor utilise slotted laminations. The laminations used in industrial motors are either 0.35 or 0.5 mm thick non-oriented low-loss electrical steel. The stator and rotor of domestic appliances fractional-horsepower motors are generally built from silicon-free materials and the lamination are usually 0.65 mm thick.

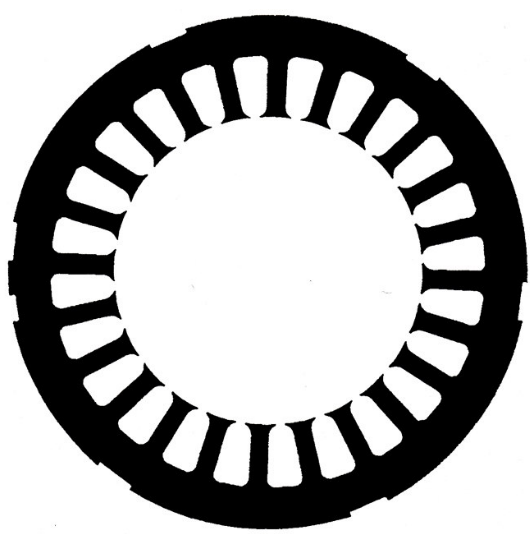


Figure 2.12: Partly closed tapered slots used for small induction motors.

Motors having an outside diameter of the stator core of up to 1 metre use one-piece core laminations. This would cover the size range for use in electric and HEV. The stator and rotor are punched from the same sheet and there is, practically, no material wasted. For small motors, partly closed slots are used and the teeth, not the slots, have parallel sides, as illustrated in Fig. 2.12.

The stator laminations are assembled into a core pack. The stack is pressed and the laminations are then either welded or riveted together. In the case of welded stator core, the weld is applied directly to the laminations in groves specially punched for this purpose. Fig. 2.13 shows a riveted stator core assembly.

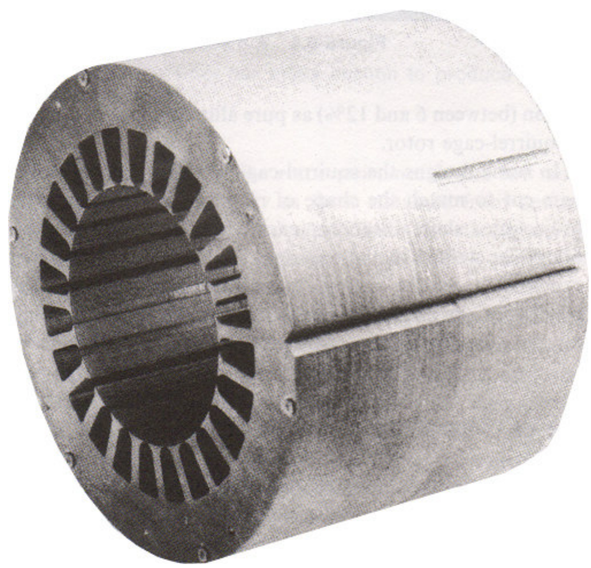


Figure 2.13: A riveted stator core assembly

Rotor laminations of wound-rotor motors are usually assembled directly on the shaft, keyed to it and pressed together with the aid of washers or thrust rings.

In the case of squirrel-cage rotors, the laminations are stacked in a core mould into which molten aluminium is forced under pressure. With this production method, the rotor bars, end-rings, and cooling fins are cast at the same time and the rotor is manufactured in a single process. In practice, the aluminium is alloyed with silicon (between 6 and 12%) as pure aluminium does not cast well. Fig. 2.14 shows a squirrel-cage rotor.

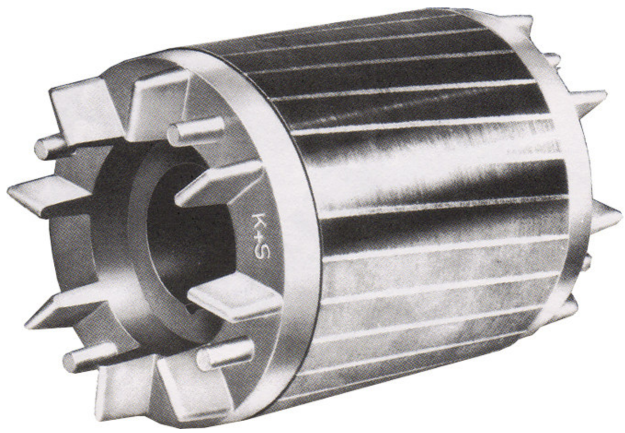


Figure 2.14: A die-cast squirrel-cage rotor

In some designs the squirrel-cage winding utilises round or rectangular copper bars cut to match the shape of rotor slots. The bars are driven into rotor slots, projecting a short distance on each end of the core. The bar ends are then soldered brazed or welded to the end-rings. This manufacturing method is obviously more expensive than die-casting. It may be noted that die-casting of a copper cage winding is possible, but it requires a much higher casting temperature than that required for aluminium casting.

2.3.2. Design Calculation

This procedure follows conventional design procedure [1,2]. The various symbols used in this section are as follows:

D	Armature diameter or stator bore, m
L	Stator core length, m
N_s	Synchronous speed, rev/s
p	Number of poles
τ	Pole-pitch, m
K_w	Winding factor
f	Supply frequency
Q	Machine rating, kVA
Co	Output coefficient
μ_0	Permeability of the free space or (air) = $4\pi x10^{-7}$ H/m
λ_s	Slot permeance coefficient

Table 2.3: Symbols and definitions

In case of induction motors, values of power factor and efficiency, magnetic and electric loadings are determined according to rated steady state output power and cooling method to be employed. The machine's kVA input is then calculated and air-gap D^2L is determined from a standard output equation [1]. Suitable choice of the ratio of stack length/pole-pitch (to control end-winding leakage) enables separation of the motors air-gap diameter and length. Using the air-gap diameter, a standard lamination database is searched for the nearest available one.

The design procedure begins by estimating the value of the power factor ($cos\ \phi$) expected full-load efficiency (η), and specific electric (ac Ampere-conductor/m) and magnetic (B_{av}, T) loadings from data such as that shown in Table 2.4.

The rated output power (P_o, kW) of the machine is calculated as follows:

$$P_o = T\,\omega_m \tag{4}$$

where T is the shaft torque in Nm and ω_m is the mechanical speed in rad/s.

The data of Table 2.4 is programmed in the software developed here. The torque and speed values provided by the user are used to calculate the output power. A number of decision-making steps (programmed in VB) yield appropriate values of all the variables given in Table 2.4. These values are then stored in specified memory locations for further use in the design work.

kW	Efficiency	Power Factor	Electric Loading	Magnetic Loading
$0 < kW \leq 2$	0.7	0.7	22	0.5
$2 < kW \leq 15$	0.85	0.8	26	0.5
$15 < kW \leq 40$	0.9	0.9	28	0.6
$40 < kW$	0.95	0.9	30	0.65

Table 2.4: Efficiency, power factor and specific loadings of induction machines [1].

The motor's kVA (Q) input is related to the motor's main dimensions as follows:

$$Q = C_o D^2 L n_s \tag{5}$$

$$C_o = 1.11\pi^2 B_{av} K_w \times 10^{-3} \tag{6}$$

where,

where D and L are the air-gap diameter and core length, m; respectively, K_w is the winding factor and n_s is the synchronous speed, rev/s.

The kVA input (Q) is determined in terms of the rated output power (P_o), efficiency (η) and power factor as follows:

$$Q\,(kVA) = \frac{P_o\,(kW)}{\eta \cos\phi} \tag{7}$$

The winding factor (K_w) is:

$$K_w = K_d\, K_c \tag{8}$$

Where K_d and K_c are the primary winding distribution and the chording (or pitch) factor, respectively.

Initially, the distribution factor may be taken as 0.955 for three-phase motors (corresponding to 3 slots per pole per phase) and the pitch factor may be assumed unity. The winding factor can be determined when the stator (primary) winding is designed but this is not programmed in the developed software.

After calculating the D^2L product it becomes necessary to split it and to find the motor's main dimension, namely D and L. This is achieved by selecting a suitable value for the ratio L/τ.

While designing with the largest possible diameter would, in general, yield high specific output, the induction motor design must also have an acceptable power factor. This is important because low power factors would increase the size and cost of the associated inverter. The magnetising current of an induction motor decreases (and the power factor improves) as the diameter is increased, but it increases as the number of poles is increased. L/τ ratio between 1 and 1.25 gives a good power factor and a value between 1.5 and 2 yields minimum cost [1,2]. Here L/τ is taken as 1.5 and this is regarded as a balanced choice for HEV applications.

The steps described by equations 5-8 above are programmed and the computations are invisible to the user. The *D* and *L* can now be calculated and the user is provided with this information in the form of the screen shot shown in Fig. 2.15.

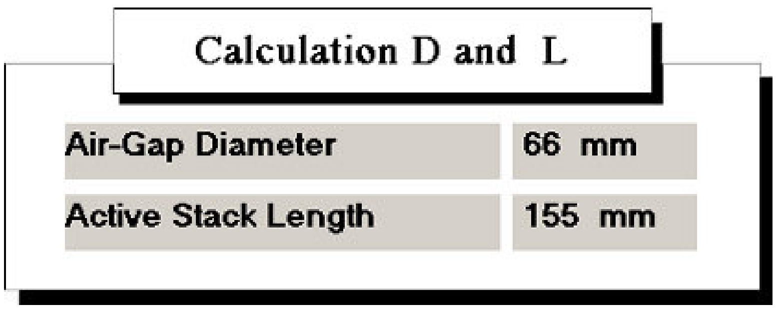


Figure 2.14: Calculation of the motor's main dimensions.

2.3.3. Searching for Matching Standard Laminations

In the database of the developed software it is stored hundreds of laminations as specified by IEC standards. After calculating the air-gap diameter values, search is performed to see if the requirement can be met by standard laminations. SQL language is used for the searching technique to find the lamination(s) matches that matches the calculated air-gap diameter (*D*) inside the database. An example of the SQL search statement is as follows:

DataBase.RecordSource = "select StandaredLamination where Di =" + D

where D: Calculated air-gap diameter, and
Di: Air-gap diameter value for the stored laminations.

This subject is described in details in the following chapter.

Fig. 2.16 shows a message, in case of matching lamination are found in the database.

Figure 2.16: Matching lamination found

When the required lamination stack is outside the stored database, the user is given the option to draw the lamination as it shown in Fig. 2.17.

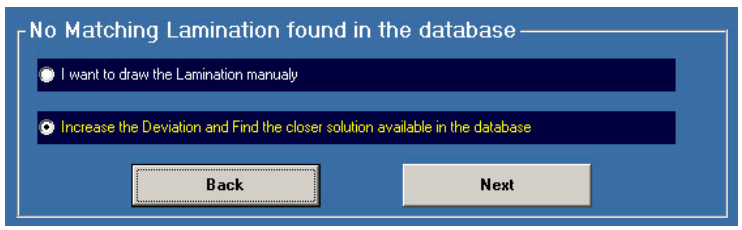


Figure 2.17: Option to draw lamination

When clicking *OK* in the lamination search results window of Fig. 2.16, the full details of the selected lamination(s) is (are) displayed as shown in Fig. 2.18. It is shown that more than one lamination that satisfies the air-gap diameter requirements are found, in this case. The user can select individual lamination to examine the number of slots and their area.

With reference to Fig. 2.18, details of stator and rotor slots are shown where the various variables are as defined in Table 2.5.

Da	Outer diameter of stator lamination	mm
Di	Inner diameter of stator lamination	mm
N	Number of slots	
A	Slots area	mm^2
Bz	Tooth width	mm
da	Rotor outer diameter	mm
di	Shaft diameter	mm

Table 2.5: Lamination details

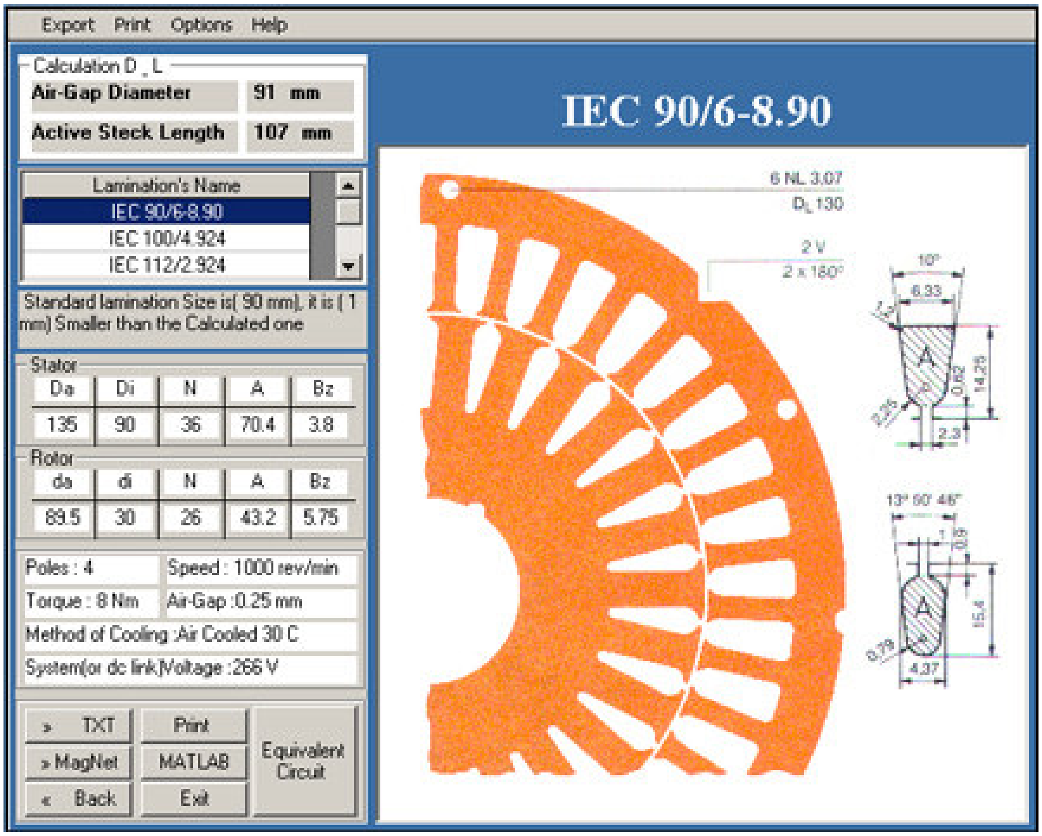


Figure 2.18: Selecting Standard lamination from those stored in the database

The air-gap value given in Fig. 2.18 is directly calculated from Di and da. However, under *Options* of the main window menu the user can change the air-gap value. If the machine is required to have a large air-gap than specified, because of the drive train requirements, this has to be provided as an input at this stage. This is because the air-gap value is used in calculating the magnetizing inductance of the machine in the next stage of computation.

Other available options, as shown in Fig. 2.18, are: *Back*, *Print*, *Exit*, *Equivalent Circuit*, *MagNet*, *TXT*, and *MATLAB*. The first three are self-explanatory; the other options are now explained.

2.3.4. Equivalent Circuit Option

Following selection of stator and rotor laminations, suitable windings are designed and the values of resistances and inductances are calculated. These may be used in the dynamic modeling, as is described below, or in deriving the steady-state equivalent circuit.

2.3.4.1. Stator Winding

The stator winding may be either of the single layer type (with one coil side per slot), or of the double layer type. Single layer much windings are the most commonly used type for small induction motors. Single layers windings of the concentric type and double layer windings are used for some industrial motors.

In the case of the mush (or basket) windings, all the coils have the same span and the coil ends belonging to coils situated in consecutive slots cross one another, i.e. they proceed to left and right alternately. There is only one coil group per phase per pole-pair and, therefore, the maximum number of parallel circuits is equal to half the number of poles. The coil span, in terms of number of slots, of mush windings must be odd for three-phase motors and even for two-phase motors.

A sine wave flux distribution may be assumed for induction motors, as the distributed stator winding produces an air-gap flux waveform, which is very nearly sinusoidal. The stator winding induced *emf* is given by:

$$E_s = 4.44\, f\, \phi T_s\, K_{ws} \tag{9}$$

where E_s is the voltage induced in the stator (primary) winding per phase, V, f is the supply frequency, Hz, ϕ is the flux per pole, Wb, T_s is the number of stator winding turns in series per phase and K_{ws} is the stator winding factor.

The induced voltage per phase may be taken equal to the terminal phase voltage minus the product of the magnetizing current and stator leakage. Initially, E_s is taken here as 0.97 times the terminal phase voltage [1,2].

The flux per pole is determined in terms of the average flux density in the air-gap (B_{av}), pole pitch (τ) and the net core iron length (L_i) as follows:

$$\phi = B_{av}\, \tau\, L_i \tag{10}$$

where ϕ is flux per pole in Weber, τ and L_i are in metres and B_{av} is in Tesla.

The number of stator turns per phase is:

$$T_s = \frac{E_s}{4.44\, f\, \phi\, K_{ws}} \tag{11}$$

The total number of conductors in the stator winding is $6T_s$ for three-phase motors.

The stator phase current is obtained as follows:

$$I_s = \frac{P_o 10^3}{3V_{ph} \cos\phi\, \eta} \tag{12}$$

The conductor current, I_{zs}, is obtained by dividing the motor phase current by the number of parallel armature circuits. The copper losses in any winding vary directly with the squared value of its current density. The temperature rise depends on the losses for a given construction. The stator current density must be so chosen that the required efficiency is obtained without exceeding the temperature rise limits. The current density in the stator winding of a standard air-cooled induction motor may be assumed between 3 and 6 A/mm^2; depending on the duty type and the ventilation method employed. In this work, the current density value is assumed according to the method of cooling specified by the user. The cross section area of the stator conductor is:

$$a_s = \frac{I_{zs}}{\delta_s} \quad mm^2 \tag{13}$$

Where δ_s is the current density in the stator conductors, A/mm^2.

Generally, the requirements of small induction motors are met by round conductors. For a single-layer winding the approximate length of the mean turn (L_{mts}) may be determined from the following empirical formula [1,2]:

$$L_{mts} = 2L + 2.3\,\tau \tag{14}$$

where the core length L and the pole pitch τ are in metres.

For a double-layer winding the length of the mean turn will be between 10 and 25 mm greater than the value given in equation (14).

When the number of conductors per slot and the conductor cross section area are known, the copper area per slot can be compared with the slot area to ensure that the winding can be accommodated in the slot.

Small induction motors utilise semi-closed slots; the slot insulation is inserted into the slots, the stator coils are automatically wound and then placed in the slot. This manufacturing method results in a typical value of slot space factor of about 0.4, generally not higher than 0.5. Therefore:

$$A_{ss} = \frac{Z_{ss}\, a_s}{0.4} \tag{15}$$

Where Z_{ss} is the number of conductors per slot. The stator slot area A_{ss} and the conductor cross-section area a_s are in millimetre squared.

Deep slots result in higher slot leakage and, generally, the ratio of slot depth to width should be between 3 and 6. However, the dimensions of the slot determine the flux density in the teeth. The slot should be so proportioned that resulting maximum tooth flux density is about 1.7 T.

The stator resistance per phase is:

$$r_s = \frac{\rho T_s\, L_{mts}}{a_s\, 10^{-6}} \tag{16}$$

Where a_s is in mm^2 and ρ is in Ω m.

Rotor Bars and End-ring

According to [1], the total rotor ampere-turns may be assumed between 0.85 and 0.9 times the total stator ampere-turns. No insulation is used between the cage bars and rotor core, and the rotor may be worked at a much higher current density than the stator. Additionally, the mean length of rotor turn is shorter than the stator and ventilation is better. The current density in the rotor bars may be taken no greater than twice that in the stator winding. Thus, the ratio of the total rotor conductor section to the total stator copper section is, approximately:

$$\frac{A_{cr}}{A_{cs}} = 0.9 \ x \frac{1}{2} = 0.45 \tag{17}$$

Where A_{cr} and A_{cs} are the total conductor section area for the rotor and stator, respectively.

However, the total rotor conductor area must be selected in relation to the length of the bar and end-ring section so that the proper rotor resistance can be obtained to meet starting torque and running performance requirements. It is generally between 50 and 80 percent of the total stator copper section.

The cross section area of each bar is obtained in terms of the total conductor area in the rotor (A_{cr}) and the number of rotor bars, or rotor slots, (S_r) as follows:

$$a_b = \frac{A_{cr}}{S_r} \tag{18}$$

No insulation is used between bars and rotor core and a clearance of between 0.15 mm and 0.4 mm may be assumed in calculating the rotor slot area.

The total resistance of the squirrel cage bars is:

$$r_b = S_r \frac{\rho L_b}{a_b} \tag{19}$$

Where L_b is the axial bar length.

The distribution of the current in the bars and end-rings of a squirrel cage winding is shown in Fig. 2.19. It is shown that the current in each bar divides in the end-ring, one half returning through a bar, a pole pitch to the right, and the other half through a bar, a pole pitch to the left.

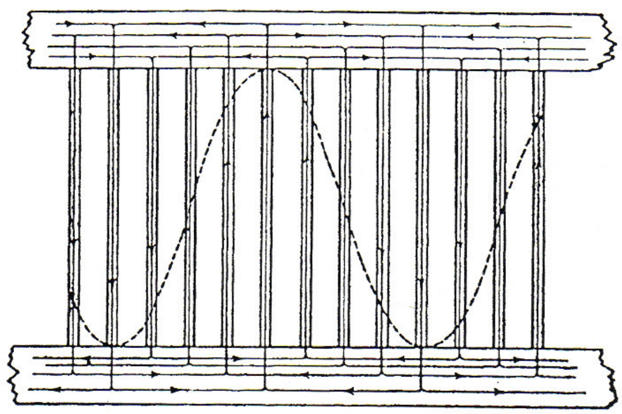


Figure 2.19: Current distribution in a cage winding

It is shown in Fig. 2.19 that at points when the current is maximum in the bars; current is zero in the end-rings, the current in the end-rings is maximum where the bar current is zero. A sinusoidal current distribution is assumed, which is valid if the harmonics in the air-gap field are of negligible magnitude.
The maximum value of end-ring current is obtained as the product of the average bar current and half the number of bars (or rotor slots) per pole. Assuming that the end-ring current, like the bar current, varies sinusoidally, the root mean square value of the end-ring current (I_{er}) becomes:

$$I_{er} = \frac{2}{\pi} \sqrt{2}\, I_b \, x \, \frac{\frac{S_r}{2}}{p} \, x \, \frac{1}{\sqrt{2}} = \frac{I_b S_r}{\pi\, p} \tag{20}$$

Where I_b is the root mean square value of the bar current.

The ventilation is generally better for the end-rings than for the rotor bars, and the current density in the end-ring may be taken equal to or slightly higher than the current density in the bars. However, in most cases the end-ring dimensions are so chosen that its contribution to the rotor winding resistance is minimal.

The cross section area of each end-ring (a_{er}) is:

$$a_{er} = \frac{I_{er}}{\delta_{er}} = \frac{I_b \, S_r}{\pi \; p \, \delta_{er}} \quad mm^2 \tag{21}$$

Where δ_{er} is the current density in the end-ring, A/mm^2.

The total area of the bars section is:

$$A_{cr} = \frac{I_b \, S_r}{\delta_b} \tag{22}$$

Where δ_b is the current density in the rotor bars, A/mm^2.

Substituting the value I_b from equation (22) into equation (21) we obtain the cross section area of each end-ring in terms of the total rotor conductor section (A_{cr}) as follows:

$$a_{er} = \frac{A_{cr}}{\pi \; p} \frac{\delta_b}{\delta_{er}} \tag{23}$$

The resistance of the two end-rings is:

$$r_{er} = 2 \, \frac{\rho \, \pi \; D_{er}}{a_{er}} \tag{24}$$

Where D_{er} is the average diameter of the end-ring.

Utilising equations (19), (20) and (24), the total rotor copper loss can be written on the form:

$$I_b^{\,2} r_b + I_{er}^{\,2} r_{er} = \rho\, I_b^{\,2} S_r^{\,2} \left[\frac{L_b}{S_r a_b} + \frac{2\, D_{er}}{\pi\, p^2\, a_{er}} \right] \tag{25}$$

Dividing equation (25) by the bar current squared we get the total rotor resistance as follows:

$$R_r = \rho\, S_r^2 \left[\frac{L_b}{S_r\, a_b} + \frac{2\, D_{er}}{\pi\, p^2\, a_{er}} \right] \Omega \tag{26}$$

The rotor resistance must be expressed in terms of the stator winding before we can utilise it in the motor's equivalent circuit. At standstill, the induction motor acts like a transformer. The rotor resistance referred to the primary voltage (viewed from the stator terminals) is therefore equal to the total rotor resistance times the square of the ratio of the effective rotor turns to the effective stator turns.

The number of phases of the squirrel cage winding is equal to the number of bars (or rotor slots) per pole, the number of turns in series per phase is equal to the number of pole pairs and the winding factor of the cage winding is equal to 1.0. Therefore, the transformation ratio is:

$$\left[\frac{m_s T_s K_{ws}}{m_r T_r K_{wr}} \right]^2 = \left[\frac{m_s T_s K_{ws}}{(S_r / p)(p/2)} \right]^2 = \left[\frac{2 m_s T_s K_{ws}}{S_r} \right]^2 \tag{27}$$

where m and S denotes the number of phases and slots; respectively.

The resistance of a squirrel cage winding per phase in terms of the stator winding is then:

$$r_r^{'} = 4 m_s T_s^2 K_{ws}^2 \rho \left[\frac{L_b}{S_r a_b} + \frac{2 D_{er}}{\pi p^2 a_{er}} \right] \tag{28}$$

When the radial width of the end-ring is large in comparison to the bar cross section area, as is the case in small motors, the end-ring component of the cage resistance must be corrected to take into account the non-uniform current distribution in the ring. The end-ring correction factor, K_R, may be taken from [1]. In this work no allowance has been made for the end-ring correction factor as this is regarded to have second order effects.

Reactance Calculations

The magnetizing reactance per phase, x_m, can be calculated when the magnetising current (I_m) and stator leakage reactance (x_l) become known as follows:

$$x_m = \frac{V_{ph}}{I_m} - x_l \tag{29}$$

In the developed software, the magnetising current (I_m) is assumed to be 30% of the phase current. This is a typical value for induction motor of the sizes considered here [1].

The winding leakage reactance of an electric machine is calculated be defining the leakage flux paths and calculating their length and area. For induction motors, it is common practice to divide the total leakage flux into: stator and rotor leakage, zigzag leakage, end-connection leakage, belt or differential leakage and skew leakage [1].

Slot leakage results because not all the flux crossing the air-gap links all of the conductor area in the slot. The slot tapering angle is usually related to the number of slots used, not the lamination dimensions. Slot leakage increases as ratio of the slot depth to width increase.

The permeance coefficient (or specific permeance) of partially closed slots usually encountered in small induction motors may be calculated with the aid of Fig. 2.20 [1].

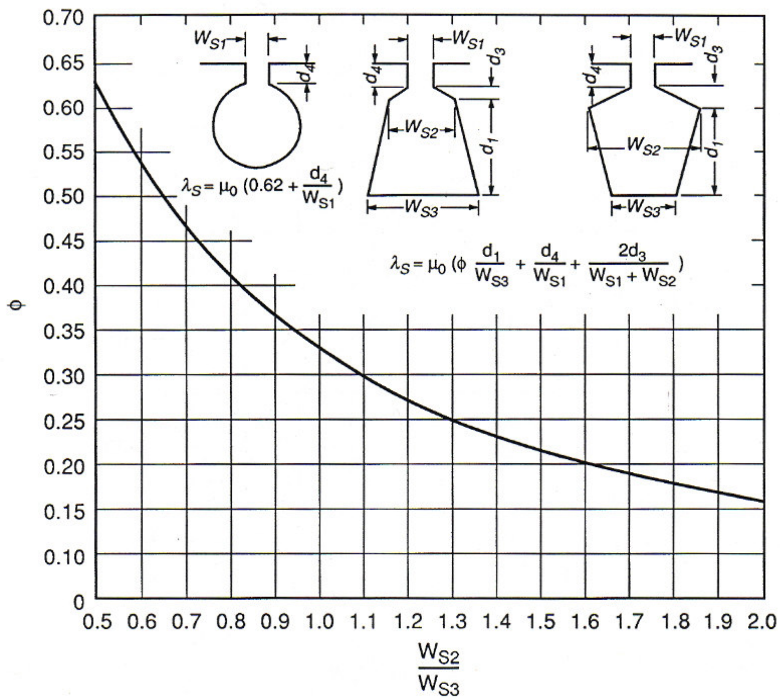


Figure 2.20: The calculation of slot permeance coefficient.

In the software developed here, Fig. 2.20 is stored in database. Details regarding how the curve is approximated on a form of discrete data points and how the search is performed are given in the following chapter.

The rotor slot leakage must be referred to stator, therefore:

$$\lambda_s = \lambda_{ss} + \lambda_{sr} \frac{K_{ws}^2 S_s}{K_{wr}^2 S_r} \tag{30}$$

where λ_{ss} and λ_{sr} are the permeance coefficient values for the stator and rotor slots, respectively.

The total slot leakage reactance per phase is:

$$x_s = \frac{8\pi f T_s^2 L}{pq} \ \lambda_s \Omega \tag{31}$$

In the software developed, the reactance of the end connections is taken as 60% of the stator slot reactance. Of course no such allowance is made in the calculating the rotor leakage reactance.

Now we have all the parameters of the equivalent circuit as shown in Fig. 2.20. The user, who clicks the *Equivalent Circuit* option on the menu of Fig. 2.18, would be, almost immediately, presented with Fig. 2.21. The user would not be aware of how extensive the design procedures that produced Fig. 2.21 are (neither does the user need to know anything regarding machine design). The equivalent circuit method may be used to determine the induction motor performance characteristics under steady state.

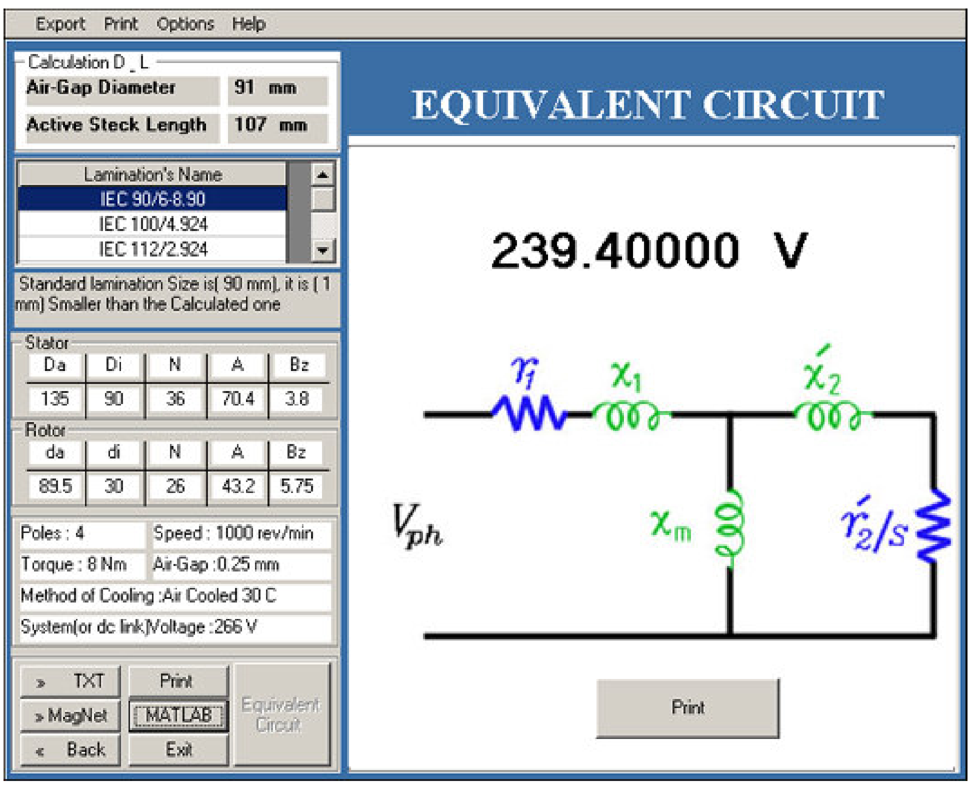


Figure 2.21: Equivalent circuit of an induction motor

2.3.5. MagNet Option

MagNet is a commercial finite element suite of programmes [21]. It is well known to engineers and scientists involved with field computations using numerical field solution that building the problem geometry (commonly known as pre-processing) is the most laborious process. This is particularly true for geometries like those of motor laminations. As, in the developed software, the lamination details are already stored in the database, this represents an opportunity to automate the drawing of lamination.

In the work reported in this thesis an interface is provided between the design tool database and *MagNet* and *AutoCAD*. By clicking the option *MagNet* (Fig. 2.17 or Fig. 2.21), the lamination details (which are stored in the database) are used in connection with a VB subroutine to draw the motor magnetic circuits as shown in Fig. 2.22.

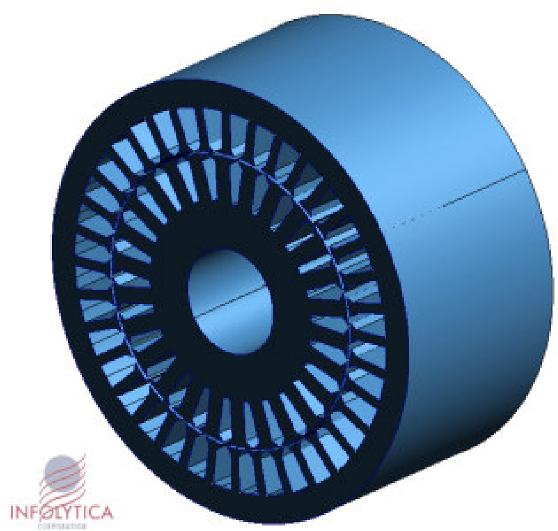


Figure 2.22: Exporting lamination data to MagNet.

The subroutine has been written in VB language, it is embedded in the main software. The codes, which describe it, are given in Appendix II.

The software provides the user with the possibility of drawing only the stator or only the rotor laminations by selecting the *Export* option from the main menu then choosing *Draw only (Rotor Or Stator)*. Fig. 2.23 shows a case when only rotor laminations are drawn.

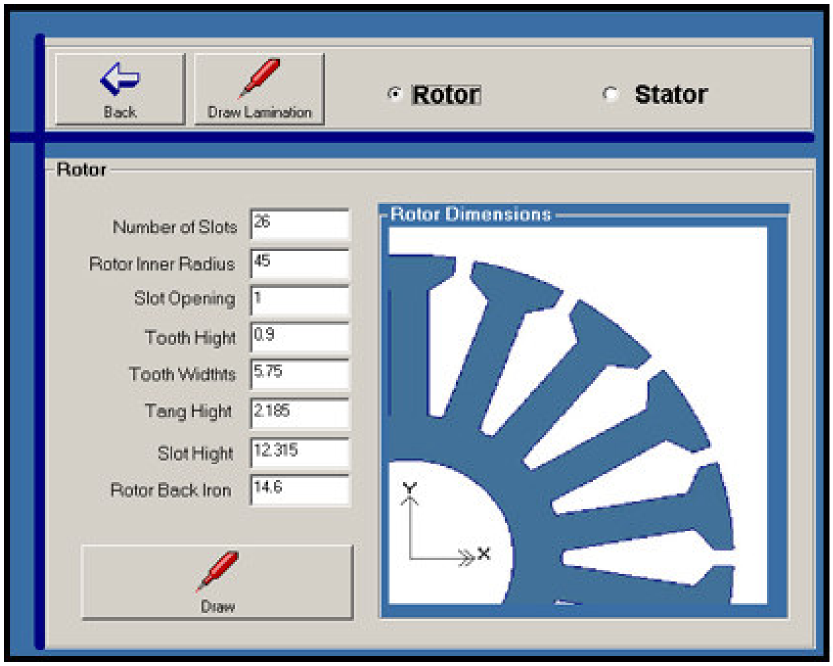


Figure 2.23: Drawing rotor lamination.

With reference to Fig. 2.23, the user is able to draw only the stator, only the rotor, or a complete magnetic-circuit, by providing own values. This window (Fig. 2.23) appears automatically when the search procedure fails to find a matching lamination in the stored database.

2.4. THIRD STAGE OF THE COMPUTATION - DYNAMIC PERFORMANCE

2.4.1. TXT & MATLAB Option

Clicking this option (Fig. 2.17 or Fig. 2.21) triggers storing of essential drive details in a Text file and ***m*** format. This enables the user to investigate transient and short-term performance of the drive system in a MATLAB environment. This stage of the work is carried out at Lund Technical University and, therefore, is mentioned here for completeness. An overview of this work is given in Appendix V.

As far as the work reported in this thesis is concerned, the developed software makes available the parameters in Table 2.6 for further investigations.

Value	**Unit**	**Description**	**Data type**	**Used for**
Type	-	AM or PMSM	String	All
p	-	Number of poles	Integer	All
Ψ_{m}	Vs	Permanent magnet flux	Float	PMSM
R_{s}	Ω	Stator resistance	Float	All
R_{r}	Ω	Rotor resistance, referred to the stator	Float	AM
L_{s}	H	Stator stray inductance	Float	AM
L_{m}	H	Main inductance	Float	AM
L_{r}	H	Rotor stray inductance, referred to the stator	Float	AM
L_{sx}	H	Stator inductance, x-component	Float	PMSM
L_{sy}	H	Stator inductance, y-component	Float	PMSM
I_{prmsmax}	A	Maximum phase current	Float	All
U_{pprmsmax}	V	Maximum phase to phase voltage	Float	-
n_{max}	rpm	Maximum speed	Float	All
U_{n}	V	Nominal voltage (phase to phase)	Float	AM
I_{n}	A	Nominal current (phase)	Float	AM
f_{n}	Hz	Nominal frequency	Float	AM
P_{n}	W	Nominal power	Float	AM
n_{n}	rpm	Nominal speed	Float	AM
y	-	Motor connection, 1 -> star, 0 -> delta	Integer	AM
B_{sln}	T	Nominal tooth flux density	Float	PMSM
B_{yn}	T	Nominal yoke flux density	Float	PMSM
syo	m	Outer stator yoke radius	Float	Thermal
syi	m	Inner stator yoke radius	Float	Thermal
ds	m	Slot height	Float	Thermal
ws	m	Slot width	Float	Thermal
agi	m	Inner air-gap radius	Float	Thermal
ryi	m	Inner rotor yoke radius	Float	Thermal
ksl	-	Copper fill faktor	Float	Thermal
l	m	Stack length	Float	Thermal
ag	m	Air gap	Float	Thermal
s	-	Number of slots	Integer	Thermal
$cool_1$	-	Natural cooling, 1 -> true, 0 -> false	Integer	Thermal
$cool_2$	-	Forced cooling, 1 -> true, 0 -> false	Integer	Thermal
$cool_3$	-	Liquid cooling, 1 -> true, 0 -> false	Integer	Thermal
kc	-	Loss factor according to Steinmetz equation	Float	Thermal

Table 2.6: Motor data available for exporting to other programs

CHAPTER III

Package Analysis

3.1. INTRODUCTION

From a computer scientist viewpoint, the developed software consists of Visual-Basic (VB) as a programming language (user interfaces), Microsoft access database as a storage media, Structured Query Language (SQL) as a programming language for the searching technique. There are several interfaces between VB and some other programs such as AutoCAD, MagNet (Finite element method software), MATLAB, and WordPad (Text editor), as shown in Fig. 3.1.

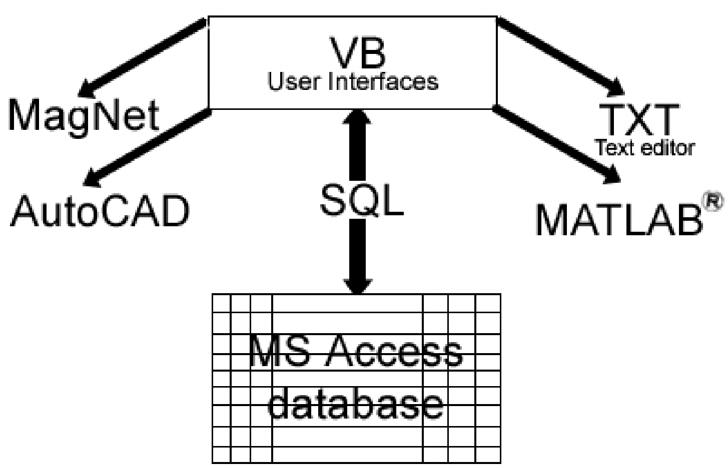


Figure 3.1: The developed software interfaces

3.2. VISUAL BASIC

Visual Basic is a programming language such as C, Fortran, and Pascal etc, which enables a programmer to write programs that are largely independent of a particular type of computer. Such languages are considered high-level because they are closer to human languages and further from machine languages. In contrast, assembly languages are considered low-level because they are very close to machine languages (The lowest-level programming language which are only language understood only by computers). While easily understood by computers, machine languages are almost impossible for human to use because they consist entirely of numbers. Fig. 3.2 shows the classifications of programming languages.

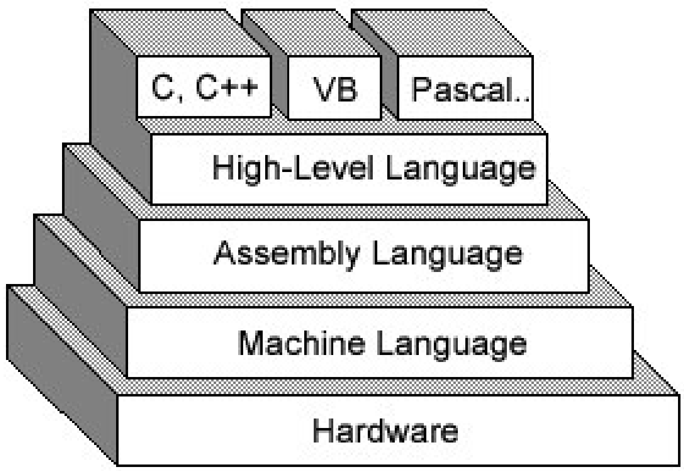


Figure 3.2: The hierarchy of programming languages

The main advantage of high-level languages over their low-level counterparts is that they are easier to read, write, and of course to maintain. Ultimately, programs written in high-level language must be translated into machine language by a compiler or an interpreter (which are languages translate source code into object code).

More people program in Visual Basic than in any other programming language. This has been the case for decades, and it is still true today. Several factors contribute to this phenomenon of its sustained popularity:

- In Visual Basic the *Visual* part refers to the method used to create the graphical user interface (GUI). Rather than writing numerous lines of code to describe the appearance and location of interface elements, you simply add rebuilt objects into place on screen. The *Basic* term refers to the BASIC (Beginners All-Purpose Symbolic Instruction Code) language, a language used by more programmers than any other language in the history of computing [13].
- Visual Basic has evolved from the original BASIC language and now contains several hundred statements, functions, and keywords, many of which relate directly to the Windows GUI. Beginners can create useful applications by learning just a few of the keywords [14].
- The Visual Basic programming language is not unique to Visual Basic. The Visual Basic programming system, applications edition included in Microsoft Excel, Microsoft Access, Microsoft PowerPoint, and many other Windows applications uses the same language [15].
- The Visual Basic Scripting Edition (VBScript) is a widely used scripting language and a subset of the Visual Basic language.

The fastest and easiest way to create applications for Microsoft Windows. Whether the user is an experienced professional or a new-comer to Windows programming, Visual Basic provides the user with a complete set of tools to simplify rapid application development, regardless of what kind of application the user wants to build, stand-alone applications, intranet applications, or internet applications. Some of these tools can be listed as the following:

- Data access features allow the user to create databases, front-end applications, and scalable server-side components for most popular database formats, including Microsoft SQL Server and other enterprise-level databases [16].
- ActiveX technologies allow the user to use the functionality provided by other applications, such as Microsoft Word Processor, Microsoft Excel spreadsheet, and other Windows applications. User can even automate applications and objects created using the Professional or Enterprise editions of Visual Basic.

- Internet capabilities make it easy to provide access to documents and applications across the Internet or intranet from within the user application, or to create Internet server applications.
- The finished application is a true *.exe* file that uses a Visual Basic Virtual Machines that user can freely distribute.

3.3. NETWORK SYSTEMS

In this developed software, several tools are used, such as the data access features, the ability to run the software as a stand-alone application, which means that the database and the execution file of the software are located in the same computer, only one user can run the software or update the lamination database, at the same time, as shown in Fig. 3.3 [22].

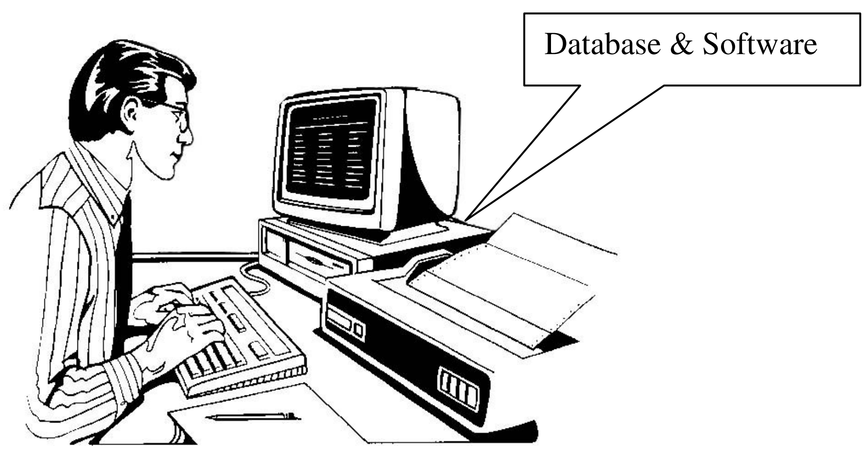


Figure 3.3: Stand-alone application

The developed software can also be used from different users at the time as well. In such a case, the execution file is installed on the clients' machines and the database is located on the server, as shown in Fig. 3.4,which called in this case *server-based networks*, because a dedicated server is one that only functions as a server and is not used as a client or work station. Servers are *dedicated* because they are optimised to quickly service request from network clients and ensure that the security of files and directories. Server-based networks have become the standard model for networking in recent years.

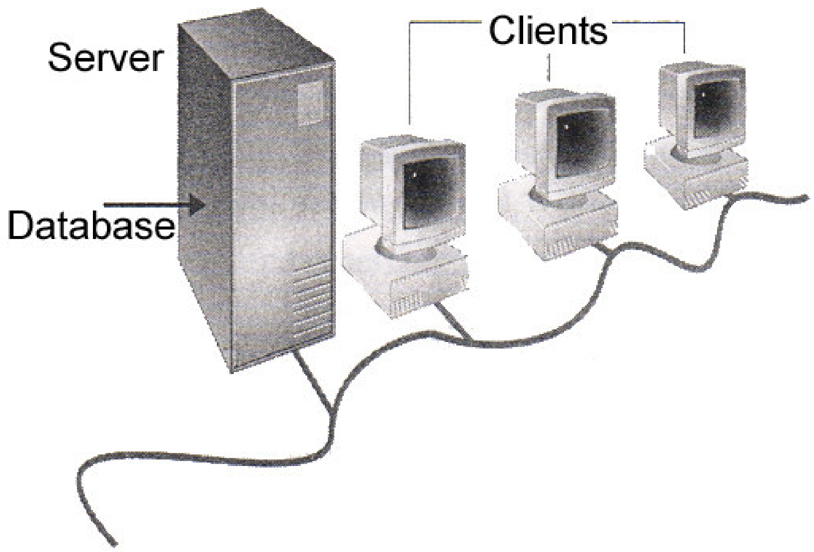


Figure 3.4: Multi-users application

The developed software can be installed in peer-to-peer networks where there are no dedicated servers or hierarchy among the computers. All of the computers are equal and therefore are known as peers. [22]
Normally, each computer function as both client and server, and there is no one assigned to be an administrator responsible for the entire network. The user of each computer determines what data on their computer gets shared on the network. The database is located in one of these computers under a shared folder, and the execution file is installed on each computer, including the one which holding the database, each user can reach the database through the sharing folder, as shown in Fig. 3.5.

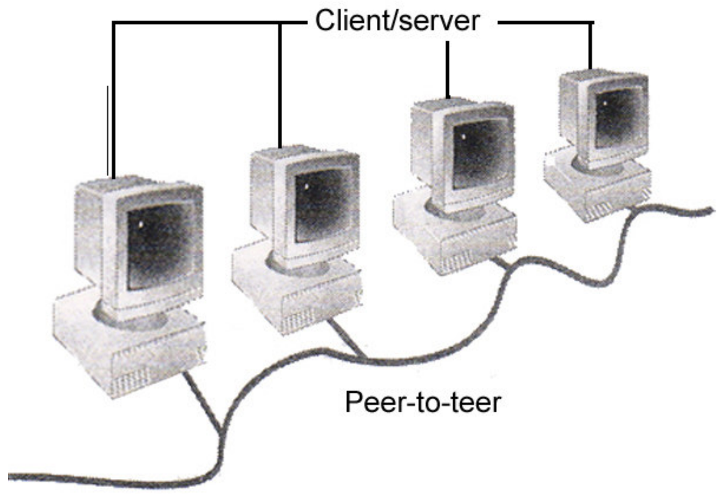


Figure 3.5 Peer-to-peer network computers

In the both mentioned network systems (Peer-to-peer networks and server-based networks), the developed software is regarded as an application server. Application servers make the server side of client/server applications, as well as the data, available to clients. For example, servers store vast amounts of data that is structured to make it easy to retrieve. This differs from a file and print server. With a file and print server, the data or file is downloaded to the computer making the request. With an application server, the database stay on the server and only the results of a request are downloaded to the computer making the request.

A client application running locally would access the data on the application server. Instead of the entire database being downloaded from the server to a local computer. For example, when the user search for the matching lamination, only lamination(s) is (are) found is (are) subsequently downloaded.

Before data can be sent over the network, the network adapter card must change it from a form that the computer can understand to another form, which can travel over a network cable. Data travels through a computer along paths called busses. These are actually several data paths placed side by side. Because several paths are side by side (parallel), data can move along them in groups instead of single (serial) data steam.

Older busses were known as 8-bit busses because they could handle only 8-bits of data at a time. Now it is possible to find 16-bit bus, which means it could move data 16 bit at a time, 32-bit bus, and even more. On the network cable, data must travel in a single bit stream. When data travels on a network cable it is said to be travelling as a serial transmission because one bit follows another. In other word, the cable is like a one-lane highway. The data on these highways always travel in one direction. The computer is either sending or receiving data.

The network adapter card takes data travelling in parallel as a group and restructures it so that it will follow through the 1-bit wide serial path of the network cable. This is accomplished through the translation of the computer's digital signals into electrical and optical signals that can travel on the network's cable. The computer responsible for this is the transceiver (transmitter/receiver).

3.4. SQL

Structured Query Language (SQL) is a standard interactive and programming language for getting information from- and updating a database. As SQL is both an ANSI (American National Standards Institute) and an ISO (International Organization for Standardization) standard, many database products support SQL with proprietary extensions to the standard language.

SQL statements are used to perform tasks such as update data on a database, or retrieve data from a database. In other word, SQL is a way of asking questions, or giving orders to a database. The database itself is of course just a collection of information.

Some common relational database management systems that use SQL are: MS Access, DB2, Informix, MS SQL Server, Oracle, Sybase, etc. Although most database systems use SQL, most of them also have their own additional proprietary extensions that are usually only used on their system. However, the standard SQL commands such as *Select*, *Insert*, *Update*, *Delete*, *Create*, and *Drop* can be used to accomplish almost everything that one needs to do with a database [17].

A database most often contains one or more tables. Each table is identified by a name (e.g. *Customers* or *Orders*). Tables contain records (rows) with data. Below is an example of a table called *Persons*, which belong to the database *Information*, which in turn contains several tables.

LastName	**FirstName**	**Address**	**City**
El hinaoui	Hasan	Tunnlandesgatan	Gothenburg
Hamdi	Essam	Cyncoed	Cardiff
Jucobsson	Frida	Kungsbacka	Kungsbacka
Andreasson	Julia	Kortedala	Kortedala

Table 3.1: Persons database

Table 3.1 contains four records (one for each person) and four columns (LastName, FirstName, Address, and City). With SQL, users can query a database and have a result set returned.

To enable the SQL to search inside the database, the user must make the connection between them, and to make the connection between SQL and the database from VB, then the tool (data) is used, and by linking the following properties the connection will be done.

"Connect": Access
"Database": Information
"Recordsource": Persons

Fig. 3.6 illustrates the connection between the VB code and the database.

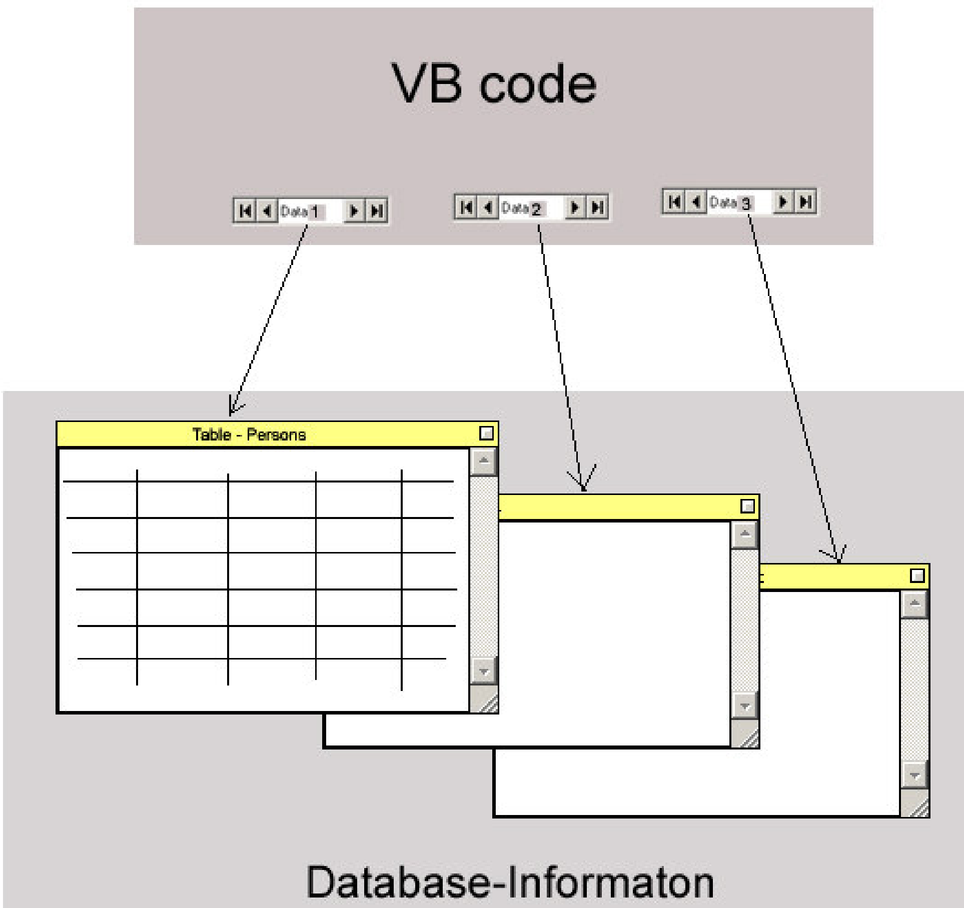


Figure 3.6: Connection between VB and a stored database

With reference to table 3.1 (which represents the database) the following examples explain how SQL works:

- **Query 1:**

Data1.RecourdSource="SELECT LastName FROM Persons"

Gives a result set like this:

LastName
El Hinaoui-Hamze
Hamdi
Jucobsson
Andreasson

- **Query 2:**

Data1.RecourdSource="SELECT FirstName FROM Persons"

Gives a result set like this:

LastName
Hasan
Essam
Frida
Julia

- **Query 3:**

Data1.RecourdSource="SELECT FirstName FROM Persons where city="Gothenburg"

Gives a result set like this:

LastName
Hasan

In this example we used SQL with a condition "city=Gothenburg".

It can be seen from the example above that SQL is a syntax for executing queries. But the SQL language also includes a syntax to update, insert, and delete records.

These query and update commands together form the Data Manipulation Language (DML) part of SQL:

SELECT - extracts data from a database table
UPDATE - updates data in a database table
DELETE - deletes data from a database table
INSERT INTO - inserts new data into a database table

The result from SQL query is stored in a result-set. Most database software systems allow navigation of the result set with programming functions, like: Move-To-First-Record, Get-Record-Content, Move-To-Next-Record, etc.

SQL is used in the developed software for searching for the matching lamination(s) in the database. The calculated air-gap diameter (D) is used as the search input. Remembering that, in the stored database, D_i is used to denote the lamination inner diameter, the following SQL statement is used to perform the search:

```
Data1.RecordSource = "select StandaredLamination where Di =" + D
```

The ***Data1*** tool is making the connection between the VB code (software), and the table ***StandaredLamination*** of the database ***GreenCar***. The ***Select*** instruction will select all the laminations from the database which have **Di** equal **D**, where **D** represents the Calculated air-gap diameter, and **Di** represents the air-gap diameter value for the stored laminations.

The search could yield one of the following results:

- One (or more) matching lamination is (are) found in the database.

- No matching laminations are found in the database.

In the second case, the user will get two options, as shown in Fig. 3.7.

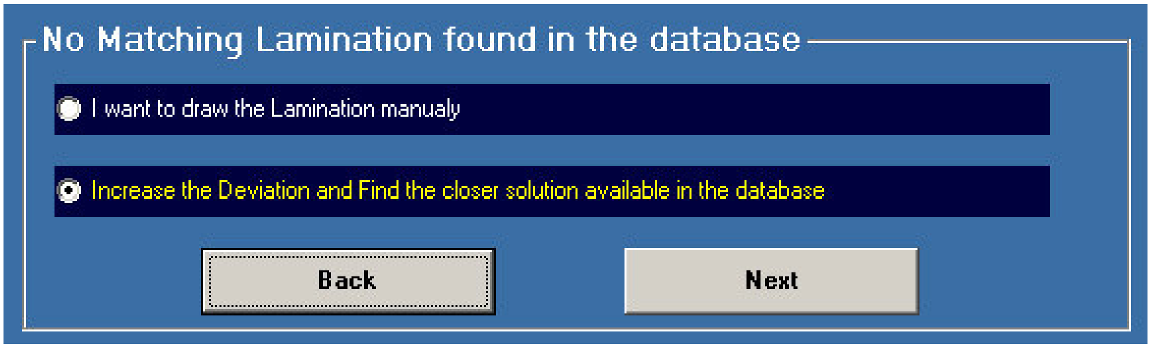


Figure 3.7: Users options when no matching standard laminations are found

The first option makes the user able to draw a lamination manually, by giving the relevant dimensional details, as shown in Fig. 3.8. The menu of Fig. 3.8 allows the possibility to draw:

- The rotor lamination only.
- The stator lamination only.
- Or the complete magnetic circuit (rotor and stator laminations)

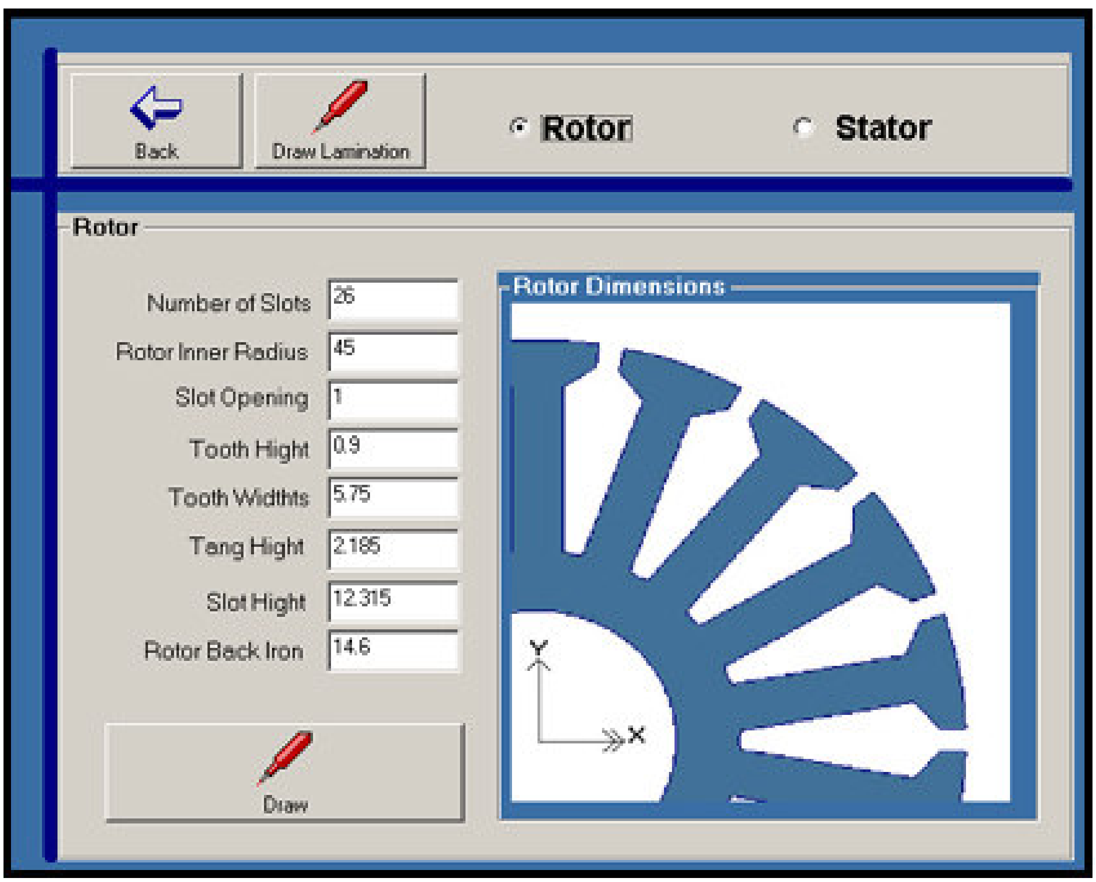


Figure 3.8: Drawing of a user-specified lamination

The second option (Fig. 3.7) enables the user to find the closer lamination(s), which match user requirements, by increasing the deviation of the searching step. For instance, assuming that **D** (Calculated air-cap diameter) is equal to 72 mm, and assuming that the database has the **Di** values for the stored laminations as per Table 3.2:

Di	Lamination's name
69 mm	IEC 70/1.32
70 mm	IEC 90/2.70
70 mm	IEC 80/4.70
70 mm	IEC 71/6-8.70
74 mm	IEC 90/7-3.80

Table 3.2: Example of stored lamination details

As tables 3.2 shows, there are no laminations stored with Di = 72 mm, in this case the deviation of the search step will be changed from 72 to (71 or 73):

Deviation of the search step = Deviation of the search step + 1

And

Deviation of the search step = Deviation of the search step – 1

Then SQL searches again in the database for matching lamination(s) with (Di = 73 or 71 mm), and as table 3.2 shows, there are no stored laminations with these values. The deviation of the search step will be changed again:

Deviation of the search step = Deviation of the search step + 2

And

Deviation of the search step = Deviation of the search step – 2

SQL searches again in the database for matching lamination(s) with (Di = 70 or 74 mm), and as table 3.2 shows, the results will as shown in Table 3.3.

D	Lamination's name
70 mm	IEC 90/2.70
70 mm	IEC 80/4.70
70 mm	IEC 71/6-8.70
74 mm	IEC 90/7-3.80

Table 3: Search output

3.5. DATABASE

A database is a system that contains many different objects used together to allow the application fast and efficient access to the data. Many examples of database systems can be used with a given application. The most common ones are as follows:

- Microsoft Access.
- Microsoft FoxPro.
- dBase.
- Oracle.
- MS SQL Server.
- Sybase.

All these databases can run on a standalone computer and allow the user to create very complex databases for their applications. Choosing the right database for the application is extremely important, because the wrong one will affect performance and make the job as a developer so much more difficult. For example, due to their complexity, Oracle and MS SQL server databases should only be used when the size of the stored data exceeds what Access and FoxPro can efficiently handle [16].

Before starting the design process, a database needs to be selected. When working with Visual Basic, the database that can be chosen fall into two distinct groups:

- Local Databases: can be accessed directly from Visual Basic through Visual Basic's database Jet engine; and
- Remote databases: cannot be accessed by using Visual Basic's standard database access capabilities.

Understanding the differences will help to select the correct database for the application. However, in Visual Basic 6, there is now a way to access both local and remote databases using the same database access controls and objects known as the ActiveX Data Objects.

3.5.1. Local Databases

Local databases are generally much smaller than the remote type of database. The following databases fall into this category:

- Microsoft Access.
- Microsoft FoxPro.
- Lotus Worksheets.
- Microsoft Excel Worksheets.
- dBase.
- Paradox.

These PC-resident database types have been around for decades and are widely available. Accessing these databases doesn't require any other software to be installed on the PC, nor does it require special knowledge on how to work with them, and these are distinct advantages. However, the size of local databases is restricted. Microsoft Access 97 databases can't exceed 1GB (gigabyte).

Local databases are generally used for single-user applications such as address books, personal information managers and processing of student's examination results. For example, some 20 years ago, my supervisor used Lotus to collate, process and prepare students' examination results at Cardiff University. Also, some teenagers have created database applications that help them keep track of baseball cards or videotapes! So in deciding what your application will address and how it will be used actually helps in the database selection process.

3.5.2. Remote Databases

By definition remote databases don't reside on the user's PC, but this is not always true. In fact, a remote database is any type of database that requires an ODBC (Open Database Connectivity) driver for an application to access it. ODBC databases fall into two main categories: one that run on a single PC and the larger corporate ones that require large, very powerful computers (called servers) with large amounts of available disk space. Database servers are used to separate the workload between the client PC (where the application executes) and the database system (where the database queries are performed). This provides the application with fast data access without slowing down the users PC.

Most the large server-type databases provide versions that will run on standalone PCs. The most common databases used in connection with large records are:

- Oracle.
- Sybase.
- Microsoft SQL Server.

Using a remote database requires either the remote database objects being available only in the Enterprise version of Visual Basic 6 or the new ActiveX Data Objects available in all releases of Visual Basic 6. When designing an application that will use one of these large databases, a good practice would be to create the application that will use the single-PC version during the design, development and testing process. Then, after testing, the application is transferred via ODBC connection to the larger, server-based version. This is indeed the approach adopted here; the software, and its associated databases, are developed and tested on a dedicated PC. Following this, operation was extended to cover multi-user server system.

3.5.3. GreenCar Database

GreenCar is the name used for the database of the developed software. This is a Visual-Basic access database and it contains several tables; for example, User input, Lamination, Stator, Rotor1, Rotor2, as shown in Fig. 3.9. These tables are described below in details.

It should be noted that there are other tables in this database (e.g. the force density values used in the first stage of the computation and the slot permeance coefficients, used in determining the slot leakage in the second stage of computation as detailed in chapter II). These tables are not shown in Fig. 3.9, in order to simplify the presentation and discussion.

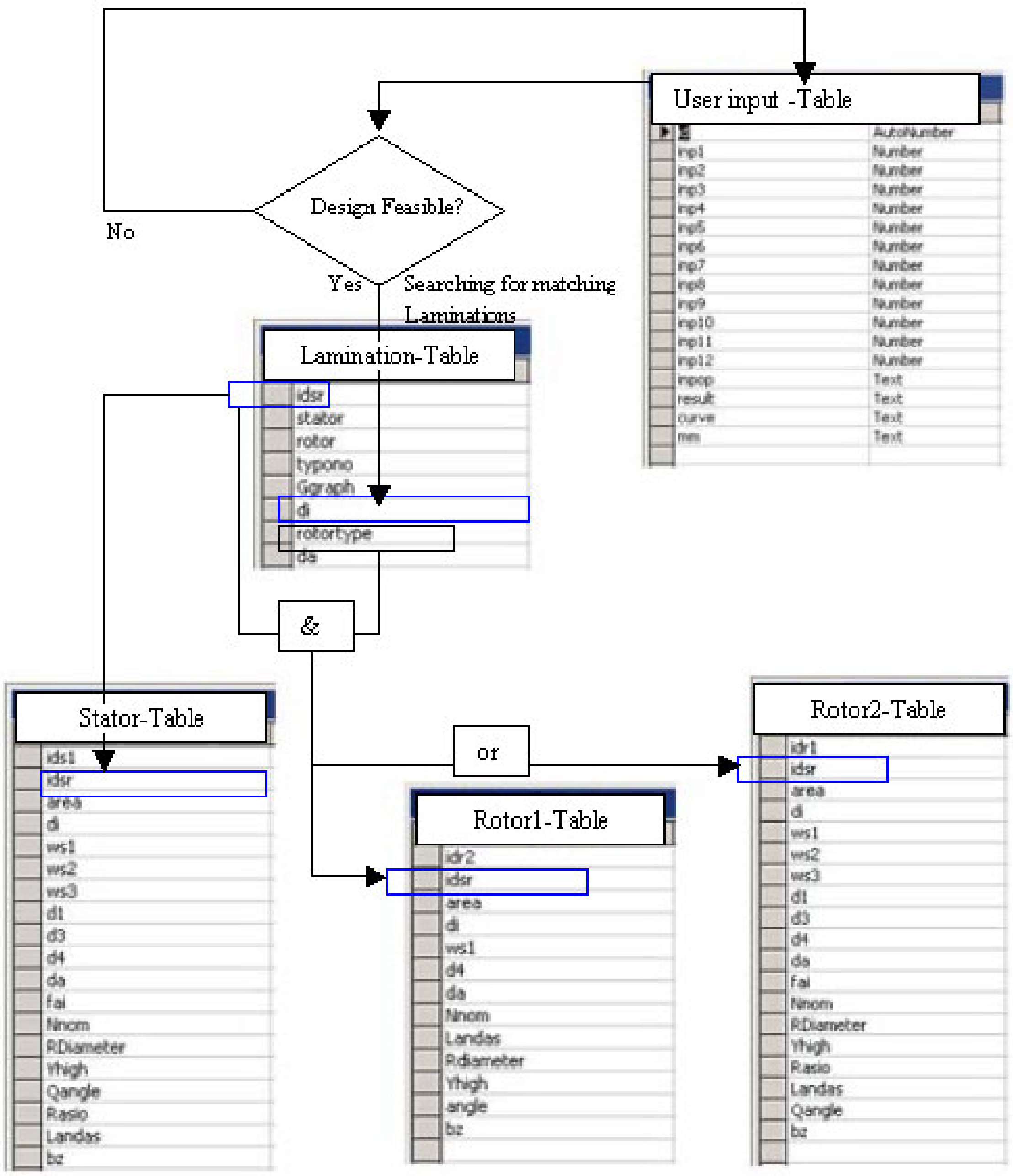


Figure 3.9: A section of the GreenCar database

3.5.3.1. User input Table

This table is accessed directly by the users. They can save, delete, and update the *user define parameters*, as shown in Fig. 3.10.

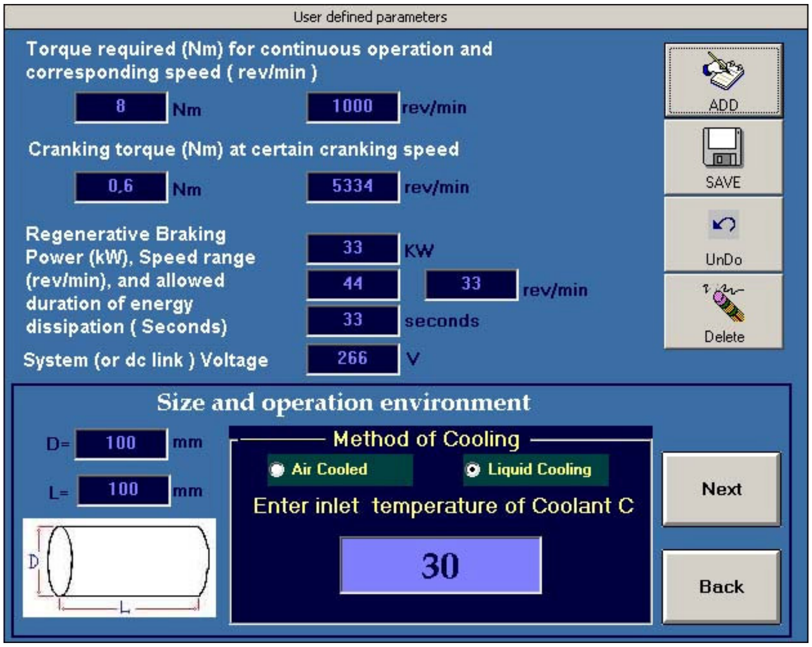


Figure 3.10: User defined parameters

User input table contains records, which in turn consists of several fields, which are used to save the user parameters such as (Torque, Speed, Method of Cooling, System Voltage, Envelope space. …etc).

Users are allowed to add a new record, delete, and update, and for any slight change in any field of the record, the record must be saved before proceeding to the next stage. The codes related to each of the virtual buttons of Fig. 3.10 are now described.

Add button:

```
Private Sub Command1_Click()
        FillData
        Data1.Recordset.AddNew

          Text1(0).SetFocus
          Option1(0).Value = True

          Command1.Enabled = False
          Command3.Enabled = False
          SSCommand3.Enabled = False
          SSCommand2.Enabled = False
End Sub
```

As shown in the code above, the *Add* code starts with a subroutine called *FillData* which is responsible to make the connection with the local or remote database, after that a new record will be added to the table by using:

Data1.Recordset.AddNew

At this moment the user has no option to proceed to the next stage. This is only possible after saving the new record or cancelling the adding operation. This is in order to protect the database from unexpected errors.

The *FillData* code is as follows:

```
Sub FillData()
        If Drv = "" Then
           masar = App.Path
           If Right(masar, 1) = "\" Then
              db2 = masar + "Green.mdb"
           Else
              db2 = masar + "\Green.mdb"
           End If


        Else
           masar = Drv 'App.Path
           If Right(masar, 1) = "\" Then
              db2 = masar + "Green.mdb"
             Else
              db2 = masar + "\Green.mdb"
           End If
        End If
        'ss = App.Path + "\Green.mdb"

        Form3.Data1.DatabaseName = db2: Form3.Data1.Refresh

        Form11.Data1.DatabaseName = db2: Form11.Data1.Refresh
        Form11.Data2.DatabaseName = db2: Form11.Data2.Refresh
        Form11.Data3.DatabaseName = db2: Form11.Data3.Refresh
        Form11.Data4.DatabaseName = db2: Form11.Data4.Refresh


        Form12.Data1.DatabaseName = db2: Form12.Data1.Refresh
        Form12.Data2.DatabaseName = db2: Form12.Data2.Refresh
        Form12.Data3.DatabaseName = db2: Form12.Data3.Refresh
        Form12.Data4.DatabaseName = db2: Form12.Data4.Refresh
```

```
        Form12.Data5.DatabaseName = db2: Form12.Data5.Refresh
        Form12.Data6.DatabaseName = db2: Form12.Data6.Refresh
End Sub
```

"Save" Button:

```
Private Sub Command2_Click()
          On Error GoTo ss
          ValueOfspeed = Text1(1)
          If ValueOfspeed < 250 Then
            Massegz = MsgBox("The speed should be at least 250 rev/min", vbOKOnly,
        "Caution")
            Text1(1).SetFocus
            Exit Sub
          End If


          Data1.UpdateRecord
          Data1.Recordset.Bookmark = Data1.Recordset.LastModified
          Command3.Enabled = True
          Command1.Enabled = True
          SSCommand3.Enabled = True
          SSCommand2.Enabled = True

          Timer2.Enabled = False

          Command2.BackColor = RGB(192, 192, 192)


          Exit Sub
        ss:     '524 or ""

          StrMsg = "Invalid input value" & vbCrLf
          StrMsg = StrMsg + "Please check the input Values"
          MsgBox StrMsg, vbCritical + vbOKOnly, "Caution"
          Command2.SetFocus

End Sub
```

The input values are examined before saving into the record and a warning message appears, for example, when the specified speed value is less than 250 rev/min, as shown in Fig. 3.11. This is because the design method programmed into the engineering tool does not cater for such a low-speed machines.

Figure 3.11: Warning messages are provided when the data specified are not catered for

Another warning message appears when one of the other required inputs has an unrealistic value, for example (Torque=-8 or system voltage=266y. …etc), as shown in Fig. 3.12.

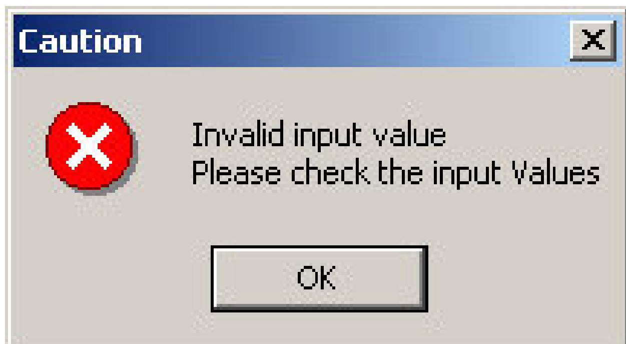


Figure 3.12

Undo Button:
This allows the user to cancel the process, if needed.

Delete Button:
This option is self-explanatory. The code is as follows:

```
Private Sub Command3_Click()
        If Data1.Recordset.RecordCount <> 0 Then
          X = MsgBox("Are you sure you want to delete this record ? ", vbInformation + vbYesNo, " Caution !")
          If X = vbYes Then
            Data1.Recordset.Delete
            Data1.Recordset.MoveNext

          Else
            Exit Sub
          End If
        Else
```

```
            ms = MsgBox("The DataBase is empty ", vbInformation + vbOKOnly, "Caution")
        End If
End Sub
```

It should be noted that there are two important things that must be taken into account when the user wants to delete a record. Firstly, the table *User input* must have at least one record to be deleted. Secondly, a warning message to warn the user that he/she is about to delete the current record, to avoid an unhappy accident, as shown in Fig. 3.13.

Figure 3.13

3.5.3.2. Lamination Table

This table holds all the information regarding the laminations: the lamination's name, its air-cap diameter, outer diameter, number of slots in the rotor, number of the slots in the stator, and the lamination's drawings, as shown in Fig. 3.14.

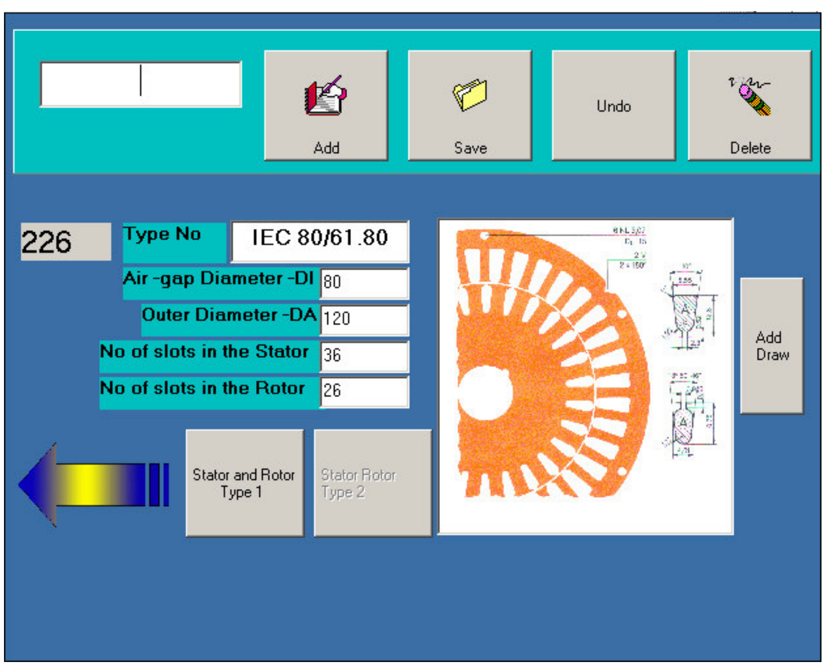


Figure 3.14: Lamination details

Access to this table is not available for the normal user; only the administrator is allowed to update this section of the database. Therefore, a user trying to update the laminations table will be asked to enter the password, as shown in Figs. 3.15 and 3.16 [18]

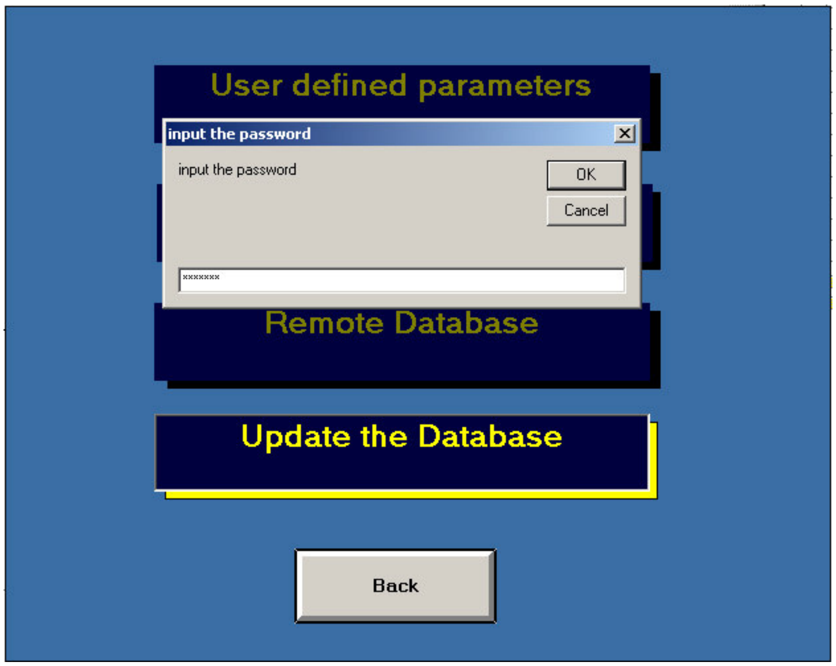


Figure 3.15

Figure 3.16

The main purpose of this facility (Add button, top left of Fig. 3.14) is to enable the administrator to update the database (add a new lamination or update old ones). In this screen (Fig. 3.14) the administrator can add a new lamination to the laminations table. While the virtual buttons *Add*, *Save*, *Undo*, and *Delete* of Figs. 3.10 and 3.14 appear to be identical (and indeed their associated functions are) only the administrator have control on those of Fig. 3.14.

The new thing in this table *Lamination* is the lamination's drawing. In general there are two ways to save the drawings in the database.

The first possibility is to save the drawing under a specific folder (outside the database), and saving a track number inside the database, with each track number refers to a specific drawing. In this case, when the search criteria match a lamination in the database, the track number leads the software to the folder containing all the lamination's drawings. Then the matching lamination drawing is loaded to the user interface program, as shown in Fig. 3.17.

The main advantage of this strategy lies in reducing the size of the database. This is because the drawings require a large space (in the database) in comparison with the track number. This in turn would affect the time of searching operation, but in the other hand, there are disadvantages for this strategy as well. For example, in case one or more drawings of laminations are deleted from the folder by accident, or if the folder is moved to another place in the hard disk, or even if the name of the folder is changed, or if one of the drawings' name is changed, in all these cases the track number will cause an unavoidable error.

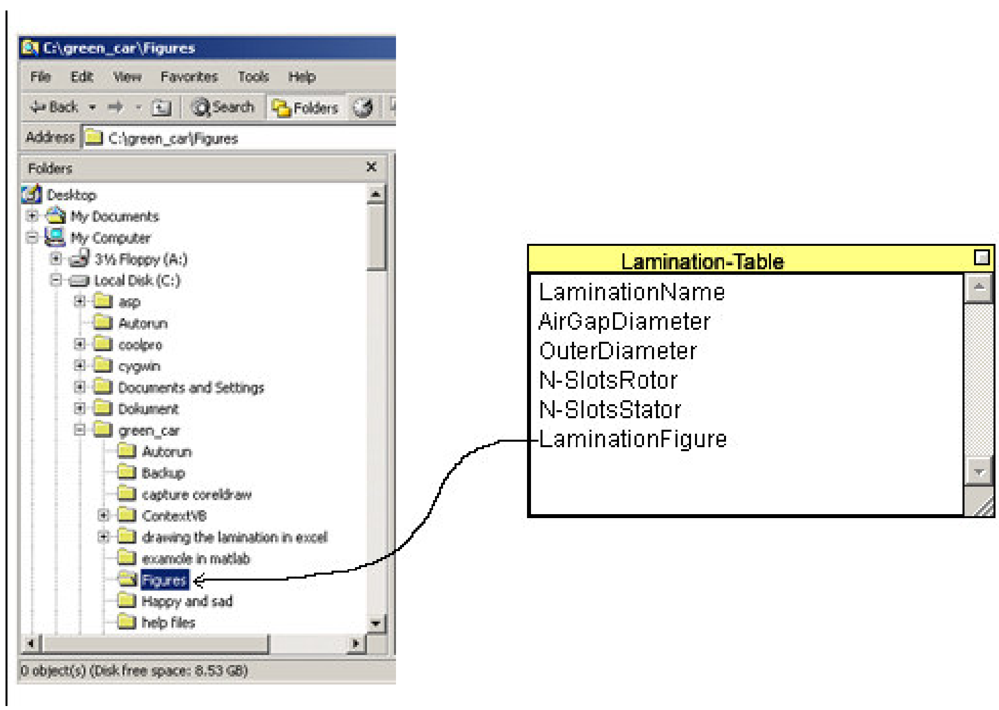


Figure 3.17.

The second way is to save the drawing inside the database itself as a binary filed. Of course this will increase the size of the database, which in turn affects the searching time (the searching time will be increased). This drawback, however, is regarded as insignificant, since SQL is a very powerful tool and can handle large databases efficiently. In effect, this (second approach) eliminates the drawback of the first method (track number, described above). This is because the drawing becomes a part of the entire record; if the record is deleted then the drawing is also deleted, there is no way to separate the drawing from the record.

Indeed, the second strategy has been used in this developed software. Extensive testing has shown that the search time, at a fraction of a second, is quite acceptable.

The GreenCar database already includes hundreds of laminations, and there are no limitations regarding the size of this database (other than the data storage capability of the computer used). This means that more laminations can be added to the database; just only the administrator is allowed to do that. Fig. 3.14 shows how to add a new lamination. By clicking on *stator* and *rotor type* buttons, stator and rotor details can be added to the database, as shown in Fig. 3.18.

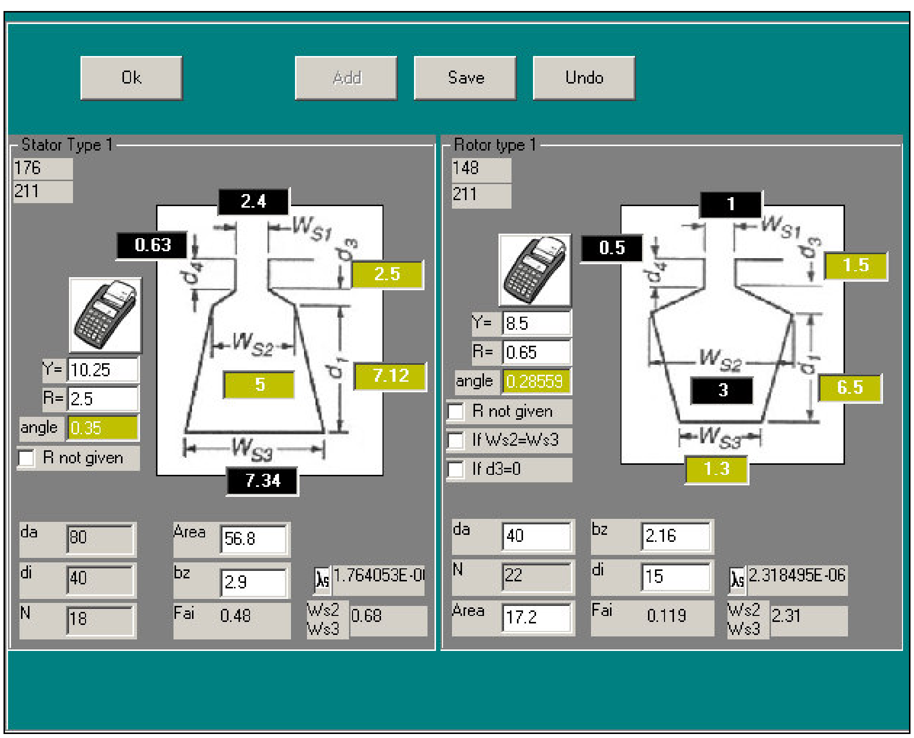


Figure 3.18: Adding new laminations to the database

Some information is available in the lamination catalogues, and can be added to the database directly by the administrator (such as the outer and inner diameter of the stator lamination, slots area, teeth width and shaft diameter) as shown in Fig. 3.19.

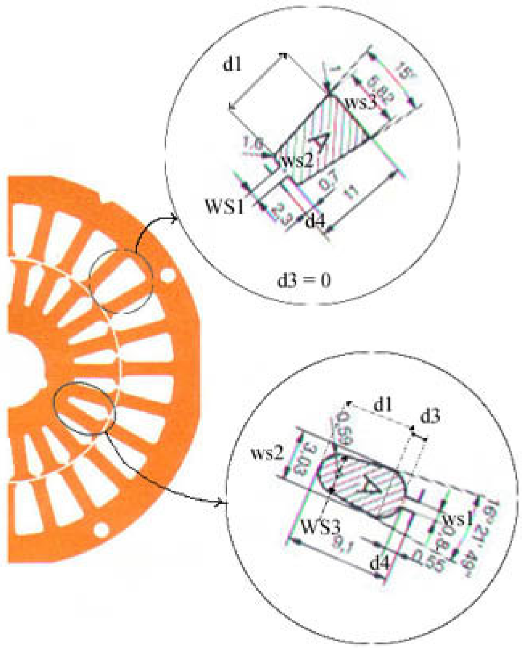


Figure 3.19: Lamination details

Other parameters need to be calculated by the software itself. For Example, the slots permeance coefficients need to be calculated and stored for further use. The slot permeance coefficient is given by the following equation [1]:

$$\lambda_s = \mu_0 (\phi \frac{d_1}{W_{S3}} + \frac{d_4}{W_{S1}} + \frac{2d_3}{W_{S1} + W_{S2}})$$

where μ_o is permeability of the free space, H/m and ϕ is (here) a dimensionless coefficient determined with the aid of Fig. 3.20.

In order to calculate the slot permeance coefficient, the data of Fig.3.20 must be stored in the database. This is done in the developed software by taking discrete data with a step of 0.01 on the x-axis ($\frac{W_{S2}}{W_{S3}}$) and storing the corresponding values of the y-axis. This data is stored in the GreenCar database as Slot-Permeance Table (details are given in Appendix IV).

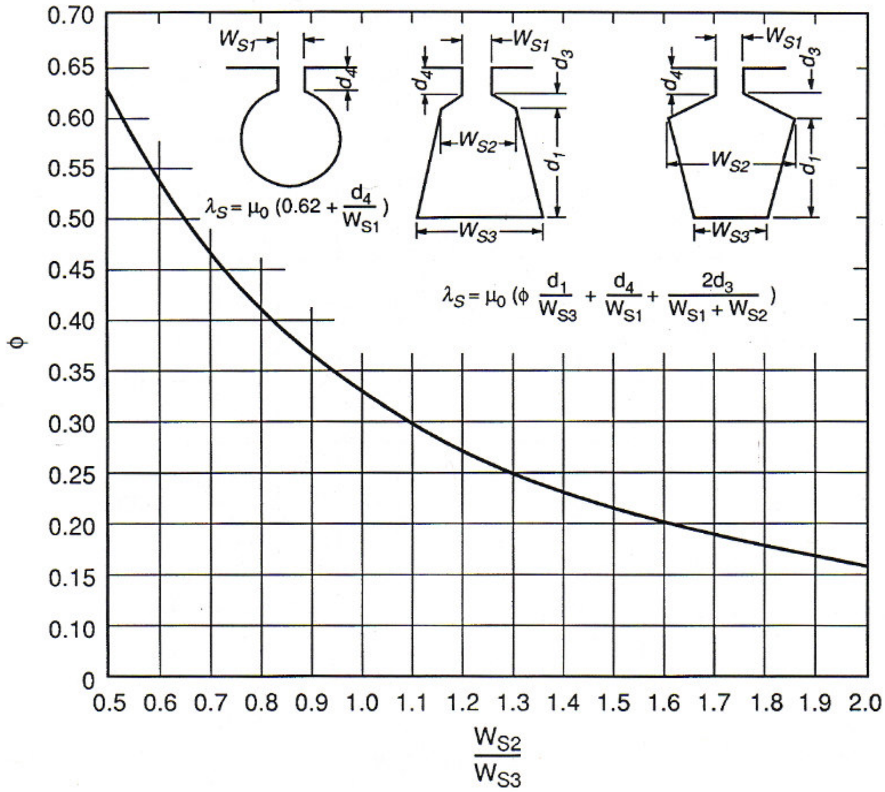


Figure 3.20: Calculation of slot permeance coefficient

After calculating $(\frac{W_{S2}}{W_{S3}})$ for a particular design, this value is approximated to the nearest stored value (see Appendix IV). Bering in mind the small step used, the error here would be insignificant. Indeed it would be within what would normally occur if the actual curve is used with the human eye.

3.5.3.3. Remote GreenCar Database

As is mentioned above, the GreenCar database can be accessed locally (from the same computer), or remotely. In the second case the database is located on a specific computer (server) and can be used by a number of remote PCs simultaneously. The window that allows a user to specify a remote database server is shown in Fig. 3.21.

Of course there are several benefits for this option and these are summarised as the follows:

- Saving disk space: sharing the GreenCar database will save some disk space on the client's computer. Only the execution file (Green-car.exe) needs to be installed on the client's computer.

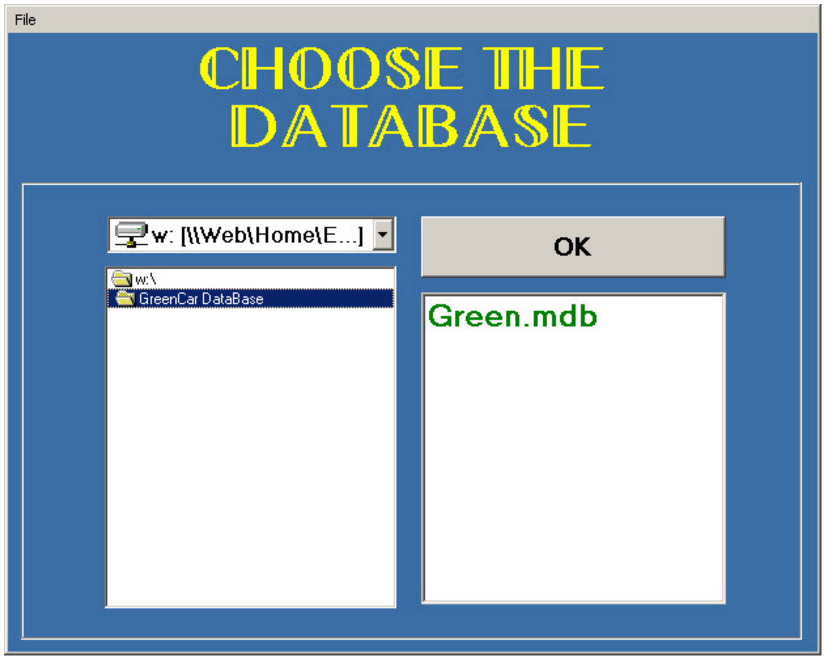


Figure 3.21

- Saving memory: by accessing the GreenCar database remotely, the database will be installed into the server memory, which always is provided with a big capacity of memory. Only relevant information will be loaded to the client's PC memory from the server database. In other word, this option will protect the user from *the out of memory* situation.
- Safety: the safety of the database is one of the important things. To protect the database from any damage caused by misusing is achieved by keeping the database in unreachable place remote from the user. The server is one of the safest places to protect data and prevent any damage or losses to the data. Only the administrator has access to the server and the server itself (and its backups) can be housed in secure buildings (in case of fire, etc).

3.5.3.4. Database Backup

The developed software allows the user to make a backup for certain parts of the database (e.g. user defined parameters). Under *File* in the main window menu (Fig. 3.21) the user can choose to make a backup copy of the database in case of an emergency, as shown in Fig. 3.22.

Figure 3.22

The code below describes how to make a backup within VB, with the possibility to make a local backup or remote backup. The backup database will be named with the current date. For example, *02012005.mdb* is the backup for the *GreenCar.mdb* database, which has been created on 02-01-2005.

```
Private Sub BackupDatabase_Click()

    On Error GoTo CcheckError

    TheDateOfTody1 = Format(Now, "dd,mm,yyyy")
    DayOfToday = Format(Now, "dd")
    MonthOf = Format(Now, "mm")
    DayOf = Format(Now, "yyyy")
    TheDateOfTody = DayOfToday + MonthOf + DayOf

    Set fso = CreateObject("Scripting.FileSystemObject")
    SetPath = App.Path
    Set a = fso.GetFile(SetPath + "\Green.mdb")
    a.Copy (SetPath + "\Backup" + "\" + TheDateOfTody + ".mdb"), False
```

```
Y = MsgBox("A backup file has been copied successfully.", vbInformation + vbOKOnly,
"Green Car")
Exit Sub

'------------CcheckError

CcheckError:

If Err = 58 Then

    StrMsg = "Database is already saved" & vbCrLf
    StrMsg = StrMsg + "Do you want to save over it?"

    ms = MsgBox(StrMsg, vbCritical + vbYesNo, "Caution")
    ddd = ms

    If ms = vbYes Then

      SetPath = App.Path
      Set a = fso.GetFile(SetPath + "\Green.mdb")
      a.Copy (SetPath + "\Backup" + "\" + TheDateOfTody + ".mdb"), True
              Y = MsgBox("The backup database has been created successfully.",
          vbInformation + vbOKOnly, "Green Car")
    Else

     Exit Sub

    End If

Else

    If Err = 76 Then

        StrMsg = "The folder *Backup* is not found," & vbCrLf
        StrMsg = StrMsg + "Do you want to create one ?"
        ms = MsgBox(StrMsg, vbCritical + vbYesNo, "Caution")

        If ms = vbYes Then

          SetPath = App.Path
          MkDir SetPath + "\Backup"
          SetPath = App.Path
          Set a = fso.GetFile(SetPath + "\Green.mdb")
          a.Copy (SetPath + "\Backup" + "\" + TheDateOfTody + ".mdb"), True
                  Y = MsgBox("A backup file has been copied successfully.", vbInformation
           +  vbOKOnly, "Green Car")

        Else
```

```
            Exit Sub
        End If
    Else
        Y = MsgBox("Unexpected error, Please contact the Programmer.", vbCritical + vbOKOnly, "Green Car")
    End If
  End If
```

End Sub

CHAPTER IV

Conclusions and Future Work

4.1. Conclusions

A software-based Engineering tool to assist designers of HEV's in selecting and sizing-up of suitable traction drives is reported in this thesis. It combines specialised electric drives know-how with software design and data search techniques in an efficient way. The developed software utilizes a special-purpose database containing the following tables:

- Force density values for a range of motor sizes and for various cooling arrangements,
- Standard laminations details, and
- Slot pearmeance coefficients.

The software offers the following features:

- Visual Basic has enabled development of a user-friendly interface and allowed adequate use of graphical illustrations.
- By utilising normalised force density curves, the software provides the users, almost instantaneously, with an indication regarding the feasibility of their requirement.
- The user can choose a particular drive system topology for further investigation.
- The modular structure of the software allows addition, and deletion, of topologies as the need arises. This makes it possible to modify the

engineering tool so that it can be used in conjunction with applications other than HEVs.

- Slot permeance coefficients are calculated for every lamination set and are stored, in the laminations table, together with other details.
- While actual lamination drawings and details are stored in the database, the search method used (SQL) yields results in a fraction of a second (on a standard PC).
- Equivalent circuit parameters are calculated and are available for exporting to other system analysis and design programs.
- An interface is provided with a commercial finite element suite of programs. This would enable detailed motor design work to be carried out, should the need arise.

Preliminary testing of the developed software has been very encouraging and warrants further development and extension of its scope. The main lines of extending the software are:

- To fully develop topologies such as SR (switched reluctance) and brushless permanent-magnet drives and to add detailed topology description,
- To incorporate dynamic performance prediction into the engineering tool, and
- To integrate the traction drive selection and design software in an overall HEV system simulation program.

Appendix I

PAPER A

EL-Hinaoui-Hamze, H and Hamdi, E S, 'Development and implementation of an interactive traction motors design and selection software', WSEAS, Vol. 2, No. 3, pp.539-542, 2003.

Development and Implementation of an Interactive Traction Motors Design and Selection Software

HASAN EL HINAOUI HAMZE and ESSAM HAMDI
Department of Electrical Power Engineering
Chalmers University of Technology
Gothenburg 412 96
SWEDEN
hasan.el.hinaoui@elteknik.chalmers.se essam.hamdi@elteknik.chalmers.se
http://www.chalmers.se

Abstract: - This paper describes a computer-based design tool aimed to assist the vehicle designer to determine whether or not drive requirements can be met within available space envelope, and to decide on the most suitable system topology to be employed. Normalised curves giving achievable force density values for different cooling arrangements are utilised, after adopting piecewise linear approximations, in the developed software.

Key-Words: - Hybrid Vehicles, Electric Drives, CAD, Traction Motors

1 Introduction

Use of electric motors in vehicle traction, in fact, predates the internal combustion engines (ICEs). Electric vehicles powered by dc motors were known as far back as 1890. Compared with vehicles powered by ICEs, the main drawback of the early electric vehicles was, and still is, their limited range and long recharging time. To overcome this, in recent years, developments have been focused on hybrid electrical vehicles (HEVs) combining two sources of energy: an ICE of a conventional vehicle and an electric motor (or motors). Such a hybrid vehicle enables the driver (or the vehicle computerised energy management system) to decide which source of power is appropriate for a particular journey.

Drive Trains for HEVs are of course more complex than both conventional ICEs and totally electric vehicles. Also, the energy management system is of vital importance. Design of HEVs, therefore, encompasses several technologies and the vehicle designer cannot be expected to be an expert in all related fields. In Sweden, it is recognised that vehicle designers may not be familiar with electric drive systems and the work reported here is commissioned by a consortium of interested parties (including Volvo and Saab) to provide a design tool to enable decisions to be made on type and feasibility of traction motors to meet specific design requirements.

2 Software Description

The search input to the CAD package would be drive requirements such as torque, power, efficiency and speed range. The output should be a recommendation for a specific motor type, size, number of poles, and field weakening range. The developed software therefore consists of three main parts as illustrated in Fig. 1.

Figure 1: Illustration of the developed software

The flowchart of Fig. 2 illustrates the functional organisation of the interactive CAD facility.

The first step in the development is to decide on the programming language and data storage and search tools. Visual Basic is a programming language concept in which a program is viewed as collection of discrete objects that are self-contained collections of data structures and routines that interact with other objects. It is an Object-Oriented Programming (OOP) Language in which a variable comprising both routines and data is treated as a discrete entity. As Visual Basic emulates the human way of seeing things, it is therefore adopted in programming the user-friendly interface as well as to perform computational steps within the software.

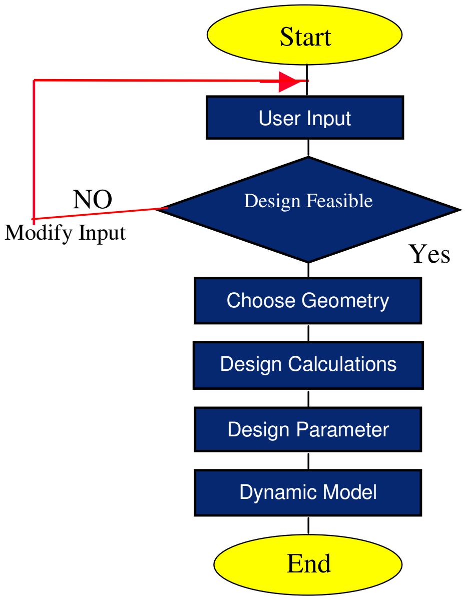


Figure 2: Organisation of CAD Package

Structured Query Language (SQL) can perform the search operation for specific information within a huge database in a fraction of second, regardless the database type. SQL can also be embedded within a Visual Basic medium. As the developed software makes use of expert drive system design knowledge, which is stored on the form of a database, SQL is utilised to perform the data search. The data itself is stored using Access.

The software developed here can be divided into three main parts, as shown in Fig. 3. If we take into account that VB, SQL, and Access all of these products are fully supported by Microsoft, as indeed is the target operating system (Windows), and this should eliminates any incompatibility problems.

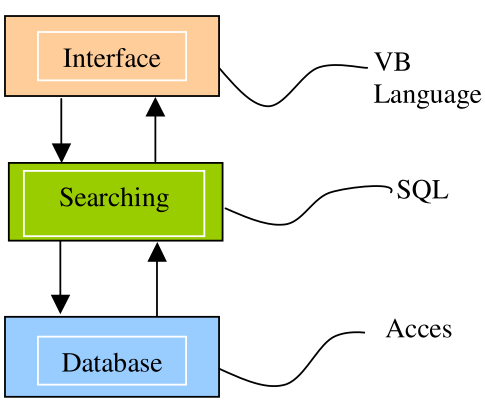


Figure 3: Integration of VB, SQL and Access in the developed software

3 Feasibility of Drive Requirements and Design

Rather than perform detailed and lengthy design calculations and find out that the space available in the vehicle system is insufficient to accommodate the electric drive system, the first stage of the CAD package examines whether or not the traction requirements, in terms of required force density, are theoretically possible. As a minimum, the vehicle designer specifies the system voltage, space envelope available, cooling method, ambient or inlet coolant temperature and rated torque and speed. From this an assumption is made regarding the likely pole numbers and the air-gap diameter is, roughly, estimated (as a percentage of the outer or envelope diameter). The required force density and pole-pitch are therefore calculated and an armature winding current density value is assumed following established design roles [1,2].

In the database section of the software, it is stored theoretical force density values for a range of current densities and pole-pitches [3]. Figure 4 shows one such family of data for a given pole-pitch [3]. Such a data is stored utilising a piecewise linear approximation. From experience, the practical force density limits are expected to be about 50% of the theoretical limits [2,3]. Using the assumed pole-pitch and current density values as input to the search process, the appropriate limits are found and an indication of the relative location of the drive requirements with respect to the theoretical and/or practical limits is returned to the user, as illustrated in Fig. 5. The user can opt to either modify the requirements or proceed with the design process, as appropriate.

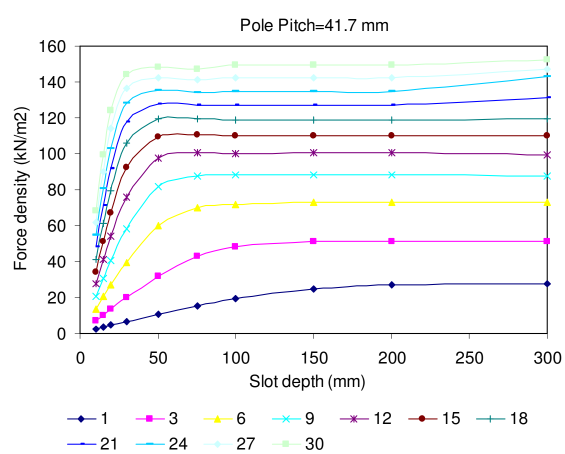


Figure 4: Typical force values for different winding current densities

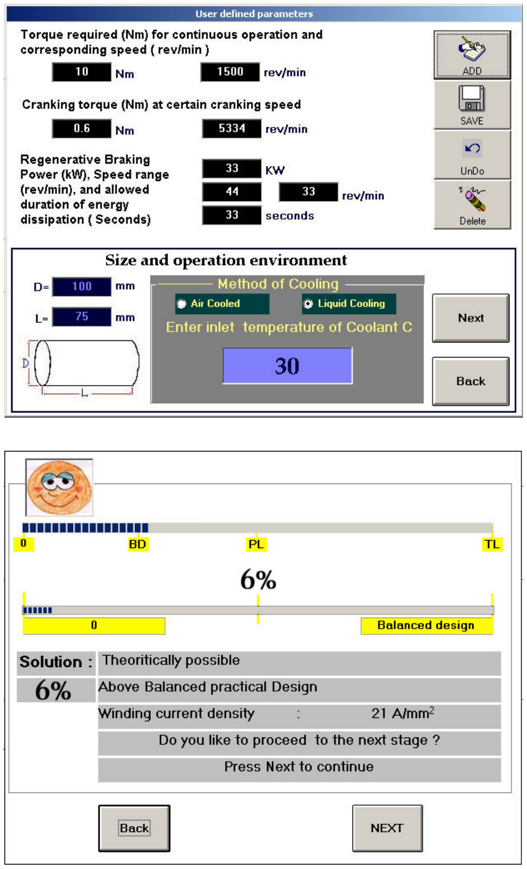


Figure 5: User input (top) and outcome of the feasibility study

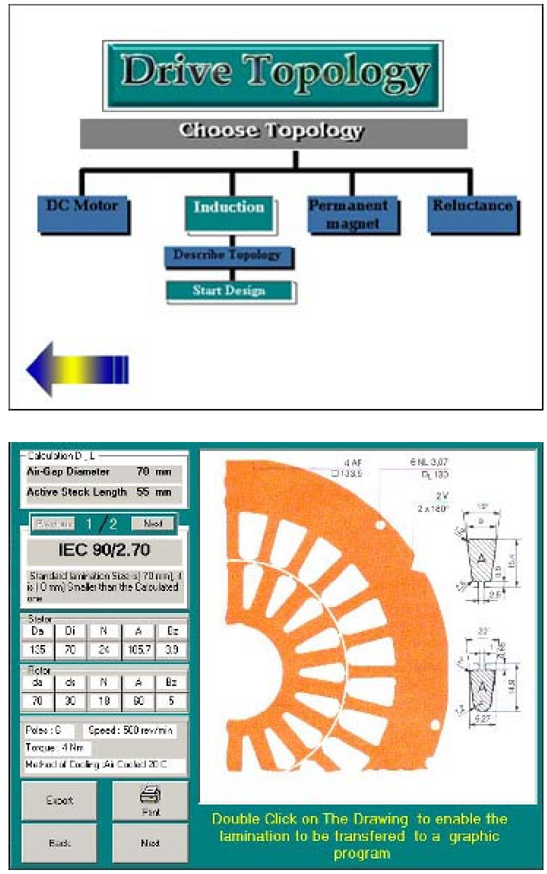


Figure 6: Drive topology selection (top) and search output for suitable lamination stamp

After establishing that the drive requirements are feasible and indicating that they wish to proceed to the next stage, the vehicle designers are asked to choose a drive topology to be considered. To assist the decision, salient features of each topology are provided, as illustrated in Fig. 6 (a). The calculations follow established design methodology and roles [1,2] and after calculating the main dimensions a database containing hundreds of standard lamination stamping is searched to identify suitable laminations. Fig. 6 (b) shows the results of such a search when an induction motor was chosen. The search results also give the slot dimensions, from which permeance coefficients are calculated as illustrated in Fig. 7. A suitable winding is then designed and its per-phase resistance and the slot leakage inductances are determined. The design process is then completed as described in [1,2] and the equivalent circuit as well as other relevant design parameters are returned.

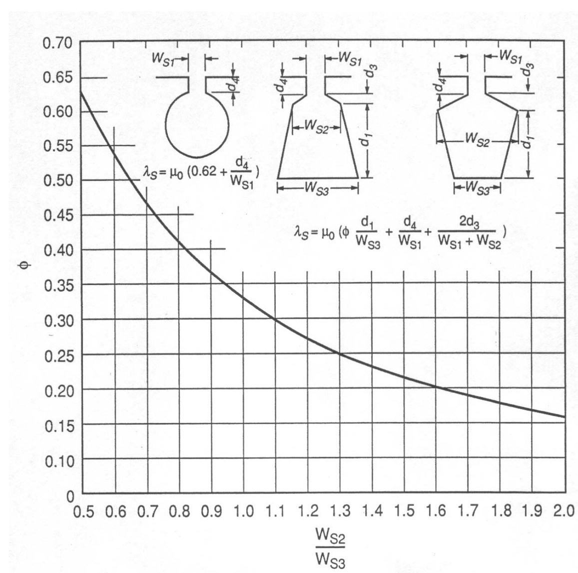


Figure 7: Determination of permeance coefficient

4 Conclusions

A design database for traction motors intended for hybrid electrical vehicles (HEVs), for which the search input would be properties such as torque, power, space envelope available and speed range, and the output is, for a specific motor type, size, number of poles, lamination details, winding layout and circuit parameters, is developed and presented.

The purpose of this database is to support the vehicle designer in the selection of suitable traction motor type. Preliminary testing of the developed software has been very encouraging and warrants further development and extension of its scope. The main line of extending the software is marrying it to a simulation model for each traction motor type. The simulation model will represent the static and transient torque response of a specific motor type, taking into account limitations introduced by the traction battery voltage, switching frequency in the power electronics, temperature and motor parameters. The time resolution of the model will be shorter than 0.1 seconds. It is hoped to report on this in due course.

References:

[1] Hamdi, E S: *Design of small electric machines*, John Wiley & Sons, 1994, 1996, 1998.

[2] Hamdi, E S: *Permanent magnet and variable reluctance drive systems*, ETI Press, 2003.

[3] Kasinathan, P: *Integrated in-wheel motors for low-power traction applications*, PhD thesis, Chalmers University of Technology, Gothenburg, Sweden, 2003.

Appendix I

PAPER B

EL-Hinaoui-Hamze, H and Hamdi, E S, ' Steady State and Transient Performance of HEV-CAD Tool', Norfa Summer Seminar Proc, June 4-6 2004, Tallinn, Estonia, pp.81-85.

Slot Permeance Coefficient Steady State and Transient Performance of HEV- CAD Tool

HASAN EL HINAOUI HAMZE and ESSAM HAMDI
Electrical Machines and Drive Systems
Chalmers University of Technology
Gothenburg 412 96
SWEDEN
hasan.el.hinaoui@elteknik.chalmers.se essam.hamdi@elteknik.chalmers.se
www.elteknik.chalmers.se/~EMDS/EMDS/index.htm

Abstract: - It was reported earlier how normalised curves, giving achievable force density values for different cooling arrangements, are utilised to assist HEV designers to determine whether or not drive requirements can be met within available space envelope [1]. This paper describes further development of the design tool. Analytical design methods are employed to determine design details and after a full electromagnetic design is obtained, an interface is created with MATLAB to determine transient performance.

Key-Words: - Hybrid Electric Vehicles, Electric Drives, CAD, Lamination.

1. Introduction

Hybrid electric vehicles (HEVs) have been found to overcome the two main drawbacks of electric cars; namely the limited range and long recharging time. Therefore, a HEV consists of two sources of mechanical energy: an electric motor powered by a battery (as is the case in an electric vehicle), and an internal combustion engine (ICE) of a conventional vehicle. Depending on the drive shaft arrangement and the energy management scheme, HEVs can be classified into three types as described below.

1.1 Hybrid Vehicle - Series Type:

In a series HEV, only the electric motor is connected directly to wheels. The engine-generator set is only operated to generate electric power, which is stored in a battery, which, in turn, feeds the electric motor. Fig. 1 illustrates a series hybrid drive train arrangement. For the purpose of the illustration, it is shown an electric motor on the drive shaft. Of course this is not the only possible arrangement, for example, the motor could be incorporated or integrated with the wheel. Examples of one topology of permanent magnet in-wheel motors can be found in [2]. Similar argument applies to Figs. 2 and 3 below, for the parallel and the dual-mode hybrids; respectively.

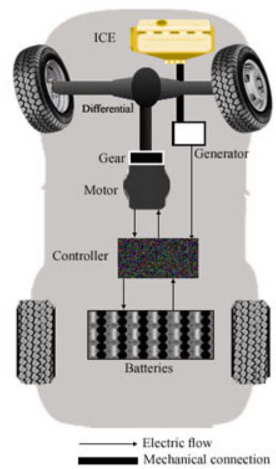


Figure 1: Series HEV.

1.2 Hybrid Vehicle - Parallel Type:

In a parallel HEV, the wheels can be driven directly by both the electric motor and the ICE simultaneously, to obtain acceleration boost. Normal operation would involve only the electric motor in built up areas. In out of town driving on longer journeys, the ICE provides the traction power as well as driving the motor to operate as generator to charge the batteries. Therefore, HEVs of the parallel type do not need a dedicated generator, since the electric motor is used as generator to recharge the batteries, as shown in Fig 2.

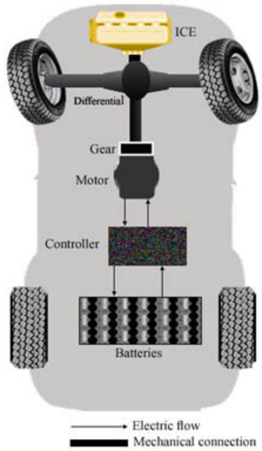


Figure 2: Parallel HEV.

1.3 Hybrid Vehicle-Dual or Mixed Mode:
As the name implies, mixed hybrids are a blend of the two configurations discussed above. This type combines the desirable features of both series and parallel hybrids. A dual-mode HEV can be driven directly by either the electric motor or the ICE and from both in some conditions to provide the vehicle with maximum traction power. ICE is used to charge the batteries through a generator, as the electric motor can be used as generator to recharge batteries (regenerative braking), as shown in Fig 3.

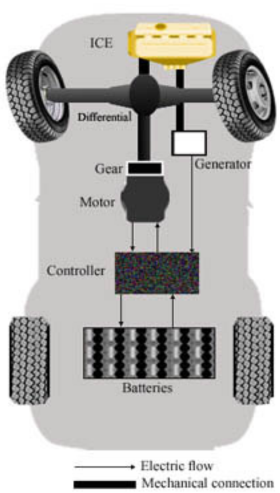


Figure 3: Dual-mode HEV.

2. Aim of the Present Work

HEVs are of course more complex than both conventional and electric vehicles. Their design involves ICE, electric motor, energy management system, vehicle aerodynamics, and control electronics. Therefore, design of HEVs encompasses several technologies, and the vehicle designer cannot be expected to be expert in all related fields.

For decades vehicle designers have used ICEs and therefore they have developed a feel regarding what size engine can be housed in a given space. Most designers, however, have little or no appreciation regarding what space would be required to accommodate an electric motor capable of producing a certain output. In addition, the issue of the relative merits of various electric drive topologies and their suitability for integration with the conventional drive train mechanisms is rather complicated.

The aim of the work reported here is to provide a design tool to enable decisions to be made on type and feasibility of traction motors to meet specific design requirements and to provide steady state and transient performance of the chosen design.

3. Description of the CAD Tool

The modular structure and the overall organisation of the developed software are described in [1]. The calculations are divided into three distinct stages and upon the completion of each stage the user decides whether or not to proceed to the next.

Rather than perform detailed and lengthy design calculations and find out that the space available in the vehicle system is insufficient to accommodate the electric drive system, the first stage of the CAD package examines whether or not the traction requirements, in terms of required force density, are theoretically possible.

The second stage commences by the user choosing a particular drive topology to be fully investigated. Thereafter, conventional design procedures are followed until a full electromagnetic design is obtained and its steady state parameters are derived. Invisible to the user, the software stores the design details in a suitable format to enable execution of the third stage of evaluation.

In the third stage, the software developed here is married to a MATLAB environment, where the drive systems transient performance, such as acceleration capability, overload capacity and controllability, are investigated.

3.1 First Stage of the Computation:

The start up menu requires the inputs shown in Fig. 4 to be provided by the user.

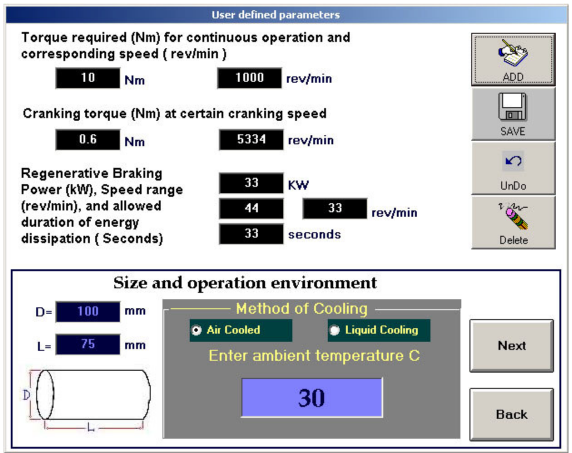


Figure 4: User input

Speed & Envelope Space

The computation utilises the specified drive base speed (N, rev/min) to determine a preliminary number of poles (p), which is, together with the diameter of envelope available (D, mm) are used to calculate the pole-pitch (τ, mm) as follows:

$$\tau = \frac{0.56 D \pi}{p}$$

As it is shown in the equation above, the air-gap diameter has been assumed as 0.56 of the user defined envelope diameter (to allow for stator slot or magnet and core depths and thickness of housing frame). This value is based on experience and can easily be verified by examining laminations catalogue.

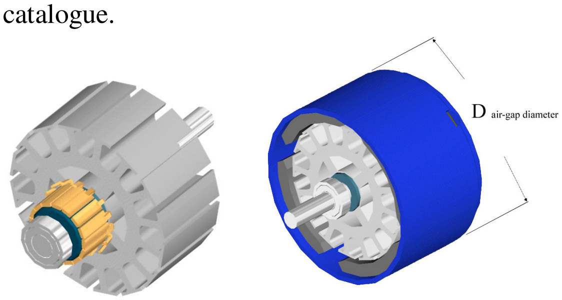


Figure 5: Air-gap diameter.

In the database it is stored sets of normalised available force density curves, obtained according to the work of Kasinathan [2]. The calculated pole-pitch (τ) points to the appropriate data set in the database.

Method of Cooling

The specified method of cooling is used to decide winding current density, which in turn is used to determine which curve in the family is to be used to determine the force density limits [2]. Each curve is divided into two main regions. The first shows an increase of obtainable force density as the slot depth (and machine size) is increased (and is approximated by a straight-line equation on the form $y = mx$). The second Region (saturated part of the curve) indicates no farther increase of force density is obtainable. This region is approximated by a straight-line equation on the form $y = C$.

The theoretical maximum value of the force density available is determined by the intersection of the two regions. This value is denoted theoretical limit (TL). The practical (PL) and balanced practical design (BPD) points are determined from design experience as half and 25% of the theoretical limit. The second Region (saturated part of the curve) indicates no farther increase of force density is obtainable.

Required Force Density

Computation utilises the continuous torque (T Nm), the assumed initial air-gap diameter and available length of the space envelope (L, mm) to estimate a value of the force density required (kN/m^2).

Feasibility of the Design

After calculating the force density, the users will be provided with an indication regarding the feasibility of their design as shown in Fig 6. Also provided is information regarding how practical the requirements are.

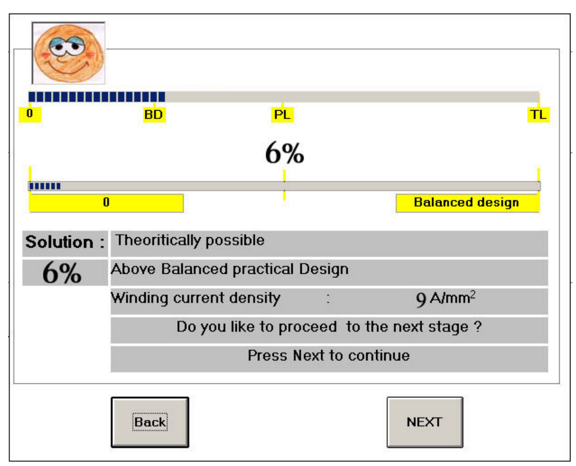


Figure 6: Screen shot of outcome of feasibility study

3.2 Second Stage of the Computation:

This stage follows conventional design procedure [3,4]. In case of induction motors, values of power factor and efficiency, magnetic and electric loadings are determined according to rated steady state output power and cooling method to be employed. The machine's kVA input is then calculated and air-gap D^2L is determined from a standard output equation [3]. Suitable choice of the ratio of stack length/pole-pitch (to control end-winding leakage) enables separation of the motors air-gap diameter and length. Using the air-gap diameter, a standard lamination database is searched for the nearest available one (see Fig. 7).

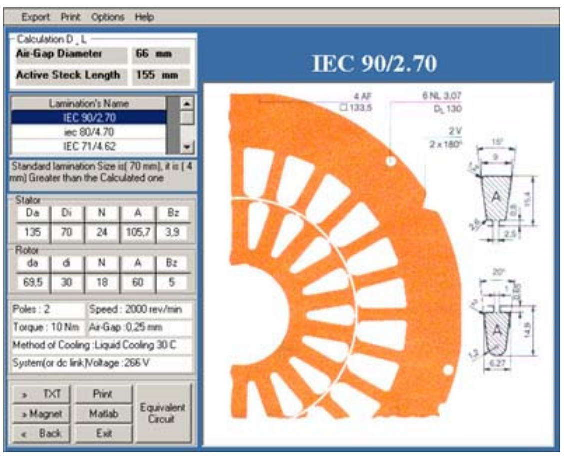


Figure 7: Induction motor laminations.

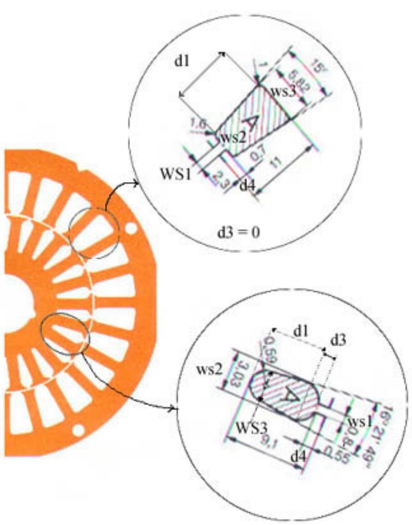


Figure 8: Slot dimensions.

The search results also give the slot dimensions, from which permeance coefficients, and subsequently slot leakage inductances, are calculated [3,4]. Fig. 8 illustrates the slot dimensions for an induction motor solution.

The system voltage is then considered together with the topology of the power electronics converter and the winding phase voltage (for the fundamental component) is estimated. Allowance is made for the stator impedance drop and the winding emf is determined. There follows computation of the number of turns in series per phase and a suitable winding is designed to match the chosen laminations [3].

Rotor design is performed assuming an aluminium cast cage (for the induction motor case). Various components of leakage flux are then computed as well as the winding and cage resistance.

This stage concludes by offering the user the options to examine the equivalent circuit (parameters calculated at fundamental frequency), and export the lamination details to a commercial drawing tool (AntoCad) or to a commercial finite element suite of programmes (MagNet). An equivalent circuit for the relative design will be done based on conventional design procedure [3,4], as shown in Fig 9.

The software also stores essential drive details in text and *m* format. This enables the user to investigate transient and short-term performance of the drive system in MATLAB environment.

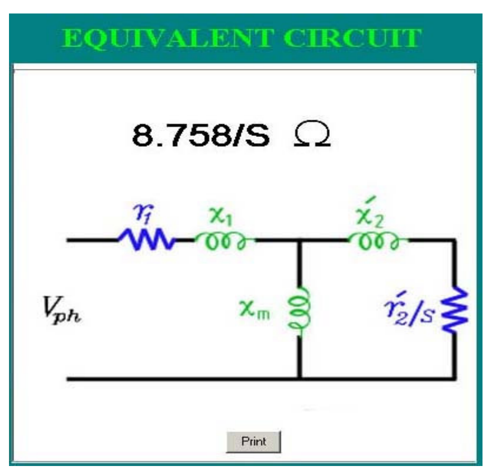


Figure 9: Equivalent circuit.

3.3 Third Stage of the Computation:

As stated above, after the second stage of computation, the CAD package developed at Chalmers University of Technology is married to the commercial MATLAB suite of programmes. This part of the work is carried out at Lund Technical University. An example of this stage's output is shown in Fig. 10.

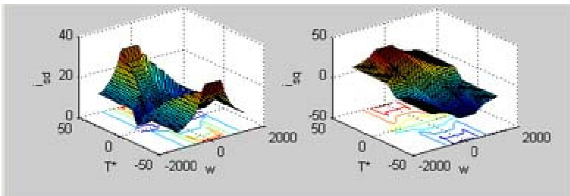


Figure 10: Variation of reference torque with current

4. Conclusions

A design database for traction motors intended for HEVs, for which the search input would be properties such as torque, power, space envelope available and speed range, and the output is, for a specific motor type, size, number of poles, lamination details, winding layout and circuit parameters, is developed and presented.

The purpose of this database is to support the vehicle designer in the selection of suitable traction motor type. Preliminary testing of the developed software has been very encouraging and warrants further development and extension of its scope. The main line of extending the software is marrying it to a simulation model for each traction motor type. The simulation model will represent the static and transient torque response of a specific motor type, taking into account limitations introduced by the traction battery voltage, switching frequency in the power electronics, temperature and motor parameters.

References:

[1] EL-Hinaoui-Hamze, H and Hamdi, E S: *Development and implementation of an interactive traction motors design and selection software*, WSEAS Transactions on Systems, Vol. 2, No. 3, pp.539-542, 2003.

[2] Kasinathan, P: *Integrated in-wheel motors for low-power traction applications*, PhD thesis, Chalmers University of Technology, Gothenburg, Sweden, 2003.

[3] Hamdi, E S: *Design of small electric machines*, John Wiley & Sons, 1994, 1996, 1998.

[4] Hamdi, E S: *Permanent magnet and variable reluctance drive systems*, ETI Press, 2003.

[5] Bates, B: *Electric and Hybrid Electric Vehicle Technology,* Society of Automotive Engineers, 1994.

[6] Hodkinson R and Fenton J: *Lightweight electric/Hybrid Vehicle Design,* Ltd Press, 2001.

[7] Cllie M.J: *electric and Hybrid Vehicle,* NOYES DATA CORPORATION, 1979.

Appendix II

MagNet & AutoCAD Procedure

Introduction:

As mentioned in the thesis, MagNet is a commercial finite element suite of programmes [20]. It is well known to engineers and scientists involved with field computations using numerical field solution that building the problem geometry is the most laborious process. This is particularly true for geometries like those of motor laminations. As, in the developed software the lamination details are already stored in the database, this represents an opportunity to automate the drawing of lamination.

An interface is provided between the design tool database and MagNet and AutoCAD. By clicking the option “MagNet” (Figure 2.18 or Figure 2.21), the lamination details (which are stored in the database) are used in connection with the following VB subroutine to draw the motor magnetic circuits as shown in Fig. A2.1

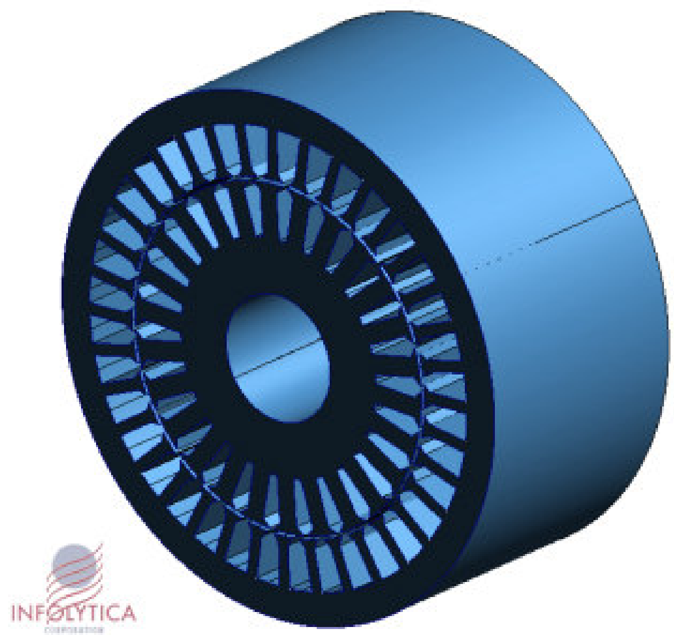


Figure A2.1

Magnet & Autocad procedures:

1-Run Magnet

```
'--------------------start magnet
StartMagnetTrue 'to run Magnet
```

Sub StartMagnetTrue()

```
   Set MN6 = CreateObject("magnet.application")
   Set Dev = MN6.NewDocument
   Set Con = MN6.GetConstants

   Selection = Con.infoSetSelection

   Surface = Con.infoSliceSurface

   MN6.Visible = True

   Call Dev.setDefaultLengthUnit("Millimeters")
   Call Dev.setScaledToFit(True)
```

```
Call Dev.setSnapMode(Con.infoSnapPerpendicular)
```

End Sub

2-Rotor Parameters

```
' get the values of the Rotor    (Lstk,R1,Ns,SO ,Trooth,tgh,Tw,SH,Sb)

    Lstk = 65
    R1 = Val(0.5 * (Val(Label91.Caption) - AirGapValue)) '(da/2) - 0.25
    Ns = Val(Label79.Caption)             'N
    SO = Val(Label80.Caption)             'Ws1
    Trooth = Val(Label85.Caption)         'd4
    tgh = Val(Label84.Caption)            'd3
    Tw = Val(Label89.Caption)             'bz = tooth width
    SH = Val(Label83.Caption)             'd1
    Sb = R1 - ((0.5 * Val(Label90.Caption)) + Val(Label87.Caption))
```

CallRotor ' draw the Rotor

3- Drawing the Rotor lamination

Sub CallRotor()

```
   R2 = R1 – Trooth

   SCinner = 2 * Pi * R1  'Stator core inner Circumference

   TotalSOD = SO * Ns     'Distance occupied by slot opening

           TotalToothD = SCinner - TotalSOD    'Distance occupied by Tooth
           EachToothD = TotalToothD / Ns
           a1 = (360 / SCinner) * EachToothD  'Tooth angle
           a2 = a1 / 2

   SOC = SO / 2

           A3 = (360 / SCinner) * SO
           A4 = A3 / 2
```

'Point1

X1 = 0
Y1 = R1

'Point2

X2 = R1 * Sin(a2 * Pi / 180)
Y2 = R1 * Cos(a2 * Pi / 180)

'Point3

y3 = R2 * Cos(a2 * Pi / 180)
x3 = R2 * Sin(a2 * Pi / 180)

'Point4

x4 = Tw / 2

```
If tgh = 0 Then
y4 = y3
Else
y4 = R2 - tgh   'tgh is tang hight
End If
```

'Point5

x5 = x4
y5 = y4 - SH

'Slot opening

R3 = y5

A5 = a2 + A4 'important angle
y6 = R3 * Cos(A5 * Pi / 180)
x6 = R3 * Sin(A5 * Pi / 180)

'Stator outer
R4 = y5 - Sb

```
   Call Dev.newConstructionSliceArc(0, 0, X2, Y2, X1, Y1)  'newArc (x center, y center,
x start, y start, x end, y end)

   Call Dev.newConstructionSliceLine(X2, Y2, x3, y3)
   Call Dev.newConstructionSliceLine(x3, y3, x4, y4)
   Call Dev.newConstructionSliceLine(x4, y4, x5, y5)
   Call Dev.newConstructionSliceArc(0, 0, x6, y6, x5, y5)


   'Mirror

   Call Dev.getCurrentView.SelectAll(infoSetSelection)
   Call Dev.getCurrentView.mirrorSelectedEdges(0, 0, 0, 1, Con.infoTrue)
   Call Dev.getCurrentView.mirrorSelectedEdges(0, 0, 0, 1, Con.infoTrue)

                 'This section rotates each slot using the for loop

                     k = Ns
                     t = 360 / k
                     W = k - 1

                  Call Dev.getCurrentView.SelectAll(infoSetSelection)

                  For Count1 = 1 To W

                  d = t * Count1
                  Call Dev.getCurrentView.rotateSelectedEdges(0, 0, d, Con.infoTrue)

                  Next

                  'Outer circle
                  Call Dev.newConstructionSliceCircle(0, 0, R4)


   'Asigns material to stator parts of model'

   xtc = 0   'Tooth center
   ytc = y4

   ArrayOfValues(0) = infoSliceSurface

   Call Dev.selectAtWithObjectCode(xtc, ytc, Con.infoSetSelection,
Con.infoSliceSurface)

   ArrayOfValues(0) = "RotorLamination"
```

```
   Call Dev.getCurrentView.makeComponentInALine(Lstk, ArrayOfValues,
"Name=CR10: Cold rolled 1010 steel", infoTrue)

'   Call Dev.getCurrentView.exportDXF("C:\Lamination
DRAWING\LaminationRotor1.dxf") 'For total layout

   Call MN6.saveDocument(App.Path + "\Laminationdrawing\RotorPair.mn")
'   Call Dev.getCurrentView.exportDXF(App.Path +
"\Laminationdrawing\RotorOnly.dxf")

   Call MN6.CloseDocument
   Call MN6.Exit

   Set MN6 = Nothing

'   Call Dev.getCurrentView.exportDXF("C:\Green
Car\Laminationdrawing\RotorOnly.dxf")


End Sub
```

4-Stator Parameters

```
' get the values of the Stator    (Lstk,R1,Ns,SO ,Trooth,tgh,Tw,SH,Sb)

    R1 = Val(0.5 * Val(Label76.Caption))  '0.5 di
    Ns = Val(Label65.Caption)            'N
    SO = Val(Label66.Caption)            'Ws1
    Trooth = Val(Label71.Caption)         'd4
    tgh = Val(Label70.Caption)           'd3
    Tw = Val(Label75.Caption)            'bz = tooth width
    SH = Val(Label69.Caption)            'd1
    Sb = (0.5 * Val(Label77.Caption)) - (R1 + Val(Label73.Caption)) ' the width of the
stator

CallStator ' draw the Stator
```

5- Drawing the Stator lamination

Sub CallStator()

'This section creates the x and y coordinates of the overall problem'

```
'   Pi = 3.141592
    R2 = R1 + Trooth
    SCinner = 2 * Pi * R1  'Stator core inner Circumference
    TotalSOD = SO * Ns     'Distance occupied by slot oppening
            TotalToothD = SCinner - TotalSOD    'Distance occupied by Tooth
            EachToothD = TotalToothD / Ns
            a1 = (360 / SCinner) * EachToothD  'Tooth angle
            a2 = a1 / 2

    SOC = SO / 2
            A3 = (360 / SCinner) * SO
            A4 = A3 / 2


    'Point1

    X1 = 0
    Y1 = R1


    'Point2

    X2 = R1 * Sin(a2 * Pi / 180)
    Y2 = R1 * Cos(a2 * Pi / 180)


    'Point3

    y3 = R2 * Cos(a2 * Pi / 180)
    x3 = R2 * Sin(a2 * Pi / 180)


    'Point4

    x4 = Tw / 2
    y4 = R2 + tgh   'tgh is tang hight


    'Point5

    x5 = x4
    y5 = y4 + SH
```

'Slot opening

R3 = y5

A5 = a2 + A4 'important angle

y6 = R3 * Cos(A5 * Pi / 180)

x6 = R3 * Sin(A5 * Pi / 180)

'Stator outer

R4 = y5 + Sb

Call Dev.newConstructionSliceArc(0, 0, X2, Y2, X1, Y1) 'newArc (x center, y center, x start, y start, x end, y end)

Call Dev.newConstructionSliceLine(X2, Y2, x3, y3)

Call Dev.newConstructionSliceLine(x3, y3, x4, y4)

Call Dev.newConstructionSliceLine(x4, y4, x5, y5)

Call Dev.newConstructionSliceArc(0, 0, x6, y6, x5, y5)

'Mirror

Call Dev.getCurrentView.SelectAll(infoSetSelection)

Call Dev.getCurrentView.mirrorSelectedEdges(0, 0, 0, 1, Con.infoTrue)

Call Dev.getCurrentView.mirrorSelectedEdges(0, 0, 0, 1, Con.infoTrue)

'This section rotates each slot using the for loop

k = Ns
t = 360 / k
W = k - 1

Call Dev.getCurrentView.SelectAll(infoSetSelection)

For Count1 = 1 To W

```
   d = t * Count1
   Call Dev.getCurrentView.rotateSelectedEdges(0, 0, d, Con.infoTrue)

   Next
```

'Outer circle

```
   Call Dev.newConstructionSliceCircle(0, 0, R4)


        'Asigns material to stator parts of model'

   xtc = 0   'Tooth center

   ytc = y4

   ArrayOfValues(0) = infoSliceSurface

   Call Dev.selectAtWithObjectCode(xtc, ytc, Con.infoSetSelection,
Con.infoSliceSurface)

   ArrayOfValues(0) = "StatorLamination"
```

' put material in magnet

```
   Call Dev.getCurrentView.makeComponentInALine(Lstk, ArrayOfValues,
"Name=CR10: Cold rolled 1010 steel", infoTrue)
```

'Importing rotor

```
   Call Dev.importModel(App.Path + "\Laminationdrawing\RotorPair.mn")

   Call Dev.getCurrentView.viewAll

   Call Dev.getCurrentView.SelectAll(infoSetSelection)

   ArrayOfValues(0) = "StatorLamination"

   Call Dev.getCurrentView.extractEdges(ArrayOfValues)

   ArrayOfValues(0) = "RotorPair#1,RotorLamination"

   Call Dev.getCurrentView.extractEdges(ArrayOfValues)
```

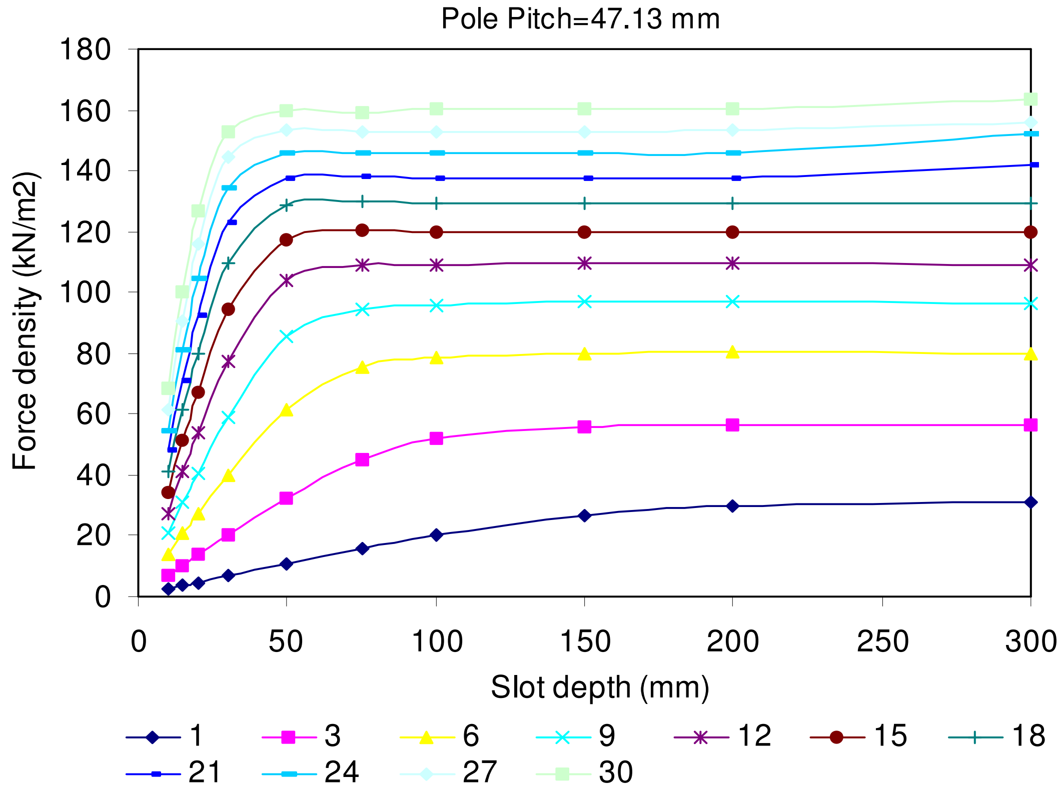
```
''Kill "C:\Green Car\drawing the lamination\Rotordrawing\RotorPair.mn"

''Call MN6.saveDocument("C:\Green Car\drawing the lamination\Rotordrawing\All.mn")

''Call Dev.getCurrentView.exportDXF("C:\Green Car\drawing the
lamination\Rotordrawing\all.dxf")

Mname = LaminationName + ".mn"

Aname = LaminationName + ".dxf"

Kill App.Path + "\Laminationdrawing\RotorPair.mn"

Call MN6.saveDocument(App.Path + "\Laminationdrawing\" + Mname)

Call Dev.getCurrentView.exportDXF(App.Path + "\Laminationdrawing\" + Aname)


'Kill App.Path + "\LaminationName"

End Sub
```

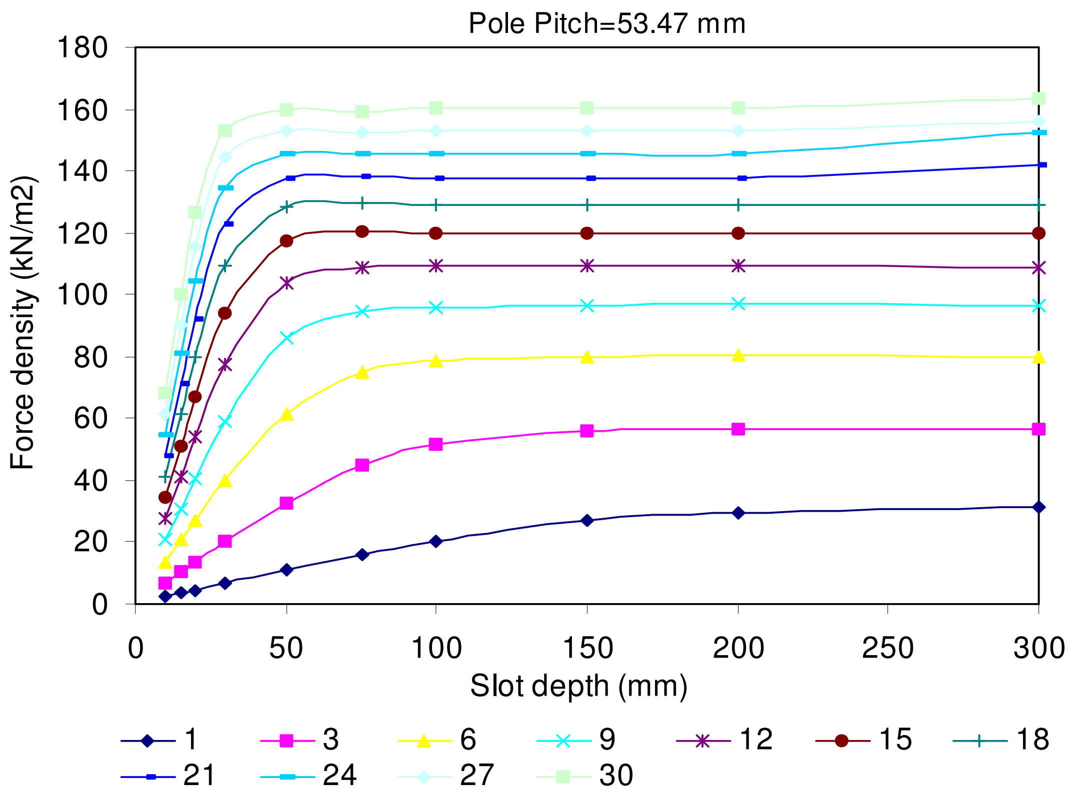
Appendix III

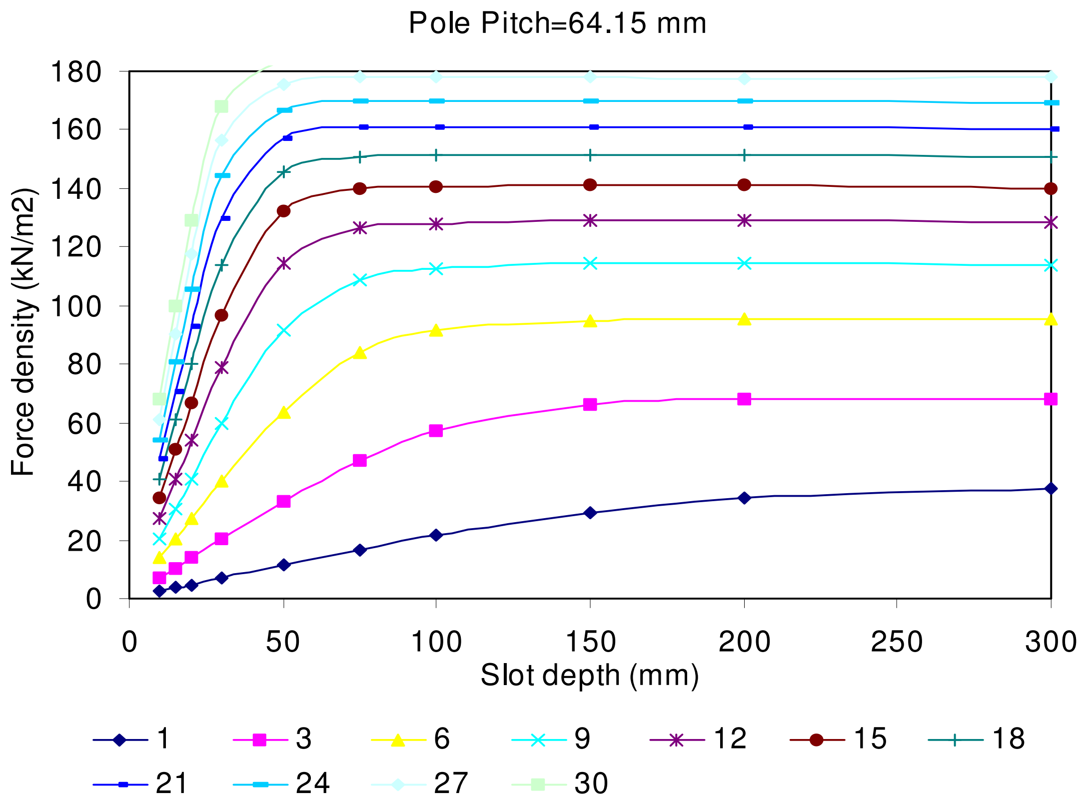
Normalised Force Density Curves

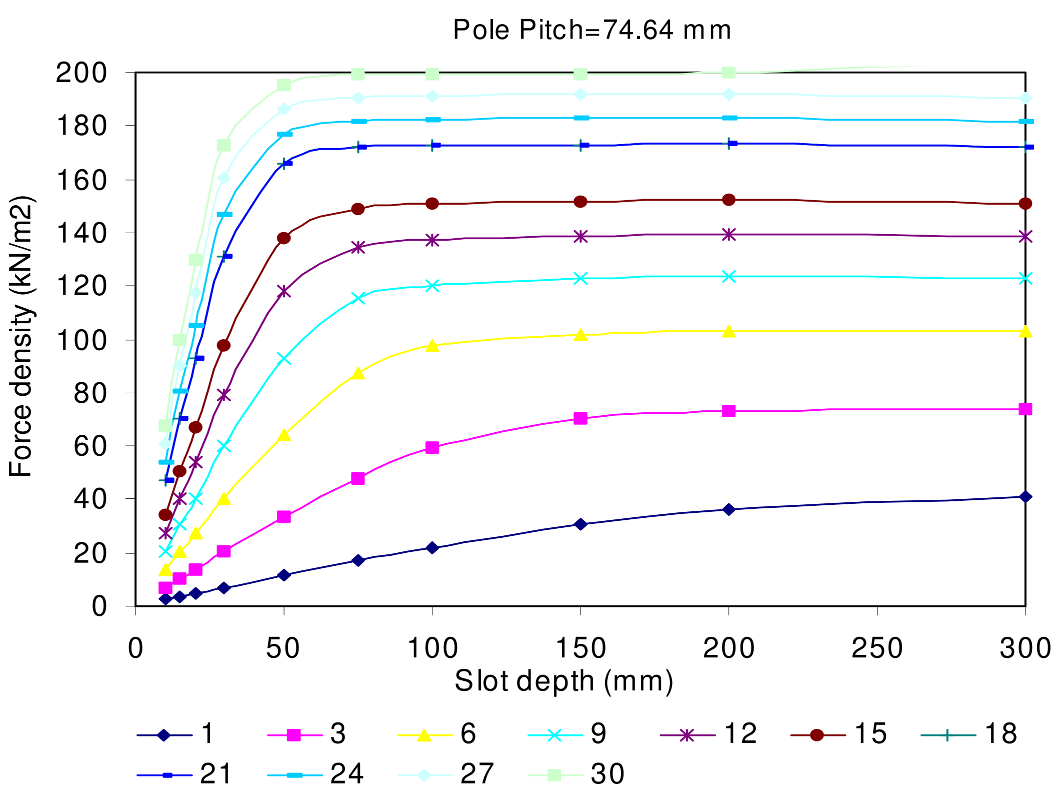
Introduction:

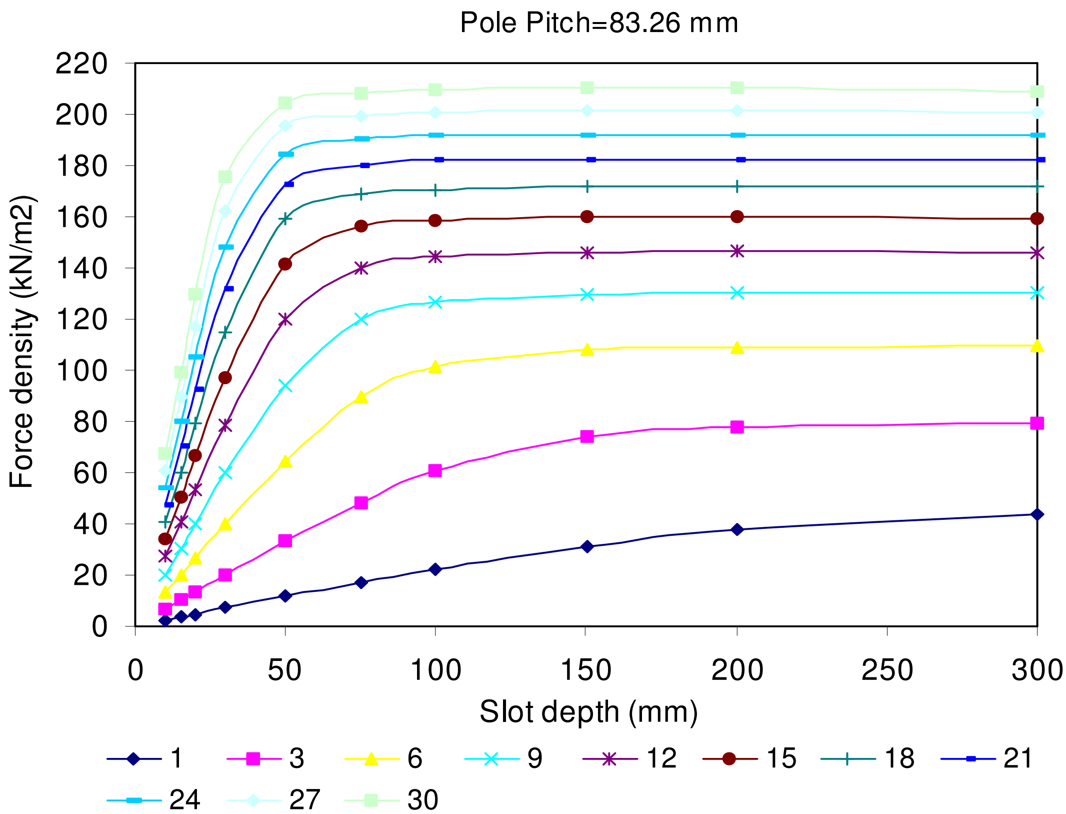
In the database it is stored sets of normalised available force density curves for deferent pole-pitch values [41.7 mm up to 213.47 mm], obtained according to the work of Kasinathan [3]. The calculated pole-pitch (τ) points to the appropriate data set in the database. The following curves are stored in the database. These are reproduced from reference [3].

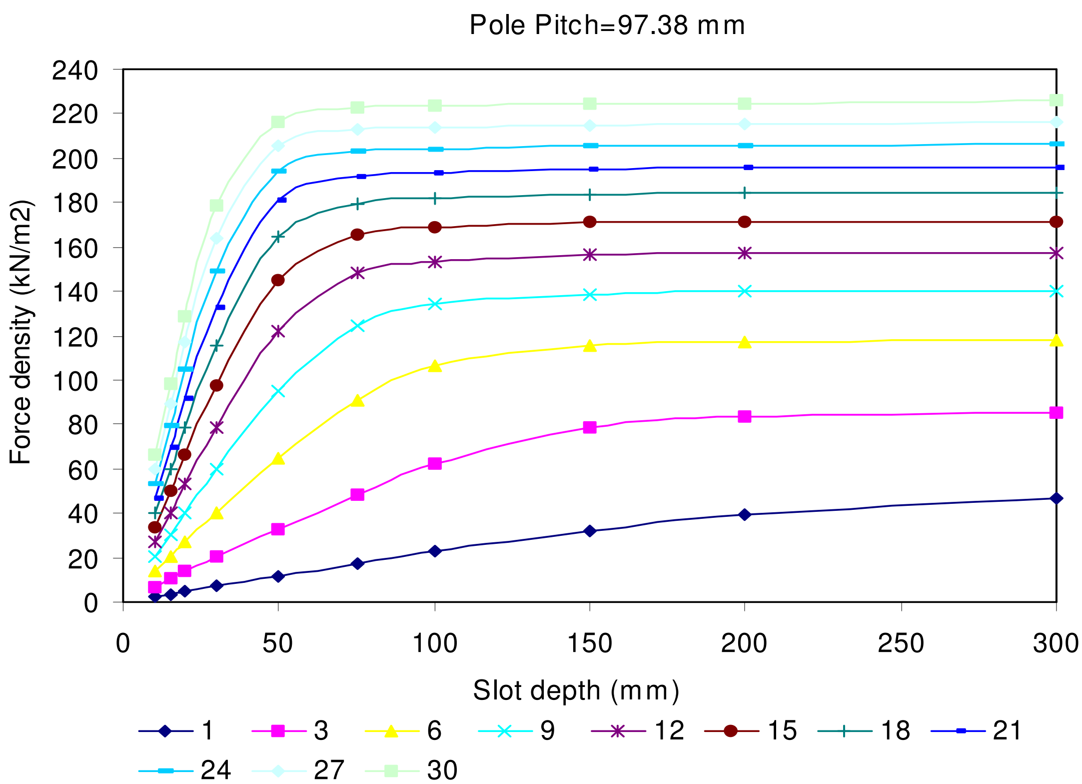

Pole Pitch=47.13 mm
Force density (kN/m2)
Slot depth (mm)
0
20
40
60
80
100
120
140
160
180
0
50
100
150
200
250
300
1
3
6
9
12
15
18
21
24
27
30

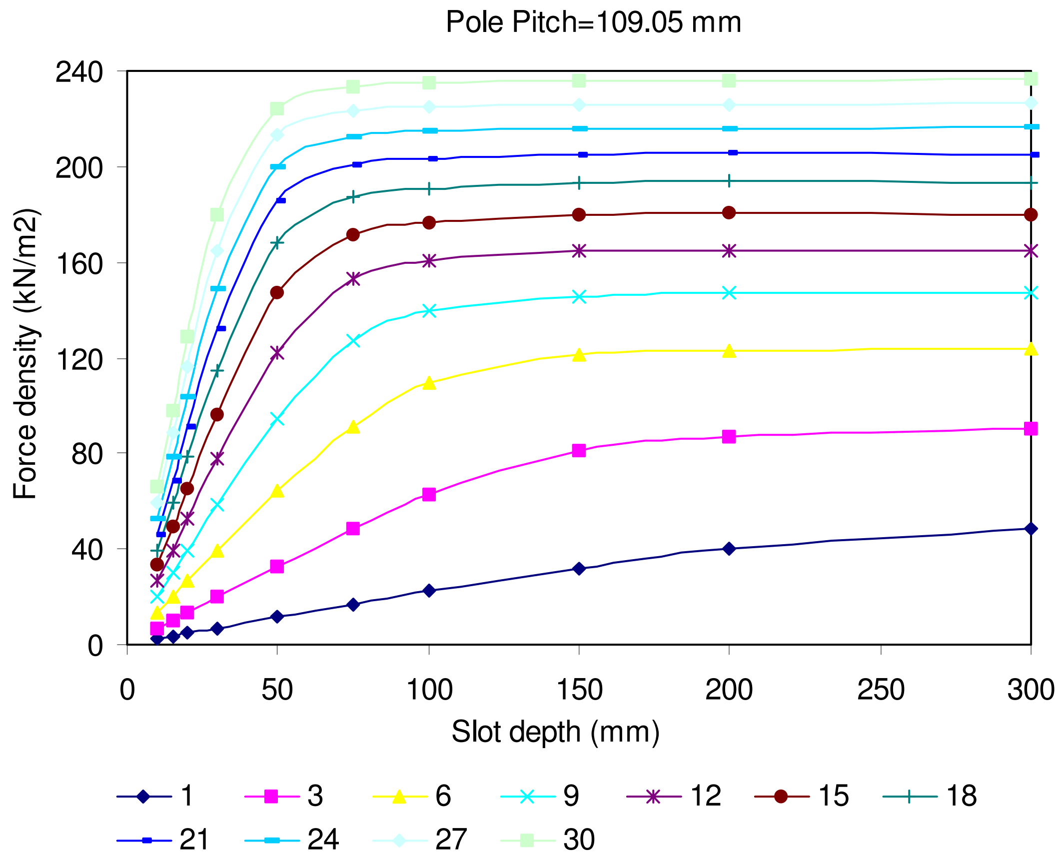

Pole Pitch=53.47 mm
Force density (kN/m2)
Slot depth (mm)
0
20
40
60
80
100
120
140
160
180
0
50
100
150
200
250
300
1
3
6
9
12
15
18
21
24
27
30

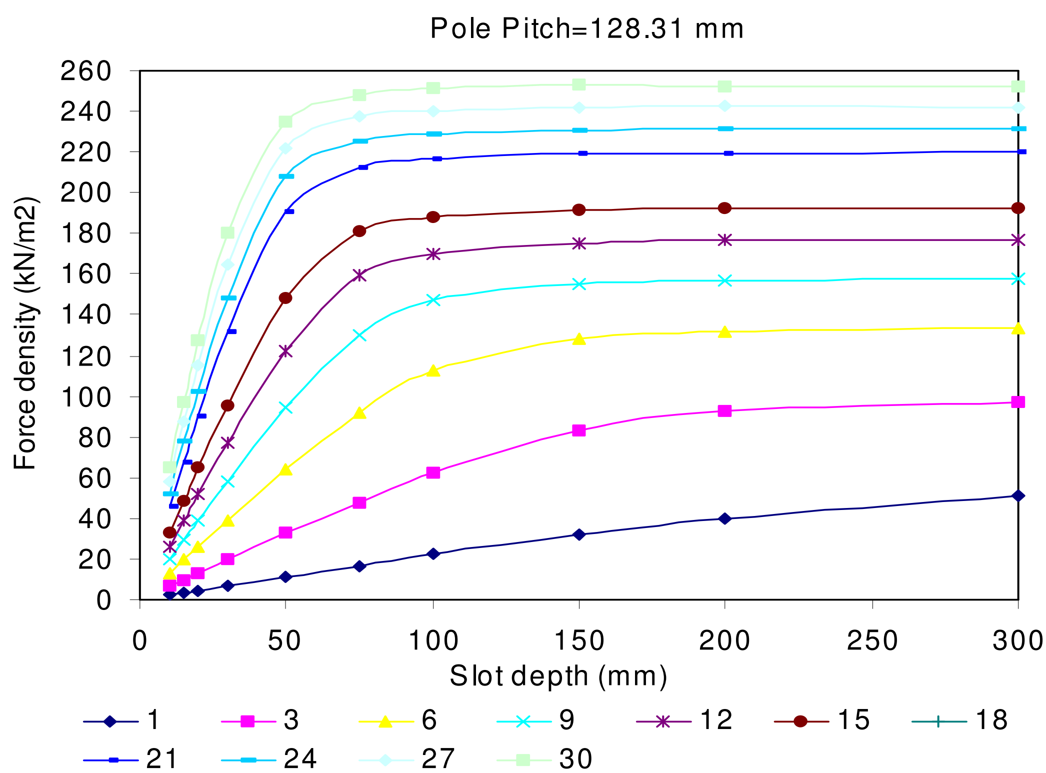

Pole Pitch=64.15 mm
Force density (kN/m2)
180
160
140
120
100
80
60
40
20
0
0
50
100
150
200
250
300
Slot depth (mm)
1
3
6
9
12
15
18
21
24
27
30

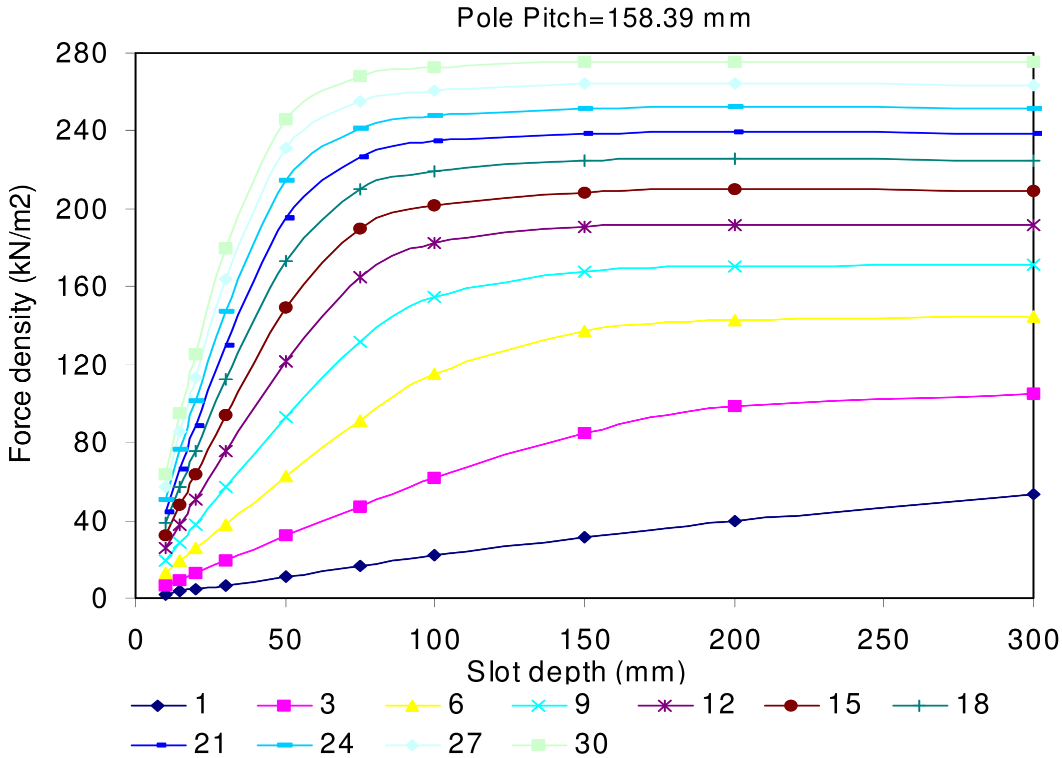

Pole Pitch=74.64 mm
Force density (kN/m2)
200
180
160
140
120
100
80
60
40
20
0
0
50
100
150
200
250
300
Slot depth (mm)
1
3
6
9
12
15
18
21
24
27
30

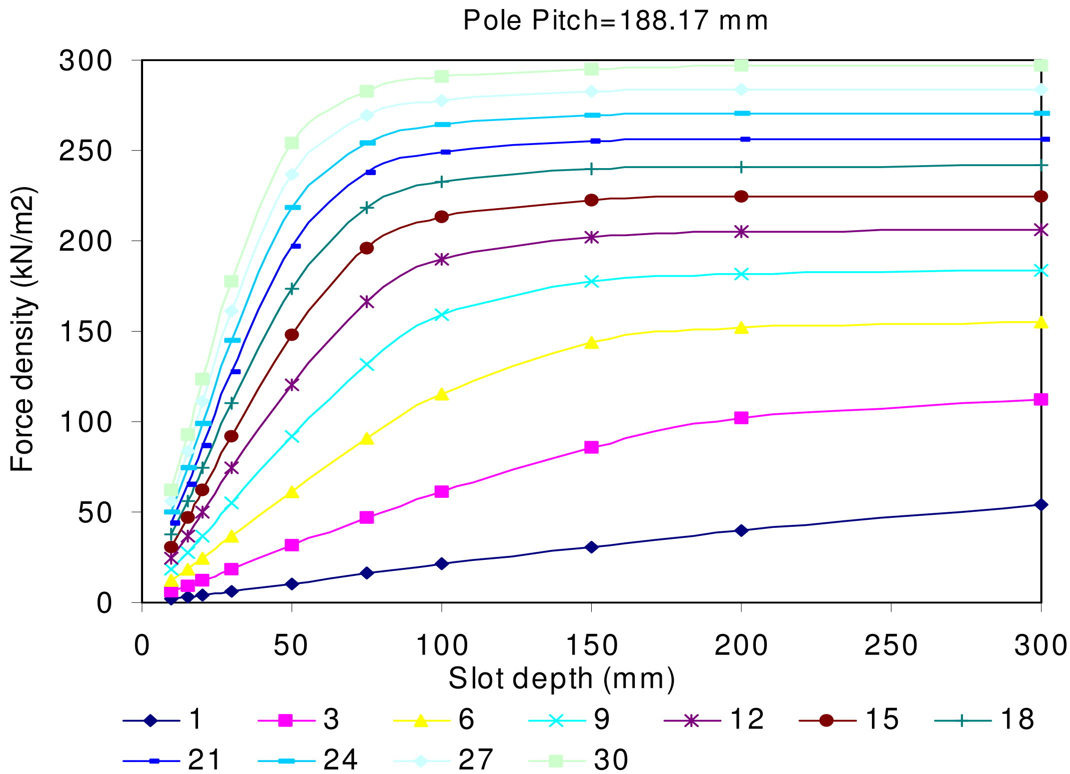

Pole Pitch=83.26 mm
Force density (kN/m2)
Slot depth (mm)
220
200
180
160
140
120
100
80
60
40
20
0
0
50
100
150
200
250
300
1
3
6
9
12
15
18
21
24
27
30

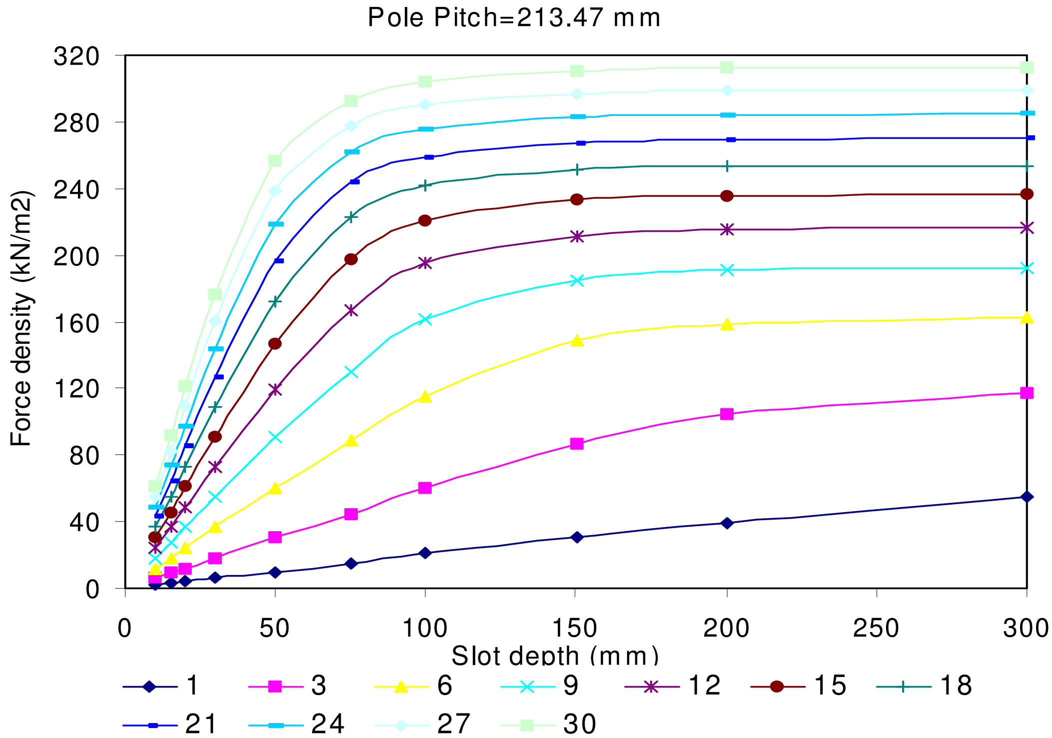

Pole Pitch=97.38 mm
Force density (kN/m2)
Slot depth (mm)
240
220
200
180
160
140
120
100
80
60
40
20
0
0
50
100
150
200
250
300
1
3
6
9
12
15
18
21
24
27
30


Pole Pitch=109.05 mm
Force density (kN/m2)
Slot depth (mm)
0
40
80
120
160
200
240
0
50
100
150
200
250
300
1
3
6
9
12
15
18
21
24
27
30


Pole Pitch=128.31 mm
Force density (kN/m2)
Slot depth (mm)
0
20
40
60
80
100
120
140
160
180
200
220
240
260
0
50
100
150
200
250
300
1
3
6
9
12
15
18
21
24
27
30


Pole Pitch=158.39 mm
Force density (kN/m2)
Slot depth (mm)
0
40
80
120
160
200
240
280
0
50
100
150
200
250
300
1
3
6
9
12
15
18
21
24
27
30


Pole Pitch=188.17 mm
Force density (kN/m2)
Slot depth (mm)
0
50
100
150
200
250
300
0
50
100
150
200
250
300
1
3
6
9
12
15
18
21
24
27
30


Pole Pitch=213.47 mm
Force density (kN/m2)
Slot depth (mm)
0
40
80
120
160
200
240
280
320
0
50
100
150
200
250
300
1
3
6
9
12
15
18
21
24
27
30

Appendix IV

Slot Permeance Coefficient

Introduction:

The permeance coefficients (or specific permeance) of partially closed slots usually encountered in small induction motors are calculated with the aid of Fig. 3.20 (page 72). This data are stored in the developed software by taking a step of 0.01 on the x-axis and storing the corresponding values of the y-axis.

id	$\frac{W_{S2}}{W_{S3}}$	ϕ
1	0.4	0.75
2	0.41	0.73
3	0.42	0.72
4	0.43	0.71
5	0.44	0.7
6	0.45	0.69
7	0.46	0.68
8	0.47	0.67
9	0.48	0.65
10	0.49	0.64
11	0.5	0.63
12	0.51	0.62
13	0.52	0.61
14	0.53	0.6
15	0.54	0.59
16	0.55	0.58
17	0.56	0.57
18	0.57	0.56
19	0.58	0.55
20	0.59	0.54
21	0.6	0.53
22	0.61	0.528
23	0.62	0.524
24	0.63	0.51
25	0.64	0.505
26	0.65	0.5
27	0.66	0.49
28	0.67	0.485
29	0.68	0.48
30	0.69	0.475
31	0.7	0.47
32	0.71	0.46
33	0.72	0.455
34	0.73	0.45
35	0.74	0.435

id	$\frac{W_{S2}}{W_{S3}}$	ϕ
36	0.75	0.43
37	0.76	0.425
38	0.77	0.42
39	0.78	0.418
40	0.79	0.413
41	0.8	0.41
42	0.81	0.405
43	0.82	0.4
44	0.83	0.398
45	0.84	0.393
46	0.85	0.39
47	0.86	0.385
48	0.87	0.38
49	0.88	0.375
50	0.89	0.372
51	0.9	0.37
52	0.91	0.365
53	0.92	0.36
54	0.93	0.355
55	0.94	0.35
56	0.95	0.345
57	0.96	0.34
58	0.97	0.338
59	0.98	0.334
60	0.99	0.329
61	1	0.325
62	1.01	0.32
63	1.02	0.319
64	1.03	0.318
65	1.04	0.316
66	1.05	0.314
67	1.06	0.31
68	1.07	0.308
69	1.08	0.306
70	1.09	0.303

id	$\frac{W_{S2}}{W_{S3}}$	ϕ
71	1.1	0.3
72	1.11	0.295
73	1.12	0.292
74	1.13	0.288
75	1.14	0.285
76	1.15	0.28
77	1.16	0.277
78	1.17	0.276
79	1.18	0.274
80	1.19	0.272
81	1.2	0.27
82	1.21	0.268
83	1.22	0.266
84	1.23	0.264
85	1.24	0.262
86	1.25	0.26
87	1.26	0.258
88	1.27	0.256
89	1.28	0.254
90	1.29	0.252
91	1.3	0.25
92	1.31	0.248
93	1.32	0.247
94	1.33	0.246
95	1.34	0.243
96	1.35	0.241
97	1.36	0.239
98	1.36	0.238
99	1.37	0.236
100	1.38	0.234
101	1.39	0.232
102	1.4	0.23
103	1.41	0.229
104	1.42	0.227
105	1.43	0.226

id	$\frac{W_{S2}}{W_{S3}}$	ϕ
106	1.44	0.224
107	1.45	0.22
108	1.46	0.219
109	1.47	0.217
110	1.48	0.216
111	1.49	0.214
112	1.5	0.213
113	1.51	0.21
114	1.52	0.209
115	1.53	0.208
116	1.54	0.207
117	1.55	0.205
118	1.56	0.204
119	1.57	0.203
120	1.58	0.202
121	1.59	0.201
122	1.6	0.2
123	1.61	0.199
124	1.62	0.198
125	1.63	0.197
126	1.64	0.196
127	1.65	0.195
128	1.66	0.194
129	1.67	0.193
130	1.68	0.192
131	1.69	0.191
132	1.7	0.19
133	1.71	0.189
134	1.72	0.188
135	1.73	0.187
136	1.74	0.186
137	1.75	0.185
138	1.76	0.184
139	1.77	0.183
140	1.78	0.182

id	$\frac{W_{S2}}{W_{S3}}$	ϕ
141	1.79	0.181
142	1.8	0.18
143	1.81	0.179
144	1.82	0.178
145	1.83	0.177
146	1.84	0.176
147	1.85	0.175
148	1.86	0.174
149	1.87	0.173
150	1.88	0.172
151	1.89	0.171
152	1.9	0.17
153	1.91	0.169
154	1.92	0.168
155	1.93	0.167
156	1.94	0.166
157	1.95	0.165
158	1.96	0.164
159	1.97	0.163
160	1.98	0.162
161	1.99	0.161
162	2	0.16
163	2.01	0.159
164	2.02	0.157
165	2.03	0.153
166	2.04	0.151
167	2.05	0.15
168	2.06	0.149
169	2.07	0.147
170	2.08	0.146
171	2.09	0.142
172	2.1	0.14
173	2.11	0.139
174	2.12	0.138
175	2.13	0.137

id	$\frac{W_{S2}}{W_{S3}}$	ϕ
176	2.14	0.136
177	2.15	0.135
178	2.16	0.134
179	2.17	0.133
180	2.18	0.132
181	2.19	0.131
182	2.2	0.13
183	2.21	0.129
184	2.22	0.128
185	2.23	0.127
186	2.24	0.126
187	2.25	0.125
188	2.26	0.124
189	2.27	0.123
190	2.28	0.122
191	2.29	0.121
192	2.3	0.12
193	2.31	0.119
194	2.32	0.118
195	2.33	0.117
196	2.34	0.116
197	2.35	0.115
198	2.36	0.114
199	2.37	0.113
200	2.38	0.112
201	2.39	0.111
202	2.4	0.11
203	2.41	0.109
204	2.42	0.108
205	2.43	0.107
206	2.44	0.106
207	2.45	0.105
208	2.46	0.104
209	2.47	0.103
210	2.48	0.102

id	$\frac{W_{S2}}{W_{S3}}$	ϕ
211	2.49	0.101
212	2.5	0.1
213	2.51	0.075
214	2.52	0.079
215	2.53	0.078
216	2.54	0.077
217	2.55	0.076
218	2.56	0.075
219	2.57	0.074
220	2.58	0.073
221	2.59	0.072
222	2.6	0.071
223	2.61	0.07
224	2.62	0.069
225	2.63	0.068
226	2.64	0.066
227	2.65	0.065
228	2.66	0.064
229	2.67	0.063
230	2.68	0.062
231	2.69	0.06
232	2.7	0.059
233	2.71	0.058
234	2.72	0.057
235	2.73	0.056
236	2.74	0.055
237	2.75	0.05
238	2.76	0.049
239	2.78	0.048
240	2.79	0.047
241	2.8	0.046
242	2.81	0.044
243	2.82	0.04
244	2.83	0.038
245	2.84	0.035

id	$\frac{W_{S2}}{W_{S3}}$	ϕ
246	2.85	0.033
247	2.86	0.03
248	2.87	0.029
249	2.88	0.027
250	2.89	0.026
251	2.9	0.025
252	2.91	0.02
253	2.92	0.019
254	2.93	0.018
255	2.94	0.017
256	2.95	0.016
257	2.96	0.015
258	2.97	0.014
259	2.98	0.013
260	2.99	0.011
261	3	0

Appendix V

MATLAB Simulation [24]

1 Introduction

This document is the final report on the project “Traction Motors” of the “Grön Bil” FC-HEV research program. The project focused on a technology and feasibility survey of traction motors for electric and hybrid vehicles, as a result a traction motor design database and a traction drive simulation model were developed. The design database tool estimates the machines weight and size and the electric and magnetic parameters from a set of input data like torque and speed. The drive model is designed for use in vehicle simulations and includes both the converter and the machine model. The design database was developed at the department of Electrical Power Engineering at Chalmers University of Technology, the dynamic simulation model by Mats Alaküla and Raissa Öttl at the department of Industrial Electrical Engineering and Automation at Lund University.

The report is build up as follows: Chapter 2 contains a user guide for the design database, in chapter 3 the design principle is explained more in detail. Chapter 4 includes an introduction to drive modeling and simulation and a user guide for the different modeling tools. In the last chapter the calculation methods of the different model parts are explained more in detail.

2 User Guide Design Database

3 Traction Motor Design Principles

4 User Guide Drive Simulation

4.1 General features

The presented electric drive model is designed for use in vehicle simulations. The time step is expected to be large compared to the motor and power electronics time constants. For this reason all motor and power electronics transient effects are neglected and the machine model is based on look-up tables. The tables represent the machine torque, current, voltage and losses in dependency of the reference torque, the actual speed and the DC link voltage, thus the motor model becomes independent of the machine type.

In the drive model the look-up tables representing the motor are connected with a thermal model of the machine and a converter model. As input data the drive model accepts the reference torque, the actual rotational speed and the DC link voltage. The output values are the machine torque, the electric power input to the converter and the temperature in different parts of the motor. Even other internal results as the motor phase current and phase-to-phase voltage and the machine and converter losses may be connected as outputs.

The look-up tables are calculated in a machine model generation tool, which is adapted to different motor types. In this first version of the program, tools for generating induction motor (AM) and permanently magnetized synchronous motor (PMSM) models are implemented. Moreover generation tools for the converter model and the thermal motor model are included.

4.2 User Guide

To start model generation open Matlab and execute DriveDataBuild.m. When Matlab is started from the motor design database the model generation is initiated automatically.

The user will be asked to choose a motor and a converter data file. A consistency check will be performed on the input data; the user will be prompted to correct any inconsistent values. Moreover simulation ranges and calculation accuracy can be adjusted. When model generation is finished the user has the possibility to save the drive data for further use. There is even the possibility to open the model automatically while the drive data still is in the workspace. To start a simulation from Matlab load a saved drive-data MAT-file and then run the model EMDrive.

4.2.1 Input data

The m-file for generation of the look-up tables needs a set of motor specific input data. Moreover appropriate ranges for the DC-link voltage and the speed should be chosen. For advanced control of the model's performance even the table sizes can be adjusted.

The input data is divided into two parts, the motor and the converter data, which may be provided in TXT- or MAT-files. In case of MAT-file representation the motor data shall be supplied as a structure called 'machine' and the converter data as a structure called 'converter' with all their elements named as in the table below. In case of TXT-files the data shall be given in the second row of the file and in the order as stated below. The values have to be white space separated. The first row of the TXT-file may be used for the parameter names. It is also possible to provide one part of the input data in a TXT-file and the other one as a MAT-file. Examples for the input data files can be found at the end of this chapter.

The following table shows the motor input data.

Value	**Unit**	**Description**	**Data type**	**Used for**
Type	-	AM or PMSM	String	All
p	-	Number of poles	Integer	All
Ψ_m	Vs	Permanent magnet flux	Float	PMSM
R_s	Ω	Stator resistance	Float	All
R_r	Ω	Rotor resistance, referred to the stator	Float	AM
L_s	H	Stator stray inductance	Float	AM
L_m	H	Main inductance	Float	AM
L_r	H	Rotor stray inductance, referred to the stator	Float	AM
L_{sx}	H	Stator inductance, x-component	Float	PMSM
L_{sy}	H	Stator inductance, y-component	Float	PMSM
$I_{prmsmax}$	A	Maximum phase current	Float	All
$U_{pprmsmax}$	V	Maximum phase to phase voltage	Float	-
n_{max}	rpm	Maximum speed	Float	All
U_n	V	Nominal voltage (phase to phase)	Float	AM
I_n	A	Nominal current (phase)	Float	AM
f_n	Hz	Nominal frequency	Float	AM
P_n	W	Nominal power	Float	AM
n_n	rpm	Nominal speed	Float	AM
y	-	Motor connection, 1 -> star, 0 -> delta	Integer	AM
B_{sln}	T	Nominal tooth flux density	Float	PMSM
B_{yn}	T	Nominal yoke flux density	Float	PMSM

Value	Unit	Description	Data type	Used for
syo	m	Outer stator yoke radius	Float	Thermal
syi	m	Inner stator yoke radius	Float	Thermal
ds	m	Slot height	Float	Thermal
ws	m	Slot width	Float	Thermal
agi	m	Inner air-gap radius	Float	Thermal
ryi	m	Inner rotor yoke radius	Float	Thermal
ksl	-	Copper fill faktor	Float	Thermal
l	m	Stack length	Float	Thermal
ag	m	Air gap	Float	Thermal
s	-	Number of slots	Integer	Thermal
$cool_1$	-	Natural cooling, 1 -> true, 0 -> false	Integer	Thermal
$cool_2$	-	Forced cooling, 1 -> true, 0 -> false	Integer	Thermal
$cool_3$	-	Liquid cooling, 1 -> true, 0 -> false	Integer	Thermal
kc	-	Loss factor according to Steinmetz equation	Float	Thermal

Table 1: Motor Input Data

Notice that TXT-files always must contain the whole set of data. Unused parameters may be set to zero (e.g. PMSM parameters for an AM model and vice versa). When providing data in a MAT-file, only the parameters necessary for the specific model have to be defined.

The following table shows the converter input data.

Value	Unit	Description	Data type	Used for
U_{dcmax}	V	Maximum DC link voltage	Float	All
U_{dcmin}	V	Minimum DC link voltage	Float	All
f_{sw}	Hz	Switching frequency	Float	PE
t_{on}	s	Rise time	Float	PE
t_{off}	s	Fall time	Float	PE
v_{condt}	V	Transistor voltage in the conducting stage	Float	PE
v_{condd}	V	Diode voltage in the conducting stage	Float	PE
I_{max}	A	Maximum converter current	Float	All
U_{num}	-	DC link voltage vector size (total size = U_{num}+1)	Integer	All
T_{num}	-	Torque vector size (total size = T_{num}+1)	Integer	All
ω_{num}	-	Speed vector size (total size = ω_{num}+1)	Integer	All

Table 2: Converter Input Data

Allowed values for the size of the torque and speed vectors are 10, 20, 30, 40, 50 or 60, for the size of the DC link voltage vector 1, 2, 5, 10, 15 and 20. Model generation of course is much faster for a smaller table size. However, keep in mind that the size of the torque and speed vector has direct influence on the accuracy of the generated model. A very small table size may be a good choice for rough estimations but may not result in a model feasible for more exact simulations. Because of the finite number of steps of the current and flux results especially the phase-to-phase voltage, which is calculated from these, is affected. As an example the pictures below show the calculated tables for T_{num} and w_{num} set to 30 in the first case and to 10 in the second.

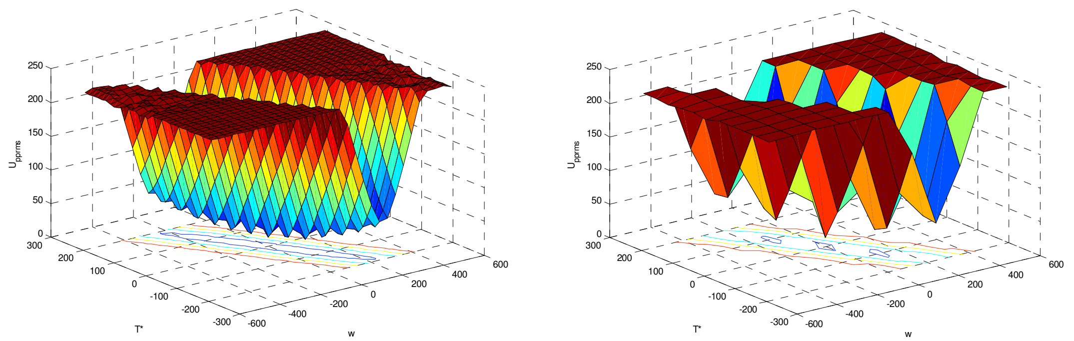
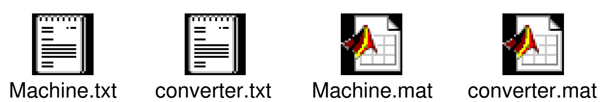


Figure 1: Phase-to-phase voltage table with T_{num} and w_{num} set to 30 (left) and to 10 (right)

The model generation tool will validate the input data. If inconsistent or missing data is detected, the user will be asked to correct the values. Notice that some data, which is not essential for the basic model, may be omitted. In this case some parts of the model (e.g. loss calculation or the thermal model) will not work. Changed data may be saved in MAT-format for further use.

Examples for the machine and the converter TXT- and MAT-files can be found below.



4.2.2 Generation process

Depending on the machine characteristics, the chosen table size and the computer's performance model generation may take from a few minutes up to a couple of hours.

When generating an induction motor model, a sample of the torque, flux, current and voltage tables at the lowest DC link voltage is presented in three-dimensional plots. For a user conversant with Matlab it will be possible to evaluate if the results are feasible for simulation or if a higher resolution or different boundary values have to be chosen. The following picture shows an example of the output plot.

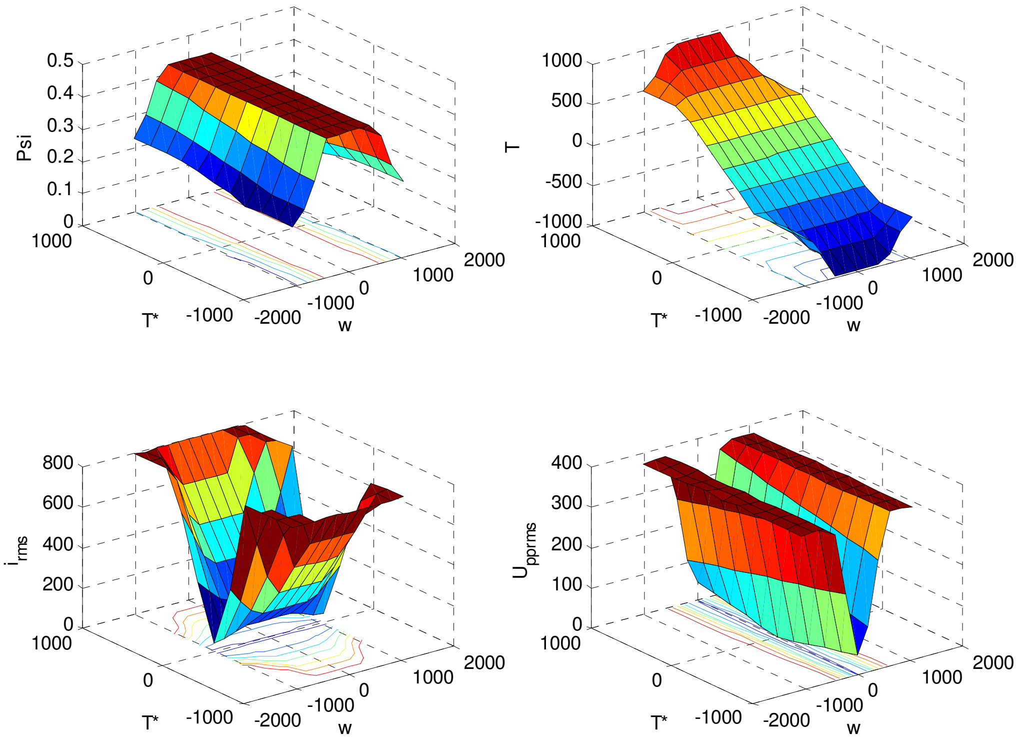


Figure 2: Induction motor generation output plot

4.2.3 Output data

The output data consists of three different parts:

- The electrical characteristics of the machine
- The thermal characteristics of the machine
- The electrical characteristics of the converter

The electrical characteristics of the machine include the following vectors and tables.

Value	**Unit**	**Description**
U_{dc_drive}	V	DC link voltage vector
$\omega_{machine}$	rad/s	Rotational speed vector
$T_{ref\_machine}$	Nm	Reference speed vector
$T_{machine}\ (U_{dc\_drive},\omega_{machine},T_{ref\_machine})$	Nm	Machine torque table
$\Psi_{machine}\ (U_{dc\_drive},\omega_{machine},T_{ref\_machine})$	Vs	Stator flux table
$I_{prms\_machine}\ (U_{dc\_drive},\omega_{machine},T_{ref\_machine})$	A	Phase current table
$U_{pprms\_machine}\ (U_{dc\_drive},\omega_{machine},T_{ref\_machine})$	V	Phase-to-phase voltage table
$P_{Cu\_machine}\ (U_{dc\_drive},\omega_{machine},T_{ref\_machine})$	W	Stator copper loss table
$P_{Fe\_machine}\ (U_{dc\_drive},\omega_{machine},T_{ref\_machine})$	W	Stator iron loss table
$P_{r\_machine}\ (U_{dc\_drive},\omega_{machine},T_{ref\_machine})$	W	Rotor loss table

Table 3: Electrical machine output data

The thermal characteristics consist of the thermal resistances and capacitances for a theoretical part of the machine including one stator slot. The motor model incorporates the number of stator slots to correct for this dependency.

The electrical converter data consists of the following vectors and tables.

Value	Unit	Description
$I_{\mathrm{prms\_conv}}$	A	Phase current vector
$U_{\mathrm{pprms\_conv}}$	V	Phase-to-phase voltage vector
$\eta\ (I_{\mathrm{prms\_conv}}, U_{\mathrm{pprms\_conv}})$	-	Converter efficiency table

Table 4: Converter output data

After the model generation all output data can be saved in a MAT-file.

5 Traction Drive Simulation Principles

5.1 Power Electronics Model

The power electronics simulation model represents a three phase two level self-commutated converter as shown in the following picture.

Picture 3_15 from Power Electronic Control, page 27.

Figure 3: Three phase self-commutated converter

To take full advantage of the DC link voltage, sinus modulation with modified sinus is assumed as control strategy, which results in the voltage waveforms shown below.

Picture 3_21 from Power Electronic Control, page 31.

Figure 4: Sinus modulation with modified sinus

When used in drive simulation, the DC link voltage, the motor's phase current and phase-to-phase voltage are known for the power electronics model. For this reason calculation focuses on loss estimation for each combination of motor current and voltage.

The average loss in the power electronics components is calculated from the sum of the instant losses over one electric period. Assuming motor voltage and current to be in phase a worst-case analysis is done. The instant switching and on-stage loss of the diodes and the thyristors are estimated as described in chapter 5.1.1 and 5.1.2. From the power electronics losses and the converter output power, efficiency and input power can be determined easily. The losses of the control and driver circuits are neglected in the model.

5.1.1 Switching losses

For simplifying calculations it is supposed that both the voltage over and the current through each particular switch are varying uniformly.

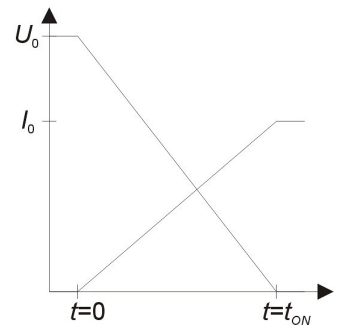


Figure 5: Assumed current and voltage waveforms at turn-on

Thus, current and voltage during switch-on can be described as follows:

$$I(t) = \frac{I_0}{t_{ON}} t \quad U(t) = U_0 - \frac{U_0}{t_{ON}} t \qquad (5\text{-}1)$$

For a particular switch the instant power loss can be expressed as the product of current and voltage. The turn-on energy can then be derived as the integral of the instant power over the switch-on time.

$$W_{ON} = \int_0^{t_{ON}} P(t)dt = \left[\frac{U_0 I_0}{2t_{ON}} t^2 - \frac{U_0 I_0}{3t_{ON}^2} t^3 \right]_0^{t_{ON}} = \frac{1}{6} U_0 I_0 t_{ON} \qquad (5\text{-}2)$$

The average power loss at turn-on is found as the energy divided by the carrier wave period, which corresponds to the inverse of the switching frequency.

$$P_{ON} = \frac{1}{6} U_0 I_0 t_{ON} f_{sw} \qquad (5\text{-}3)$$

In the same way the instant turn-off loss can be calculated. To cover worst case the voltage over the switch at turn-off and turn-on is assumed to be rising to or falling from the maximum DC link voltage.

5.1.2 On-stage Losses

For the calculation of the on-stage losses different equations are worked out for the different switching devices. If the instant phase current is positive either the upper thyristor or the lower diode will carry the current (see Figure 3). For negative currents accordingly the lower thyristor or the upper diode are active. This is accounted for by evaluating the sign of the instant current.

The average on-stage loss during one carrier wave period can be derived from the voltage and current waveforms, which are presented in the following picture.

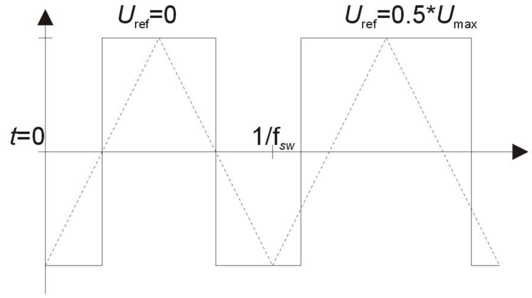


Figure 6: Carrier wave and instant phase voltage for different reference values

Provided that the current is positive, the upper thyristor will be active when the instant phase voltage is positive. During the remaining time the lower diode will carry the current. This leads to the following equation for the upper thyristor's power loss.

$$P_{ON} = U_{cond\_t} \cdot i \cdot \frac{1}{2}\left(1 + \frac{U_{ref}}{U_{DC}/2}\right) \tag{5-4}$$

Accordingly the lower diode's on-stage power loss can be expressed as follows.

$$P_{ON} = U_{cond\_t} \cdot i \cdot \frac{1}{2}\left(1 - \frac{U_{ref}}{U_{DC}/2}\right) \tag{5-5}$$

The same considerations lead to related equations for the lower thyristor and the upper diode. For worst-case calculation the DC link voltage is assumed to be at its maximum.

5.2 Induction Motor Model

For the induction motor model a short-circuited rotor cage is assumed. The model can be used for simulation of both star and delta connected motors. Following the calculation of the torque, current and flux tables is described for the induction motor model. All equations can be found in chapter 5.2.2.

5.2.1 Calculation principle

As a first step the nominal flux is calculated from the nominal motor data and some help variables are defined. The calculation ranges and the table dimensions are determined from the input data limits and the desired precision. Even the flux accuracy is defined here.

In the next step the maximum positive torque at zero speed and nominal flux is calculated with respect to the maximum allowed phase current and the maximum DC-link voltage. This value is used as limit for the reference torque vector. The remaining reference values are determined according to the desired table size. By varying the flux the maximum available torque for each combination of DC-link voltage, reference torque and speed is calculated. For positive speed this calculation is done independently for acceleration and deceleration. The values for negative speed are obtained by mirroring the results for positive speed.

Beyond the values needed for simulation of the motor performance some more tables used in other parts of the model are created. These include the stator flux and the losses tables.

5.2.2 Equations

5.2.2.1 Main equations in stator flux coordinates

In stator flux coordinates, the rotational speed of the coordinate system is corresponding to the rotational frequency of the stator flux and Ψ_{sq} is zero.

$$\omega_k = \omega_s \quad \Psi_{sq} = 0 \tag{5-6}$$

Furthermore it is assumed that the rotor is short-circuited.

$$u_r = 0 \tag{5-7}$$

As transient effects will be neglected due to the time step of about 0.1 s, the rotational speeds are assumed to be constant. Then the following correlation can be established.

$$\omega_s = \omega + \omega_r \tag{5-8}$$

This leads to the stationary main equations in stator flux coordinates.

$$u_{sd} = R_s i_{sd} \tag{5-9}$$

$$u_{sq} = R_s i_{sq} + \omega_s \Psi_{sd} \tag{5-10}$$

$$0 = R_r i_{rd} - \omega_r \Psi_{rq} \tag{5-11}$$

$$0 = R_r i_{rq} + \omega_r \Psi_{rd} \tag{5-12}$$

$$\Psi_{sd} = L_s i_{sd} + L_m i_{rd} \tag{5-13}$$

$$0 = L_s i_{sq} + L_m i_{rq} \tag{5-14}$$

$$\Psi_{rd} = L_r i_{rd} + L_m i_{sd} \tag{5-15}$$

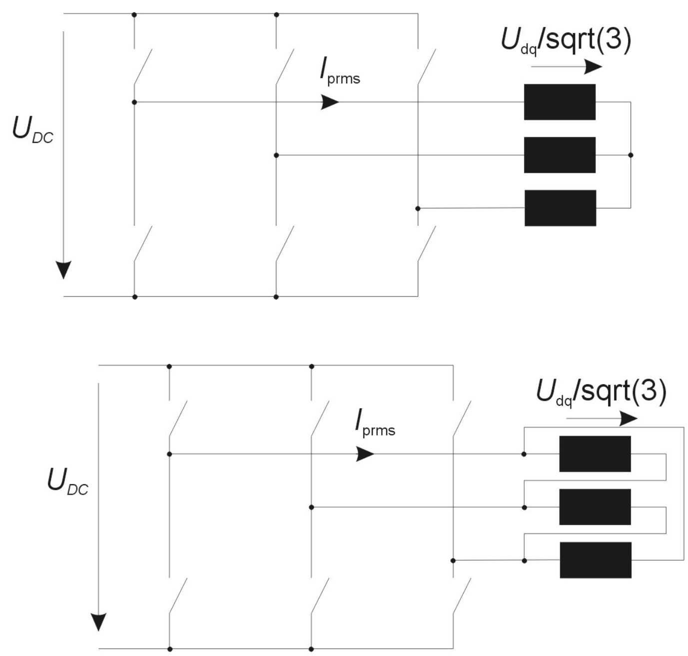
$$\Psi_{rq} = L_r i_{rq} + L_m i_{sq} \tag{5-16}$$

$$T = p\Psi_{sd} i_{sq} \tag{5-17}$$

$$\sigma = 1 - \frac{L_m^2}{L_s L_r} \tag{5-18}$$

$$\tau_r = \frac{L_r}{R_r} \tag{5-19}$$

5.2.2.2 Restrictions



Picture 5-1: DC-AC converter with y- and d-connected motor

$$\sqrt{\frac{2}{3}} \cdot \sqrt{3} \cdot \sqrt{u_{sd}^2 + u_{sq}^2} = \sqrt{2} \cdot \sqrt{u_{sd}^2 + u_{sq}^2} \le U_{DC} \text{ for y - connected motor} \tag{5-20}$$

$$\sqrt{\frac{2}{3}} \cdot \sqrt{u_{sd}^2 + u_{sq}^2} \le U_{DC} \text{ for d - connected motor} \tag{5-21}$$

$$\frac{1}{\sqrt{3}} \cdot \sqrt{i_{sd}^2 + i_{sq}^2} \le I_{MAX} \text{ for y - connected motor} \tag{5-22}$$

$$\sqrt{i_{sd}^2 + i_{sq}^2} \le I_{MAX} \text{ for d - connected motor} \tag{5-23}$$

5.2.2.3 Calculation of nominal flux

5.2.2.3.1 Motor star connected

$$\sqrt{\frac{u_{sd,N}^2 + u_{sq,N}^2}{3}} = \frac{U_N}{\sqrt{3}} \tag{5-24}$$

$$\sqrt{\frac{i_{sd,N}^2 + i_{sq,N}^2}{3}} = I_N \tag{5-25}$$

with (5-25):

$$i_{sd,N}^2 = 3I_N^2 - i_{sq,N}^2 \tag{5-26}$$

with (5-17):

$$i_{sq,N} = \frac{T_N}{p\Psi_{sd,N}} \tag{5-27}$$

(5-24) with (5-9) and (5-10):

$$U_N = \sqrt{R_s^2 i_{sd,N}^2 + \left(R_s i_{sq,N} + \omega_{s,N}\Psi_{sd,N}\right)^2} \tag{5-28}$$

with (5-26) and (5-27):

$$U_N = \sqrt{R_s^2\left(3I_N^2 - \left(\frac{T_N}{p\Psi_{sd,N}}\right)^2\right) + \left(R_s \frac{T_N}{p\Psi_{sd,N}} + \omega_{s,N}\Psi_{sd,N}\right)^2} \tag{5-29}$$

$$U_N = \sqrt{R_s^2 3I_N^2 + 2\cdot R_s \frac{T_N}{p\Psi_{sd,N}}\omega_{s,N}\Psi_{sd,N} + \omega_{s,N}^2\Psi_{sd,N}^2} \tag{5-30}$$

$$0 = R_s^2\cdot 3\cdot I_N^2 p + 2\cdot R_s T_N\omega_{s,N} + p\omega_{s,N}^2\Psi_{sd,N}^2 - pU_N^2 \tag{5-31}$$

$$0 = 3\cdot\frac{R_s^2 I_N^2}{\omega_{s,N}^2} + \frac{2\cdot R_s T_N}{p\omega_{s,N}} + \Psi_{sd,N}^2 - \frac{U_N^2}{\omega_{s,N}^2} \tag{5-32}$$

$$\Psi_{sd,N} = \sqrt{\frac{U_N^2}{\omega_{s,N}^2} - 3\cdot\frac{R_s^2 I_N^2}{\omega_{s,N}^2} - \frac{2\cdot R_s T_N}{p\omega_{s,N}}} \tag{5-33}$$

5.2.2.3.2 Motor delta connected

$$\sqrt{\frac{u_{sd,N}^2 + u_{sq,N}^2}{3}} = U_N \tag{5-34}$$

$$\sqrt{\frac{i_{sd,N}^2 + i_{sq,N}^2}{3}} = \frac{I_N}{\sqrt{3}} \tag{5-35}$$

with (5-25):

$$i_{sd,N}^2 = I_N^2 - i_{sq,N}^2 \tag{5-36}$$

with (5-17):

$$i_{sq,N} = \frac{T_N}{p\Psi_{sd,N}} \tag{5-37}$$

(5-24) with (5-9) and (5-10):

$$U_N = \frac{1}{\sqrt{3}} \cdot \sqrt{R_s^2 i_{sd,N}^2 + \left(R_s i_{sq,N} + \omega_{s,N}\Psi_{sd,N}\right)^2} \tag{5-38}$$

with (5-26) and (5-27):

$$U_N = \frac{1}{\sqrt{3}} \cdot \sqrt{R_s^2\left(I_N^2 - \left(\frac{T_N}{p\Psi_{sd,N}}\right)^2\right) + \left(R_s \frac{T_N}{p\Psi_{sd,N}} + \omega_{s,N}\Psi_{sd,N}\right)^2} \tag{5-39}$$

$$U_N = \frac{1}{\sqrt{3}} \cdot \sqrt{R_s^2 \cdot I_N^2 + 2 \cdot R_s \frac{T_N}{p\Psi_{sd,N}} \omega_{s,N}\Psi_{sd,N} + \omega_{s,N}^2 \Psi_{sd,N}^2} \tag{5-40}$$

$$0 = \frac{1}{3} \cdot R_s^2 \cdot I_N^2 p + \frac{2}{3} \cdot R_s T_N \omega_{s,N} + \frac{1}{3} \cdot p\omega_{s,N}^2 \Psi_{sd,N}^2 - pU_N^2 \tag{5-41}$$

$$0 = \frac{R_s^2 I_N^2}{\omega_{s,N}^2} + \frac{2 \cdot R_s T_N}{p\omega_{s,N}} + \Psi_{sd,N}^2 - \frac{3 \cdot U_N^2}{\omega_{s,N}^2} \tag{5-42}$$

$$\Psi_{sd,N} = \sqrt{\frac{3 \cdot U_N^2}{\omega_{s,N}^2} - \frac{R_s^2 I_N^2}{\omega_{s,N}^2} - \frac{2 \cdot R_s T_N}{p\omega_{s,N}}} \tag{5-43}$$

5.2.2.4 Calculation of the current equations

(5-11) with (5-16):

$$0 = R_r i_{rd} - \omega_r L_r i_{rq} - \omega_r L_m i_{sq} \tag{5-44}$$

with (5-14)

$$0 = R_r i_{rd} + \omega_r L_r \frac{L_s i_{sq}}{L_m} - \omega_r L_m i_{sq} \tag{5-45}$$

with (5-13)

$$0 = R_r \frac{\Psi_{sd} - L_s i_{sd}}{L_m} + \omega_r L_r \frac{L_s i_{sq}}{L_m} - \omega_r L_m i_{sq} \tag{5-46}$$

(5-12)

$$-\omega_r = \frac{R_r i_{rq}}{\Psi_{rd}} \tag{5-47}$$

with (5-14) and (5-15):

$$-\omega_r = -\frac{R_r L_s i_{sq}}{\left(L_r i_{rd} + L_m i_{sd}\right) L_m} \tag{5-48}$$

with (5-13)

$$\omega_r = \frac{R_r L_s i_{sq}}{L_r \Psi_{sd} - L_r L_s i_{sd} + L_m^2 i_{sd}} \tag{5-49}$$

$$\omega_r = \frac{1}{\tau_r} \frac{i_{sq}}{\frac{\Psi_{sd}}{L_s} - \sigma i_{sd}} \tag{5-50}$$

(5-46) with (5-49):

$$0 = R_r \frac{\Psi_{sd} - L_s i_{sd}}{L_m} + \frac{R_r L_s i_{sq}}{L_r \Psi_{sd} - L_r L_s i_{sd} + L_m^2 i_{sd}} \left(L_r \frac{L_s i_{sq}}{L_m} - L_m i_{sq} \right) \tag{5-51}$$

$$0 = \Psi_{sd} - L_s i_{sd} + \frac{L_m^2 L_s i_{sq}^2}{L_r \left(\Psi_{sd} - L_s i_{sd} + \frac{L_m^2 L_s}{L_s L_r} i_{sd} \right)} \left(\frac{L_r L_s}{L_m^2} - 1 \right) \tag{5-52}$$

with (5-18)

$$0 = \Psi_{sd} - L_s i_{sd} + \frac{\sigma L_s^2 i_{sq}^2}{\Psi_{sd} - \sigma L_s i_{sd}} \tag{5-53}$$

$$0 = \Psi_{sd}^2 - \sigma \Psi_{sd} L_s i_{sd} - \Psi_{sd} L_s i_{sd} + \sigma L_s^2 i_{sd}^2 + \sigma L_s^2 i_{sq}^2 \tag{5-54}$$

$$0 = \Psi_{sd}^2 - (1+\sigma)\Psi_{sd} L_s i_{sd} + \sigma L_s^2 i_{sd}^2 + \sigma L_s^2 i_{sq}^2 \tag{5-55}$$

$$0 = \frac{\Psi_{sd}^2}{\sigma L_s^2} - \frac{(1+\sigma)\Psi_{sd}}{\sigma L_s} i_{sd} + i_{sd}^2 + i_{sq}^2 \tag{5-56}$$

This describes a circle in the dq-plane according to:

$$r^2 = \left(i_{sd} - i_{sd,c}\right)^2 + i_{sq}^2 \tag{5-57}$$

with the center placed on the d-axis

$$i_{sd,c} = \frac{1+\sigma}{2\sigma L_s}\Psi_{sd} \tag{5-58}$$

and the radius r, which corresponds to the maximum value of the currents q-component.

$$r = i_{sq,\max} = \frac{\Psi_{sd}(1-\sigma)}{2\sigma L_s} \tag{5-59}$$

with (5-17) the maximum torque can be calculated

$$T_{\max} = p\Psi_{sd} i_{sq,\max} = \frac{p\Psi_{sd}^2(1-\sigma)}{2\sigma L_s} \tag{5-60}$$

For operation with constant stator flux the currents d- and q-component can be derived directly from the desired torque.

$$i_{sq} = \frac{T^*}{p\Psi_{sd}} \tag{5-61}$$

$$0 = \frac{\Psi_{sd}^2}{\sigma L_s^2} - \frac{(1+\sigma)\Psi_{sd}}{\sigma L_s} i_{sd} + i_{sd}^2 + \left(\frac{T^*}{p\Psi_{sd}}\right)^2 \tag{5-62}$$

$$i_{sd} = \frac{(1+\sigma)\Psi_{sd}}{2\sigma L_s} \pm \sqrt{\frac{(1+\sigma)^2\Psi_{sd}^2}{4\sigma^2 L_s^2} - \frac{\Psi_{sd}^2}{\sigma L_s^2} - \frac{T^{*2}}{p^2\Psi_{sd}^2}} \tag{5-63}$$

$$i_{sd} = \frac{(1+\sigma)\Psi_{sd}}{2\sigma L_s} \pm \sqrt{\frac{(1-\sigma)^2\Psi_{sd}^2}{4\sigma^2 L_s^2} - \frac{T^{*2}}{p^2\Psi_{sd}^2}} \tag{5-64}$$

5.2.2.5 Field weakening

With (5-9), (5-10) and (5-20) it can be approved if the DC link voltage is sufficient for the actual combination of flux, torque and speed. This calculation is first done for a y-connected motor.

$$\sqrt{2} \cdot \sqrt{R_s^2 i_{sd}^2 + \left(R_s i_{sq} + \omega_s \Psi_{sd}\right)^2} \le U_{DC} \tag{5-65}$$

with (5-8):

$$\sqrt{2} \cdot \sqrt{R_s^2 i_{sd}^2 + \left(R_s i_{sq} + (\omega + \omega_r)\Psi_{sd}\right)^2} \le U_{DC} \tag{5-66}$$

with (5-50):

$$\sqrt{2}\cdot\sqrt{R_s^2 i_{sd}^2+\left(R_s i_{sq}+\left(\omega+\frac{1}{\tau_r}\frac{i_{sq}}{\frac{\Psi_{sd}}{L_s}-\sigma i_{sd}}\right)\Psi_{sd}\right)^2}\leq U_{DC} \tag{5-67}$$

In the same way the calculation can be done for a d-connected motor with (5-21). The result will be as follows:

$$\sqrt{\frac{2}{3}}\cdot\sqrt{R_s^2 i_{sd}^2+\left(R_s i_{sq}+\left(\omega+\frac{1}{\tau_r}\frac{i_{sq}}{\frac{\Psi_{sd}}{L_s}-\sigma i_{sd}}\right)\Psi_{sd}\right)^2}\leq U_{DC} \tag{5-68}$$

5.3 Permanent Magnet Motor Model

5.4 Thermal Model

The following assumptions are made for the development of the thermal model:

A. The electrical machine consists of *z* similar parts with *z* representing the number of stator slots. The thermal model consists of one of these parts including the stator slot with its coils, half of the stator teeth and the stator yoke parts adjacent on both sides of the slot and the appropriate part of the rotor.

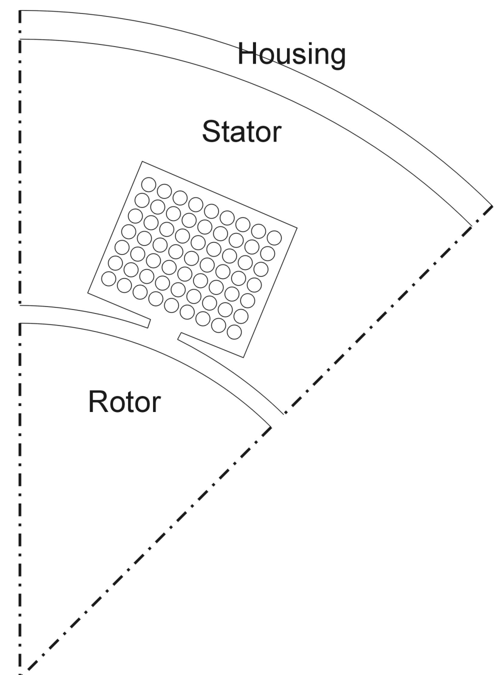


Figure 7: z-th part of the machine

B. The machine is thermally symmetric about a radial plane through its center. The heat flow to the shields on both ends of the machine is merged into one connection with the appropriate thermal resistance in the model.

C. The heat distribution is uniform throughout each solid part of the motor, which means the thermal resistances of iron and copper can be neglected.

D. Because of the model's long time constant of about 100 *m*s, the thermal capacities of the windings and the isolation are neglected.

E. The z-th part of the rotor losses is concentrated in one point on the rotor. The heat is removed over the air gap to the stator iron and over the shaft and the bearings to the end shields. Even the mechanical losses are placed in the rotor.

F. The slot area around the stator windings is filled with a standard isolation material, which has a typical specific thermal conductivity λ_{iso} of 0.35 W/m*K. All heat originating from the stator windings is transferred to the stator iron or the end shields.

G. The stator iron losses are concentrated in one point in the stator yoke. The heat is removed via the housing to the ambient and through the internal and/or the external cooling system.

With assumption A the values of the heat sources are calculated as the z-th part of the total losses:

$$P_{Cu} = \frac{P_{Cu-total}}{z}, \quad P_{Fe} = \frac{P_{Fe-total}}{z}, \quad P_r = \frac{P_{r-total}}{z}, \quad P_{mech} = \frac{P_{mech-total}}{z} \tag{5.69}$$

The following thermal resistances are used in the model:

R_{rs} air gap
R_{rh} rotor to end shield
R_{sh} stator to housing
R_{wh} end winding to end shield
R_{ws} winding to stator iron

5.4.1 Model Structure

The model is based on the known equations for thermal networks.

$$P_{out} = \frac{T_{inside} - T_{outside}}{R} \tag{5.70}$$

$$\frac{\mathrm{d}T_{inside}}{\mathrm{d}t} = \frac{P_{in} - P_{out}}{C} \tag{5.71}$$

5.4.2 Calculation of the thermal resistances

A thermal resistance is generally calculated as

$$R = \frac{d}{\lambda \cdot A} \tag{5.72}$$

In this equation d is the material thickness, A the surface area and λ the specific thermal conductivity of the material.

5.4.2.1 Heat removal from the windings

For the calculation of the thermal resistance in the stator slot, the copper wires and the isolation are assumed to act thermally as a single homogeneous material. With the copper volume V_{Cu} and the isolation volume V_{iso} the thermal conductivity of such a material mixture can be calculated according to [3].

$$\lambda_{slot} = \frac{\lambda_{iso} V_{iso} + \lambda_{Cu} V_{Cu}}{V_{iso} + V_{Cu}} \tag{5.73}$$

Using the cross sectional slot area A_{slot}, the copper fill factor k and the stack length l the copper and isolation volumes can be expressed as:

$$V_{iso} = A_{slot} \cdot (1-k) \cdot l, \quad V_{Cu} = A_{slot} \cdot k \cdot l \tag{5.74}$$

This leads to a simple formula for the resulting thermal conductivity in the slot.

$$\lambda_{slot} = \lambda_{iso}(1-k) + \lambda_{Cu} k \tag{5.75}$$

The heat source resulting from the copper losses is assumed to be concentrated in the middle of the winding. A part of this heat is removed via the stator iron. In addition heat is transferred from the end windings to the housing shields.

The heat transfer from the windings to the stator is described by the thermal resistance R_{ws}. The heat transfer surface area can be approximated with the transition area A_{ws} between the slot and the stator iron. For the material thickness d_{ws} the mean distance from the center of the slot to the stator yoke is taken.

$$A_{ws} = (2h+b) \cdot l, \quad d_{ws} = \sqrt{\frac{1}{3}\left(\left(\frac{h}{2}\right)^2 + 2\left(\frac{b}{2}\right)^2\right)} = \sqrt{\frac{1}{3}\left(\frac{h^2+2b^2}{4}\right)} \tag{5.76}$$

With this equations the thermal resistance R_{ws} is calculated.

$$R_{ws} = \frac{d_{ws}}{\lambda_{slot} \cdot A_{ws}} \tag{5.77}$$

The thermal resistance between the end windings on one side of the motor and the end shield consists of two components, the coil material and the air between coil and housing. The end windings are assumed to correspond to a massive ring, which covers the whole slot height. The thickness of this end winding ring is assumed to be half the slot width. With this suggestion the surface area for the heat transfer from the windings to the air can be calculated. The total area is divided by z to get the surface area for the model A_{wh1}.

$$A_{wh1} = \frac{2\pi \frac{r_{si}+r_{ago}}{2} \cdot \left(h + 2\frac{b}{2}\right)}{z} = \frac{\pi \cdot (r_{si}+r_{ago}) \cdot (h+b)}{z} \tag{5.78}$$

In this equation r_{si} is the inner stator yoke radius and r_{ago} is the outer air gap radius.

The transition area between air and housing A_{wh2} is supposed to be the contact area between the air and the inside of the end shield. A_{wh2} is calculated from the outer stator radius r_{so}, the inner rotor radius r_{ri} and the number of stator slots z.

$$A_{wh2} = \frac{\pi \cdot \left(r_{so}^2 - r_{ri}^2\right)}{z} \qquad (5.79)$$

The material thickness d_{wh1} is calculated as the mean distance from the center to the outside of the end winding ring. The mean distance between the end winding and the housing d_{wh2} is assumed to be about half the slot width, .

$$d_{wh1} = \sqrt{\frac{1}{3}\left(2\left(\frac{h}{2}\right)^2 + \left(\frac{b}{4}\right)^2\right)} = \sqrt{\frac{1}{3} \cdot \frac{8h^2 + b^2}{16}}, \quad d_{wh2} = \frac{b}{2} \qquad (5.80)$$

With the conductivities λ_{slot} of the coil material and λ_{air} of the air the thermal resistance R_{wh} can be determined. In the following equation the heat flow on both ends of the machine is considered by the division by two.

$$R_{wh} = \frac{1}{2}\left(\frac{d_{wh1}}{\lambda_{slot} \cdot A_{wh1}} + \frac{d_{wh2}}{\lambda_{air} \cdot A_{wh2}}\right) \qquad (5.81)$$

5.4.2.2 Heat removal from the rotor

The direct heat transfer from the rotor to the housing is dominated by the bearing resistance. For the determination of the bearing dimensions it is supposed that the shaft diameter is adapted to the static and dynamic load conditions. Then the following rough estimation of the bearing width B and its outer radius r_{bo} can be made.

$$B = \frac{1}{5} r_{ri}, \quad r_{bo} = 2r_{ri} \qquad (5.82)$$

With this values and the number of stator slots z the heat transfer area A_{rh} and the material thickness d_{rh} can be determined.

$$d_{rh} = r_{bo} - r_{ri} = r_{ri}, \quad A_{rh} = \frac{2\pi \cdot r_{ri} \cdot B}{z} = \frac{2\pi \cdot r_{ri}^2}{5z} \qquad (5.83)$$

With this values and the thermal conductivity of steel λ_{Fe} the thermal resistance for the direct heat flow from the rotor to the shields R_{rh} can be determined. The second shield is taken into account by the division by two.

$$R_{rh} = \frac{1}{2}\frac{d_{rh}}{A_{rh}\lambda_{Fe}} \qquad (5.84)$$

In addition heat will be transferred over the air gap to the stator, the main heat transfer mechanism here is convection. In the literature, [3] and [4], a general equation for the convective thermal resistance can be found, where α is the convection coefficient and A the surface area.

$$R = \frac{1}{\alpha \cdot A} \qquad (5.85)$$

In the air gap of an rotating electric machine there are two convection paths coupled in series representing the heat transfer first from the rotor to the air and then further to the stator.

$$R_{rs} = \frac{1}{\alpha_{ra} \cdot A_{ra}} + \frac{1}{\alpha_{as} \cdot A_{as}} \qquad (5.86)$$

The heat transfer surface areas A_{as} and A_{ra} correspond to the z-th part of the inner and outer circumference of the air gap r_{agi} and r_{ago} respectively.

$$A_{ra} = \frac{2\pi \cdot r_{agi}}{z} \cdot l, \quad A_{as} = \frac{2\pi \cdot r_{ago}}{z} \cdot l = \frac{2\pi \cdot (r_{agi} + \delta)}{z} \cdot l \qquad (5.87)$$

As the air gap in electrical machines usually is very small compared to the air gap radius, the following approximation of the area A_{rs} is used for both parts of the thermal resistance.

$$A_{rs} = \frac{2\pi \cdot r_{agi}}{z} \cdot l \qquad (5.88)$$

The heat transfer coefficients α_{ra} and α_{as} depend on the appropriate Nusselt numbers Nu_{ra} and Nu_{as}, the thermal conductivity of the air λ_{air} and the air-gap width δ.

$$\alpha_{ra} = \frac{Nu_{ra} \cdot \lambda_{air}}{2\delta}, \quad \alpha_{as} = \frac{Nu_{as} \cdot \lambda_{air}}{2\delta} \qquad (5.89)$$

The Nusselt number is according to [3] calculated from the Taylor parameter Ta, which describes the kind of the air flow in the air gap. A Taylor number at a value smaller than 1740 indicates laminar air flow, above that turbulent conditions are expected. The Taylor number Ta is calculated from the rotational speed ω, the radius r, the air-gap width δ and the kinematic viscosity of air υ_{air}. A possible form factor is neglected in this model.

$$Ta_{ra} = \frac{\omega^2 \cdot r_{agi} \cdot \delta^3}{\upsilon_{air}^{\;2}}, \quad Ta_{as} = \frac{\omega^2 \cdot (r_{agi} + \delta) \cdot \delta^3}{\upsilon_{air}^{\;2}} \tag{5.90}$$

Like before an approximate value for the Taylor number is used for calculation as the air gap is small compared to the air gap radius.

$$Ta_{rs} = \frac{\omega^2 \cdot r_{agi} \cdot \delta^3}{\upsilon_{air}^{\;2}} \tag{5.91}$$

With the value of Ta_{rs}, the Nusselt number can be found.

$$\begin{aligned} Nu_{rs} &= 2 \quad \text{for} \quad Ta_{rs} \leq 1700 \\ Nu_{rs} &= 0.128 \cdot Ta_{rs}^{0.367} \quad \text{for} \quad Ta_{rs} > 1700 \end{aligned} \tag{5.92}$$

As a result the thermal resistance between rotor and stator can be written as follows :

$$R_{rs} = \frac{2}{\alpha_{rs} \cdot A_{rs}} \tag{5.93}$$

To account for the extreme surface roughness in switched reluctance machines the value of the rotor Nusselt number is increased by about 10 % according to [3]. This leads to a slightly different equation for the thermal resistance.

$$R_{rs} = \frac{2.1}{1.1 \cdot \alpha_{rs} \cdot A_{rs}} \tag{5.94}$$

5.4.2.3 Heat removal from the stator

The heat transfer from the stator to the housing is affected by the thermal resistance R_{sh}. From the outer stator radius r_{so} and the stack length l the contact area to the housing A_{sh} can easily be calculated

$$A_{sh} = \frac{2\pi \cdot r_{so} \cdot l}{z} \tag{5.95}$$

Usually the contact between stator iron and housing is not completely tight. For the simulation model an air filled gap of about fifty micrometer between stator and enclosure is assumed. This leads to the following equation for R_{sh}, where r_{so} is the outer stator radius.

$$R_{sh} = \frac{50\,\mu\text{m}}{\lambda_{air} \cdot A_{sh}} \tag{5.96}$$

5.4.2.4 Heat removal from the housing

The motors are supposed to be cooled on the outside of the frame. An exact calculation of the heat removal from the housing is nearly impossible due to the fact that the dominating heat transfer mechanism is convection. The formulas available in literature are mostly based on experimental results and cannot be adapted to the actual housing geometry. For this reason it was chosen to base the calculation for this general purpose model on approved suggestions on the cooling capacity of motor housings.

The effect of the cooling is modeled based on data derived from standard motors for three different methods, the values are adapted to the model by the division by z.

Air cooling by natural convection:

$$\frac{P_{cool}}{A} = \frac{1.5}{z} \frac{k\text{W}}{\text{m}^2} \text{ at } \Delta T = 60° \tag{5.97}$$

Forced air cooling:

$$\frac{P_{cool}}{A} = \frac{5}{z} \frac{k\text{W}}{\text{m}^2} \text{ at } \Delta T = 60° \tag{5.98}$$

Liquid cooling:

$$\frac{P_{cool}}{A} = \frac{25}{z} \frac{k\text{W}}{\text{m}^2} \text{ at } \Delta T = 60° \tag{5.99}$$

As the effect of the cooling is proportional to the temperature difference the actual heat removal can be calculated at each time. With the housing area, which is approximated by the outer stator surface A_{sh}, this leads to the following expression.

$$R_{ha} = \frac{60°\text{C}}{x \cdot A_{sh}} \tag{5.100}$$

Here x is 1.5 kW/m^2, 5 kW/m^2 or 25 kW/m^2 for natural, forced or liquid cooling respectively..

5.4.3 Calculation of the thermal capacities

The thermal capacity is generally calculated as

$$C = c \cdot \rho \cdot V \tag{5.101}$$

In this equation V is the volume, c the specific thermal capacity and ρ the density of the material.

According to assumption D the thermal capacities of the stator, the rotor and the housing with the end shields are used in the model, all other capacities are neglected.

C_s stator
C_r rotor
C_h housing

5.4.3.1 Thermal capacity of the stator

The volume of the stator is calculated as the product of the stators cross sectional area and the stack length l. The cross sectional area can be derived from the outer stator radius r_{so}, the outer air gap radius r_{ago} – which is expressed as the inner air gap radius plus the air gap width – and the slot area A_{slot}. The total volume is divided by z to retrieve the part needed for the model.

$$V_s = \frac{\left(\pi\left(r_{so}^2 - \left(r_{agi} + \delta\right)^2\right) - z \cdot A_{slot}\right) \cdot l}{z} \tag{5.102}$$

With the density and the specific heat capacity of steel C_s can be found.

$$C_s = c_{Fe} \cdot \rho_{Fe} \cdot V_s \tag{5.103}$$

5.4.3.2 Thermal capacity of the rotor

The heat capacity of the rotor is calculated in the same way

$$C_r = c_{Fe} \cdot \rho_{Fe} \cdot V_r \tag{5.104}$$

with the rotor volume V_r.

$$V_r = \frac{\pi r_{agi}^2 \cdot l}{z} \tag{5.105}$$

In this calculation eventual rotor slots are neglected.

5.4.3.3 Thermal capacity of the housing

For the calculation of the volume of cylindrical housing part it is assumed that the housing thickness inclusive fins is approximately 10 % of the outer stator radius. Moreover the housing material is supposed to fill half the area between its outer and inner radius. The bearing shields are supposed to correspond to a flat plate with a thickness of 5 % the stack length. With these suggestions the following formula for the z-th part of the housing volume can be found.

$$V_h = \frac{\pi \frac{\left((1.1 \cdot r_{so})^2 - r_{so}^2\right) \cdot l}{2} + 2 \cdot \pi r_{so}^2 \cdot 0.05 \cdot l}{z} = \frac{0.15 \cdot \pi r_{so}^2 \cdot l}{z} \quad (5.106)$$

The heat capacity of the housing is then calculated in the same way as above.

$$C_h = c_{Fe} \cdot \rho_{Fe} \cdot V_h \quad (5.107)$$

5.4.4 List of Symbols

A	general area
A_{ra}	z-th part of the transition area rotor to air gap
A_{rh}	z-th part of the bearing transition area
A_{as}	z-th part of the transition area air gap to stator
A_{slot}	cross sectional slot area
A_{wh1}	z-th part of the transition area end windings to air
A_{wh2}	z-th part of the transition area air to bearing shield
A_{ws}	surface area between stator slot and stator iron
b	slot width
B	bearing width
c	general specific heat capacity
c_{Fe}	specific heat capacity of steel
C	general heat capacity
C_h	z-th part of the housing heat capacity
C_r	z-th part of the rotor heat capacity
C_s	z-th part of the stator heat capacity
d	general material thickness
d_{rh}	material thickness shaft to housing
d_{wh1}	mean distance end winding center to air
d_{wh2}	mean distance end winding to housing
d_{ws}	mean distance slot center to stator iron
D	bearing outer diameter

F_c	dynamic load
G	bearing inner diameter
h	slot height
k	copper fill factor
l	stack length
m_r	rotor mass
Nu	general Nusselt number
Nu_{ra}	rotor Nusselt number
Nu_{as}	stator Nusselt number
P	general thermal effect
P_{cool}	z-th part of the heat removal
P_{Cu}	z-th part of the copper losses
P_{Fe}	z-th part of the stator iron losses
P_{mech}	z-th part of the mechanical losses
P_r	z-th part of the rotor losses
q_{rh}	help variable calculated from bearing dimensions
r	general radius
r_{agi}	inner air gap radius
r_{ago}	outer air gap radius
r_{ri}	inner rotor yoke radius
r_{si}	inner stator yoke radius
r_{so}	outer stator yoke radius
R	general thermal resistance
R_{rs}	z times the thermal resistance of the air gap
R_{rh}	z times the thermal resistance rotor to end shield
R_{sh}	z times the thermal resistance stator to housing
R_{wh}	z times the thermal resistance winding to end shield
R_{ws}	thermal resistance winding to stator iron
Ta	general Taylor number
Ta_{ra}	rotor Taylor number
Ta_{as}	stator Taylor number
T	general temperature
T_a	ambient temperature
T_h	housing temperature
T_r	rotor temperature
T_s	stator temperature
T_w	winding temperature
V	general volume
V_{Cu}	copper volume in one slot
V_{iso}	isolation volume in one slot
V_h	z-th part of the housing volume
V_r	z-th part of the rotor volume
V_s	z-th part of the stator volume

z	number of stator poles
α	general convection coefficient
α_{ra}	convection coefficient rotor to air gap
α_{as}	convection coefficient air gap to stator
δ	air gap width
λ	general specific thermal conductivity
λ_{air}	specific thermal conductivity of air
λ_{Cu}	specific thermal conductivity of copper
λ_{iso}	specific thermal conductivity of isolation material
λ_{slot}	substitute specific thermal conductivity in the stator slot
λ_{Fe}	specific thermal conductivity of steel
υ_{air}	viscosity of air
ρ	general density
ω	general angular frequency
ω_{max}	maximum angular frequency

5.4.5 References

[1] Bertotti, Giorgio (1988), General Properties of Power Losses in Soft Ferromagnetic Materials. *IEEE Transactions on Magnetics*, 24, p. 612-630.

[2] Schulte, Stephan (2000), *Thermische Modellierung einer Geschalteten Reluktanzmaschine*, Aachen Technical University

[3] Driesen, Johan (2000), *Coupled Electromagnetic-thermal problems in Electrical Energy Transducers*, Katholieke Universiteit Leuven.

[4] Lindström, Joachim (1999), *Thermal Model of a Permanent-Magnet Motor for a Hybrid Electric Vehicle*, Chalmers University of Technology.

[5] Henneberger, Gerhard (2001), E*lektrische Maschinen I*, Aachen Technical University.

References

[1] Hamdi, E S: Design of small electric machines, John Wiley & Sons, 1994, 1996, 1998.
[2] Hamdi, E S: Permanent magnet and variable reluctance drive systems, ETI Press, 2003.
[3] Kasinathan, P: Integrated in-wheel motors for low-power traction applications, PhD thesis, Chalmers University of Technology, Gothenburg, Sweden, 2003.
[4] Didik, F: History and directory of electric cars, www.didik.com/ev_hist.htm.
[5] Miller, J M: Propulsion systems for hybrid vehicles, The Institution of Electrical Engineers, Stevenage, UK, 2004.
[6] Smith, J R: Hybrid vehicle and hydrogen fuel efficiency, University of California, USA, www.llnl.gov/str/Hybrid.html.
[7] Burch, S D: Trading off HEV fuel economy and emissions through optimization, 1999, National Renewable Energy Laboratory (NREL), USA, www.ctts.nrel.gov/analysis/documents/HV_TOPs4.PDF
[8] Electric Power Research Institute, Electric drive, market transformation, www.svmg.org/Committees/Energy/Related_Documents/pmgraham.pdf
[9] National Renewable Energy Laboratory (NREL), Hydrogen, fuel cells & infrastructure technologies program, www.eere.energy.gov/hydrogenandfuelcells/.
[10] Hybrid electric vehicle, Illinois Institute of Technology, Chicago, USA, www.ece.iit.edu/~hev/whatishev.html.
[11] Institute of Urban Traffic Research: HEV Hybrid vehicle, www.toshikotsu.or.jp/eng/html/hi.htm
[12] Dettmer R: Hybrid pioneers, IEE Review, January 2003, pp. 42-45.
[13] Chapman, D: Developing secure applications with visual basic, Sams, Indianapolis, Ind, USA, 2000
[14] MSDN, Visual Basic. 2005 Microsoft Corporation,

msdn.microsoft.com/library/default.asp?url=/library/en-us/vbcon98/html/vbconwelcometovisualbasic.asp
[15] Spasov, P: Programming for technology students using Visual Basic, Prentice Hall, New Jersey, USA, 2002
[16] Ek, J: VBA-programmering Office 2000, Pagina, Stockholm, Sweden, 2000.
[17] Din, I A: Structured query language (SQL), Blackwell, 2003.
[18] Pfleeger C P and Pfleeger S L: Security in computing, Prentice Hall, New Jersey, 1989, 1997, 2003.
[21] Robenson, G: Real world microsoft access database protection and security, APRESS, 2004.
[19] Robenson, G: Restricting who can use your database, msdn.microsoft.com/library/default.asp?url=/library/en-us/dnsmart04/html/sa04a11.asp?frame=true
[20] MATLAB, 1994-2005 The MathWorks, www.mathworks.com/products/combuilder.
[21] Design & analysis software for electromagnatics, www.infolytica.com/en/
[22] Madden, J (Ed): Networking essentials, Microsoft Press, Washington, USA, 1998.
[23] Kienle and Spiess (1985) Stator and rotor laminations for induction motors, Stanzund Druckgieβwerk GmbH,D-7123 Sachsenheim 1, Germany.
[24] Alaküla, M and Öttl, R: Private communication, 2004.