Copyright
by
Albert Swantner
2009
AUTOMATED SYNTHESIS AND OPTIMIZATION OF GEAR
TRAIN TOPOLOGIES

BY

ALBERT SWANTNER, B.S.

THESIS
Presented to the Faculty of the Graduate School of
The University of Texas at Austin
in Partial Fulfillment
of the Requirements
for the Degree of

MASTER OF SCIENCE IN ENGINEERING

THE UNIVERSITY OF TEXAS AT AUSTIN

MAY 2009
AUTOMATED SYNTHESIS AND OPTIMIZATION OF GEAR TRAIN TOPOLOGIES

Approved by
Supervising Committee:

__________________________

__________________________

4
ACKNOWLEDGEMENTS

First and foremost I need to thank my supervisor Dr. Matthew Campbell. Without his help and guidance I would not have even entered graduate school, much less be graduating with my master’s degree. Meeting with him every week for the past four years have been some of the best and most meaningful hours spent at the ETC. I would also like to thank Dr. Carolyn Seepersad for her guidance in my research and learning so much from her about how to work with people and be a leader while I was her TA for ME 366J. I would also like to thank the National Science Foundation for their support under grant CMMI-0448806.

The students in the automated design lab were great to work with, especially David and Pradeep who are the current members of lab who have been there the last two years. It was also a pleasure to work with Bryan, who cares more about research and works much harder than I did when I was a freshman, on the visualization.

I would also like to thank my family because without their support I would not be where I am today. My parents were always there to support me and to lend a helping hand wherever they could. I would especially like to thank my aunt Karen Palafox who has helped me with my undergraduate education more than she can imagine.

I would also like to thank my friends because without their distractions I would have gone crazy doing this work, especially Richard, Eric, Erika, Mikey, Brad, Felisa,
Natasha, Esther, and Teena. Finally, I would like to thank Joceline who has helped me every step of the way through my undergraduate/graduate education.

May 8, 2009
ABSTRACT

AUTOMATED SYNTHESIS AND OPTIMIZATION OF GEAR TRAIN TOPOLOGIES

Albert Swantner, M.S.
The University of Texas at Austin, 2009
Supervisor: Dr. Matthew Campbell

A method for automating the design of simple, compound, bevel, and worm gear trains using graph grammars is described. The resulting computational tool removes the tedium for engineering designers searching through the immense number of possible gear choices and combinations by hand. The variables that are automatically optimized by the computational tool include the gear dimensions as well as the location of the gears in space. The gear trains are optimized using a three-step process. The first step is a tree-search based on a language of gear rules which represent all possible gear train configurations. The second step optimizes the discrete values such as number of teeth through an exhaustive search of a gear catalog. The final step is a gradient-based algorithm which optimizes the non-discrete variables such as angles and lengths in the positioning of the gears. The advantage of this method is that the graph grammar allows all possible gear train combinations to be included in the search space while the method of optimization ensures the optimal candidate for a given problem is chosen with the tuned parameters.
# Table of Contents

Acknowledgements ........................................................................................................ 5

Abstract .......................................................................................................................... 7

Table of Contents .......................................................................................................... 8

List of Figures ................................................................................................................. 10

List of Tables .................................................................................................................. 12

Chapter 1: Introduction .................................................................................................. 14

Chapter 2: Background .................................................................................................. 16

Chapter 3: Rule Development ...................................................................................... 19

Chapter 4: Creating Topologies ..................................................................................... 22

Chapter 5: Optimization ............................................................................................... 24
LIST OF FIGURES
LIST OF TABLES
CHAPTER 1: INTRODUCTION

Gears have been used since the advent of rotating machinery, using wooden pegs to retract a catapult or raise and lower a ship anchor, and were even referenced by DaVinci in his drawings (Norton, 2006). Gears used in these simple implementations were sufficient until the industrial revolution which led to a completely new era in the application of gears. Gears needed to be put together in a train to create mechanical energy and to accurately and predictably change the speed of a motor. For this reason, metal gears were manufactured and used in a variety of applications and machinery. The production of gears has continuously progressed to today, when gears are precision machined metal or plastic and are used in everything, including, but not limited to, automobiles, clocks, and robotic equipment.

Designing gears involves performing a number of simple repetitive calculations, keeping in mind certain rules of thumb while performing a number of iterations. Simple, refined mathematical formulas are used to find a combination of gears that meet exact specifications and ensure a safety factor. Manual design of gears becomes increasingly tedious as gear specifications become more complex. At the same time, customization of gears is necessary as, otherwise, the design space is limited to a set of gears with discrete numbers of teeth. For example, if a 19 tooth gear is needed but not available in the design set, a different combination must be found which will give acceptable results.
Therefore, it is desirable to create a computing system that is able to automate gear
design set to the user’s specifications. The user also has rules of thumb at his or her
disposal - if the goal is to half the input speed, the designer will place a small gear
off the motor in contact with a large gear because this decreases the output speed. If a
high torque motor is being used, plastic gearing may not be able to withstand the stress
and metal gears may be more appropriate. To save time and ensure an optimal design,
the designer needs a tool which combines the designer’s knowledge about gear trains
with the efficiency and speed of a computer.

The benefit of using a computer to solve this problem is in its ability to do
calculations rapidly while having an extensive memory. For this gear train design
experiment we are taking advantage of both of these benefits. The computer can search
through an extensive database and optimize a gear train configuration in less time than a
human would take to come up with a solution. The computer also has the ability to
guarantee an optimal solution because it has stored and considered all of the other
possibilities. This would be impossible for a human designer based on the sheer number
of different gear choices available at each step of the design.

In a gear train design problem, the user is presented a multi-objective problem
that must be solved. He must keep in mind the design parameters and goals that his gear
train must meet. What is the final speed? Where should the gear train output be located?
He must balance these questions with design considerations for the gear train. What is
the stress on each of the gears? Is the gear train constrained in any direction? Figure 1
shows a graphical representation of the user requirements showing input and output location along with the bounding box. The goal of this program is to be able to answer the above questions and meet the constraints while simultaneously minimizing the weight and maximizing efficiency.

By programming knowledge about how gear trains are configured we can use the computer to generate all the possible gear trains automatically without any input from the user. To accomplish this, we have created a language which describes how gear trains are configured. This language is a graph grammar which defines the tree search space. The tool used to implement this grammar, GraphSynth, was developed by Dr. Matthew Campbell in the Automated Design Lab at the University of Texas at Austin (Campbell, 2008). Using this tool, a graph is created, using nodes and arcs, in this case representing a gear train. Some of the benefits of using a grammar are that once the rules are defined any topology can be generated, creating the rules is simple, and recognition and application routines are standardized. After the topology has been generated, the computer searches through a database of gears which includes values for the face width, pitch, and number of teeth until a combination is found which satisfies three of the five
constraints: output speed, pitch, and stress. The computer then takes this gear train candidate and optimizes the non-discrete variables for location. The location is defined based on the configuration of the gears and is optimized to satisfy the final two constraints: a target output location and staying within a user defined box.
CHAPTER 2: BACKGROUND

The equations relating to the kinematics, fatigue and wear of the gears can be found in many textbooks

(Norton, 2006),

(Budynas & Nisbett, 2008). In the results, the gears used for the implementation phase and in the exhaustive search and the information and properties are from the McMaster Carr catalog

(McMaster Carr, 2008). Previous work on gear optimization has been done by Prayoonrat and Walton who use a two-stage algorithm similar to the one we use but the gear trains that can be created are limited and different material types are not considered

(Prayoonrat & Walton, March 1998). Pomrehn and Papalambros have also worked with gear train optimization using many of the same constraints but where our approach differs is in the different possibilities for the gear trains using grammars and the minimal input required by the user

(Pomrehn & Papalambros, September 1995). Solving this type of gear problem has been done by many other researchers in the past but not using this combination of algorithms with a grammar for variety in designs. Using our method allows the designer to consider a large variety of topologies rather than being restricted by the optimization process.
Graph grammars are a branch of graph theory that defines mathematical operations such as addition and intersection in graphs. Mathematicians have developed the foundations of graph theory research, and engineering researchers have used the concept to describe complex engineering systems (Antonsson & Cagan, 2001). Graphs are used throughout engineering from circuit diagrams describing electrical connectivity to flow charts showing how a process works. There are many more applications where viewing the system as a graph helps the user visualize the system. To define what operations may be performed on a graph, there are three steps that must be considered: the rules which are applicable must be found (this is known as the recognition), the choice must be made of which rule to apply, and applying the rule to change the graph (this is referred to as application).

The recognition step involves finding if and where a rule can be called on an existing graph. This is done by taking the left hand of a rule and searching for that specific subgraph within a host graph. The addition of labels reduces the possible applications of a rule and helps narrow the possibilities. The next step is to choose which rules to apply by determining which rule would make the best modification to the graph. This choice step defines how we search the space of feasible solutions. A systematic or algorithmic process is developed to repeatedly make this choice to arrive at candidate solutions. The final change is to apply the rule to the graph according to the specifications provided by the rules. Application is the most difficult step to perform.
with a number of different theories for how this is best done. In essence, we have adopted the double pushout method (Ehrig, Pfender, & Schneider, 1973) which starts by removing the elements that exist only on the left-hand side of the rule. This step also replaces labels and variables that may be found in the host with the ones found in the rule. After the common elements are replaced, the unique elements found in the rule’s right-hand side are added to the host. In addition to the double-pushout method, a final step has been added wherein the graph is repaired by attaching or deleting dangling arcs in order keep a complete graph (Nagl, 1976).

(Campbell, 2008)

Grammars have a number of applications, from implementing a MEMS fabrication sequence (Jawalkar & Campbell, 2007) to developing neural networks (Vempati & Campbell, 2007) to more strictly mechanical application such as epicyclic gear trains (Schmidt & Chase, December 2000) and clocks (Starling & Shea, 2002). Our focus is on the possibilities of grammars to assist in the creation of gear trains including simple, compound, bevel, and worm gear trains,
ignoring epicyclic gear train configurations and helical gears. There has been research on epicyclic gear trains and the modifications to the grammar to encompass more designs (Li, Schmidt, He, & Li, 2004). In Li et al.

(Li, Schmidt, He, & Li, 2004), the focus is on a grammar based designer tool, which helps designers during the conceptual design phase. Their research does not focus on the utilization of rules to create design alternatives, only on the creation of the grammar rules. What makes the work we are presenting different is that it essentially combines Pomrehn’s

(Pomrehn & Papalambros, September 1995) and Starling’s work

(Starling & Shea, 2002). Starling uses grammar rules to come up with unique solutions while Pomrehn uses traditional optimization methods to find the optimal. We use graph grammars to create a wide range of gear trains, including bevel and worm configurations, which is something that has not been done to this point. Then we use an exhaustive and gradient based search to decide on an optimal solution while keeping the whole process completely automated.
In order for a grammar to work there must be rules which define the language. Within these rules is embedded an understanding of the recognize, choose, and apply cycle that goes on within GraphSynth to modify a graph based on the rules. This process will be described in more detail along with a description of the rules that have been developed which can create a simple gear train, compound gear train, and a gear train which involves worm and bevel gears.

To define a simple gear train there is one rule that needs to be implemented. If a gear has not had an operation performed on it, a mating gear is added to the gear train. At this point no spatial information is recorded, only how the gears are attached. A tag is added telling the computer that the gear added is in contact with the gear before it. When the algorithm goes through calculating the speed, it is necessary to know if the gear is in contact with the previous gear or on the same shaft because in one case the speed will be changed while in the other case the speed is the same. In addition to the gear being added by the rule, a shaft running through the center of the gear is added along with bearings. The gear is press fit onto the shaft and the shaft spins freely on the bearings.

Figure 2 shows a GraphSynth representation of the rule along with a computer-aided design (CAD) model of what is happening in the physical system. On the left side of the rule is the sub-graph GraphSynth looks for so that the right hand side can be
applied to the host graph. In this instance, the left side of the rule recognizes a gear which had not had an operation performed on it as indicated by the new label and a mating gear is then applied. The label new is removed from the shaded gear because no more operations can be performed on this gear. The gear that has been added is given the label new which indicates that an operation can be performed on it as shown on the right hand side of the rule.

The second rule in the set of gear rules defines a compound gear train. In this configuration a gear is added onto the same shaft as another gear with the tag common and this tells the computer that this gear shares a common shaft. Figure 3 shows a GraphSynth representation of this rule in which a gear pair must be found in order for a compound gear to be added. The right side of the figure shows how the gear is added and this rule also ensures there are only two gears per shaft. For this research, only a single output is considered so there is no need for more than two gears on a single shaft.
The third design rule that has been created is one that adds a worm gear into a gear train set. The advantage of using a worm gear is that it dramatically changes the gear ratio in a small amount of space. This addition has allowed a large range of target speeds to be introduced. When a worm gear is used, it is typically at the beginning of a gear train directly off the motor but in some cases you may want to have a standard gear train first with a worm gear added at a later point and the rules allow for both of these configurations.

To add a worm gear directly off the motor the rule is setup to recognize an input shaft and then adds a worm with the mating worm gear onto the seed shaft. The mating worm gear is given the new label and this allows the generation to continue. The worm gears that are being used for this research cannot be directly combined with spur gears so the only possible next step in the generation is for a rule similar to the compound rule to be called. This ensures that the worm and worm gear are isolated in the gear train and not combined with another type of gear.
In the second configuration where the worm is not directly off of the motor, the rule searches for a gear pair of any type where one of the gears has the new label and then adds a shaft with bearings, and a worm gear with corresponding mating gear. This rule can be called after a pair of bevel gears, a compound spur gear configuration, or even a previous worm gear pair. This configuration is shown in Figure 4 and is a slightly more complicated rule than previously seen and also adds more components than the other rules.

**Figure: GraphSynth Representation of the Worm Gear Rule**

The last design rule that has been implemented is a bevel gear rule. What is unique about bevel gears is their ability to change the direction of the output and in order for bevel gears to work correctly the bevel gears must mate with other bevel gears. In the implementation of this rule, the recognition step looks for a shaft which already has a gear and then adds a bevel gear to that same shaft as shown in Figure 5. This ensures that the bevel gear is not in contact with a gear that is not a bevel gear. Figure 5 shows a three step process for the addition of a bevel gear set and this really means two different rules have to be called. The reason for this is that there are two types of bevel gears in the catalog and they may be able mate with each other so this possibility must be accounted for. The second bevel gear rule, as shown in the third pane of Figure 5, mates a bevel gear of type bevelA to the previous bevel gear. There is another rule which mates
a gear of type bevelB to a previous bevel gear and this is why this bevel gear addition must be done in two steps while the worm addition could be completed in one step. A bevel gear can also be called directly off of an input shaft in a similar fashion as the gears that have been previously described.

**Figure: A GraphSynth representation of the two bevel rules**

Using these rules, there are an infinite number of possible gear trains combinations which can be assembled. The simplest gear train possible would be two mating gears while the most complex will have a combination of spur, bevel, and worm gears. As stated earlier, this is the step that makes this work truly unique. Many of the previous papers on other gear optimization and design are constrained either to combinations of spur gears or to epicyclic gear trains. This work goes one step further to include bevel and worm gears in addition to spur gears. If more time were available then it would be possible to add even more complexity such as helical and epicyclic gear trains in order for a truly optimal gear train to be presented. Despite the time limitations, the work completed allows for a wider range of speeds to be solved as well as much more complicated three dimensional geometries while concentrating on a single input and output.
CHAPTER 4: CREATING TOPOLOGIES

Building from an initial seed graph that contains only the input shaft, a tree of solutions is created. An example of this tree is shown in Figure 6. At the top of the tree is the seed shaft and as gears are added the tree becomes more complex. This added complexity means that there are more gears which result in more cost and size as we progress down the tree.

The first implementation of this research included a large number of rules with very little variety because the rules had the values for the gear variables embedded within them. Each distinctly different rule (e.g. simple, compound, etc...) contained only eight implementations to include eight different possibilities for number of teeth based off a partial gear family from an online catalog (McMaster Carr, 2008). As the search progressed down the tree, the stack of candidates quickly became very large due to the high branching factor at each level. Additionally, obtaining more complicated gear ratios became impossible due to the restrictions if the small catalog of gears being considered.
To increase the robustness of the grammar, the rules were simplified so that the tree search could be done in a shorter time and contain a more meaningful variety of solutions. To accomplish this, decisions had to be made about which design decisions were being made in the rules versus which decisions were being made in the optimization routine. Ultimately, the gear parameter selection was removed from the grammar and placed in the optimization routine. The grammar rules have retained the decisions on configuration, material, and pressure angle. A reason for this decision is that gears are usually manufactured as a family of gears with similar material, pressure angle, tooth shape, etc. Making these changes in decision making allowed the grammar to expand to include a large variety of gears while keeping a relatively small branching factor and ensuring a faster overall process.
SEARCH PROCESS

The rules describe how the tree should be generated but the optimization can only optimize one topology at a time. In order to select that topology, a search method needs to be invoked. In theory, the most simple way to find the optimal solution is by using a breadth first search to search every candidate but this method is restrictive in both time and memory. The average branching factor for this problem is approximately the number of different gear families being used per configuration times the number of unique rules, which is 8 (2x4), for the test problems. If the candidates could be searched quickly, it may be feasible to be use a breadth-first search, but in this case, each valid candidate requires a time-consuming optimization of its included parameters which is too time and memory restrictive to be practical. In order to make the overall process fast (at least as fast as a human designer), we have adopted a guided search process.

A guided search process uses the search time more efficiently because it sorts the candidates on the tree by how well they satisfy the posed design problem. In this case the objective function is a combination of two factors: mass and efficiency, and these are used to determine which candidate is a better solution. The candidates are sorted according to the objective function with the candidate having the lowest combination of mass and highest efficiency being the one searched first. The two are normalized so that
they are on the same scale and then weighted evenly. This weight may be optimized in later iterations. The mass component ensures that the lightest gears are searched first

while the efficiency ensures that the solutions with the fewest gears are considered irrespective of the weight. The lighter the gear train, the cheaper it will be to operate and the more efficient it is the less likely it is to fail. If the candidate selected to be searched does not satisfy the constraints, then its children are made, using the recognize, choose, and apply cycle, as described previously, and the stack of candidates is sorted again. Because mass monotonically increases as we move down the tree while efficiency monotonically decreases, we are still able to find the optimal solution based on the theory of uniform cost search even though every candidate has not been optimized

(Dechter & Pearl, July 1985).

**Parametric Setup**

When the gear train candidate is considered, it must be optimized and checked to ensure that it meets all of the constraints. In this optimization, the basic computational structures are a vector of design variables (\(x\)), a matrix of state variables (\(Y\)), and a vector of location matrices (\(Z\)). The vector \(x\) stores the number of teeth, face width, pitch, and a location variable for each gear. The location variable varies depending on how the gears are connected to each other and an explanation is shown in Figure 7. For instance, if a gear is mating with another gear then the location variable describing this location is the angle in between the two gears as shown on the right of Figure 7. Since mating spur gears
are in a two dimensional plane, and assuming the configuration shown in the figure, the x and y locations would change as a function of the angle in between the gears. The z location for these mating gears is set because in order for the gears to be mating they

must have the same z location. The solid line represents a bounding box and the dashed line shows a gear that violates this constraint. The location variable, theta, is changed in order to keep the gear within the bounding box. This angle can be used to define mating pairs of spur gears, bevel gears, and worm gears.

**Figure: Explanation of Location Variables**

In the case where gears share the same shaft the x and y location are set but the z location can vary, as the length along the shaft is the only allowed degree of freedom as shown on the left of Figure 7. As a result, n location variables are needed for n gears assuming the first gear is positioned at the input and the output shaft length is an additional degree of freedom. Additionally each of the n gears has design parameters for face width, number of teeth, and pitch which result in 4 x n decision variables. The matrix Y contains all of the dependent parameters and state variables including weight of the gear, the force at the gear teeth, the type of gear, the speed, the x, y, z locations, and

the x’, y’, z’ components of the unit vectors to fully define the location for each gear. Thus, Y is an n x 10 dimensional matrix.
The location matrices \( Z \) are locally defined and are calculated using homogeneous transforms to find the position of each gear. Homogeneous transforms generate a 4x4 matrix for a point which contains all position and orientation information. The transform takes the original points’ matrix, performs a translation and a rotation upon it by multiplying matrices and the result is a 4 x 4 matrix describing the location of the translated and rotated point

(Craig, 1986). The information that is most relevant in describing the position of the gear is contained in the first three rows of the final two columns. The third column contains the x, y, and z components of the unit vector which allows us to find the direction that the face of the gear is facing. The fourth column of the 4x4 matrix contains the x, y, z, location for the point allowing us to find the position in space of the center of the gear. Once the transform is complete, the most relevant information is stored within the local variables of the nodes.

**Constraints**

The design vector \( \mathbf{x} \), is optimized using an optimization algorithm while keeping in mind a number of \( h \) (equality) and \( g \) (inequality) constraints. The constraints depend not only on the design variables but also on the calculated state variables, \( Y(\mathbf{x}) \). When the optimization method requires evaluation of a particular candidate point, \( Y(\mathbf{x}) \) is first calculated. The constraints are designed to replicate constraints that a designer would implement when designing the gear train by hand such as the desired output speed,
ensuring the mating gears have the same pitch, the stress on the gears, the desired
output location, and a bounding box constraining the gear train. For the most part these
are the most significant and common constraints. Depending on the application, gear train
design may also be concerned with minimizing vibration, noise, and rotational inertia;
these are left for future work.

**Output Speed Constraint**

The output speed constraint is an equality constraint which checks the speed of
the last node and sees if it is equal to the user defined output speed. If the speed is within
a designated percentage (e.g. 5%) then the constraint is satisfied. The speed of the gear is
one of the variables stored in the state variables matrix \(Y\), described above. As the gear
train is put together, the speed changes based on the configuration of the gears and can be
easily accessed to check this constraint.

\[
\text{hosc}(x, Y(x))=0, \text{ iff } 0.05 > \text{soutput-starget}
\]  \hspace{1cm} (1)

**Same Pitch Constraint**

Another consideration when building a gear train is that the pitch between mating
gears must be the same. If the pitches for gears are not equal then they will not mesh
efficiently. As described earlier the gear which is in contact with another gear is tagged
with the label *contact*. This label provides a marker for identifying where the constraint is
called and allows for a simple calculation for the number of same pitch constraints in a
given configuration, one for each tag. If the pitches are not equal then the constraint is not satisfied.

\[ hP_x = P_i - P_j = 0 \] (2)

**Stress Constraint**

A more difficult constraint for a designer to keep in mind is the stress that each gear will encounter. For the spur gears, there are two accepted equations for describing stress, one being the bending stress equation and the other the surface stress. The bending stress equation was originally developed by W. Lewis in 1892 and is the basis for the American Gear Manufacturers Association (AGMA) version that is implemented here {AGMA Standard 2001-B88}. This equation takes into account fatigue fractures from the fluctuating bending stresses at the root of the tooth and is shown as \( \sigma_{bs} \) in equation 3. A more in-depth discussion of the stress equation used and the variables (shown in formulas 3-6) can be found in a machine design textbook (Norton, 2006).

\[ \sigma_{bs} = WtpdKamKsKBKIFJKm \] (3)

To calculate the stress on bevel gears is more complicated than spur gears. The equations used for these calculations represent a simplification and should be checked with a more thorough calculation if safety and stresses are of particular concern. The bending stress for bevel gears is shown as \( \sigma_{bb} \) in equation 4 and is found in {ANSI/AGMA 2003-A86}. 42
\[ \sigma_{bb} = 2T_{ppd}K_aK_mK_{sd}F_{JKKx} \]  \hspace{1cm} (4)

The second stress being calculated is the surface stress shown as \( \sigma_{cs} \) in equation 5. The surface stress accounts for surface fatigue or pitting of the tooth surfaces. If properly lubricated then the gears will not have a surface failure by adhesive, abrasive, or corrosive mechanisms but will ultimately succumb to pitting and spalling due to surface fatigue. The equation used to check the surface stress is the AGMA pitting resistance formula also defined in \{AGMA Standard 2001-B88\}.

\[ \sigma_{cs} = C_pW_tC_aC_mC_sC_fF_{ld}C_v \]  \hspace{1cm} (5)

The surface stress in bevel gears is calculated in a similar manner but there are additional adjustment factors and this equation is shown as \( \sigma_{cb} \) in equation 6. The standard from which this equation is derived is \{ANSI/AGMA Standard 2005-B88\}.

\[ \sigma_{cb} = C_pC_b2T_dC_aC_mC_sC_fC_xC_{ld}2C_vT_pT_Dz \]  \hspace{1cm} (6)

Both of these stresses should be calculated when performing a design because if designed with these stresses in mind, the gear train should not fracture a tooth in normal service from bending and it can be predicted when pitting will cause a failure. After determining \( Y \), these stresses are automatically calculated for each gear in the candidate solution. If a gear is found to have less than a safety factor (Nf) of two then it is rejected and a new topology is optimized using equation 7.

\[ g_{stressx}, \ Y_x = N_f - (strengthstress) \leq 0 \]  \hspace{1cm} (7)
OUTPUT LOCATION CONSTRAINTS

There are two constraints which manage spatial considerations. The first constraint is an equality constraint which checks the last gear and looks for its location (8). This location is matched against the user defined output location and if it is within one percent then the constraint is satisfied. The second location constraint is an inequality constraint, which keeps the gear train within a bounding box (9). This constraint is useful because when designing a gear train, a designer may want to keep the gear train confined in a certain area such as when designing a gear box of a toy car. The user defines a maximum x, y, z, and a minimum x, y, z for the box. To satisfy this constraint, each gear’s location is checked against the bounding box. Being an inequality constraint, if the value of the constraint is less than zero than the constraint is satisfied and all of the gears are within the box. If the value of the constraint is positive a portion of one of the gears lies outside of the user defined box.

\[ \text{holc}(x, Y(x)) = 0, \text{iff } 0.01 \geq x_{\text{output}} - x_{\text{target}}^2 + \ldots \] (8)

\[ \text{gbbx}, Y_{xi} = 1 = \max x_{\text{geari}} + r_{\text{geari}} - x_{\text{bb}} + x_{\text{mm}, \text{bb}} - \min x_{\text{geari}} - r_{\text{geari}} + \max y_{\text{geari}} + r_{\text{geari}} - y_{\text{bb}} + \ldots \] (9)
**OBJECTIVE FUNCTION**

The optimization is run with these constraints while minimizing the objective function: the overall mass of the gear train combined with the efficiency shown in equation 10. The mass of each gear is calculated using the density and the geometry of the gears and the efficiency is based on the type of connection. Two spur gears have an efficiency of 98 percent, two bevel gears have an efficiency of 90 percent, and a worm gear pair has an efficiency of 70 percent. The value of the efficiency will be between 0 and 1, with 1 being 100 percent efficient. When the inverse of the efficiency is calculated, larger values are obtained as the gear train becomes less efficient. The objective in this case is to minimize the objective function so the objective function as shown, will be minimized when efficiency is 1. The mass of each gear is calculated and divided by 100, the highest mass that is reasonable to obtain, in order to scale the mass to be on the same order of magnitude as the efficiency.

\[
f(x, Y(x)) = \sum_{i=0}^{n} \rho_{FW} r_2 \pi^{100+1} \text{efficiency}
\]  

Using mass as an objective, will cause the system to select the candidate with the lightest and smallest gears while efficiency ensures that number of gears is minimized. This translates to cost savings when building the gear train. Typically plastic gears will be lighter and less expensive than steel or brass gears while having smaller gears is
usually cheaper because of the reduced amount of material. In addition, a lighter gear train will be cheaper to operate because of the lower moment of inertia; Less power input to keep the train in motion and less to initially get it started. On the other hand, a light plastic gear train may not have the strength required and would break under the stress of operation so some of the constraints bound the mass from the bottom.

**Exhaustive Catalog Search**

Solving the complete decision vector \( \mathbf{x} \) at once requires the numerical optimization of both discrete and continuous variables. As described earlier each gear is specified by the number of teeth, the pitch, the pressure angle, the thickness or face width, and where the gear is positioned in space. It is assumed that the gears are purchased from an original equipment manufacturer (OEM) and not custom made as custom gearing is both time and cost prohibitive. As a result, prior to running the search process a catalog of OEM gears is provided. The gears are taken from the McMaster Carr catalog online:

(McMaster Carr, 2008), which contains five families of spur gears: plastic injection-molded gears with a pressure angle, \( \Phi \), of 14.5 degrees; steel gears (\( \Phi = 14.5^\circ \)); stainless steel (\( \Phi = 20^\circ \)); steel (\( \Phi = 20^\circ \)); and brass gears (\( \Phi = 20^\circ \)). It also contains the worm made of steel with right hand threads, the worm gears made from cast iron with a plain bore, and two families of bevel gears: molded nylon and steel with a plain bore. As
discussed earlier, the grammar rules are partly enumerated by the choices between
different gear families. This means that the choice of pressure angle is no longer a
concern of this sub-optimization step. Furthermore, the choice of number of teeth, pitch
and face width must conform to elements in the catalog. The nine gear families account
for 204 unique gears as is shown in the McMaster-Carr catalog. Therefore, it is not
necessary to independently search for these variables in the optimization process, as the
valid combinations represents a much smaller set. Since each family has on
average 23 possible gears (204/9), it is possible to perform an exhaustive search for the
optimal within a gear train up to topologies with nine gears. Exhaustive search is
obviously a slow search method, but the guarantee that it provides in the optimal solution
as well as its robustness and lack of adjustments make it useful as an automated sub-
optimization process.

Prior to the decision to use an exhaustive search, various optimization methods
were pursued. Stochastic methods like genetic algorithms

(Goldberg, 1989) and simulated annealing

(Kirkpatrick, Gelatt Jr, & Vecchi, 1983) are attractive for their abilities to handle
complex discrete problems such as this, but the poor guarantee in the solutions they
produced complicated the larger search for optimal topologies. One significant hurdle in
the choice is the fact that these optimization methods must be invoked automatically from
the larger search process, and the values for their input parameters (“knobs”) must be
automated as well since different topologies have different numbers of decision variables.
Also, it should be noted that the space is not particularly multi-modal, thus such stochastic optimization methods are over-engineered for this problem type. The constraints described earlier in the section are all explicitly provided to the process as well, and the best optimization method used should also be able to handle constraints explicitly (and not convert them penalty functions). Prior to the decision to use OEM gears, the use of a sequential quadratic programming (SQP) method

(Lawrence & Tits, 2001) as part of a branch-and-bound

(Lawler & Wood, 1966) process was attempted to optimize the number of gear teeth, pitch and face width of each gear. Results

were mixed and not robust, but this method may be re-examined in cases where custom gearing can be utilized (constraining the process to valid combinations of number of teeth, pitch and face width is tedious but doable).

**Tuning of Location Parameters**

The aforementioned exhaustive search results in candidates that satisfy the pitch and speed constraints. However, the output location constraint and the bounding box constraint require knowledge about where the gears are positioned. At this point in the optimization each gear has three defined variables (number of teeth, pitch, and face width) with one left to define (location). The topology defines the location variable as either an angle or a position along a shaft – both of which are non-discrete. Because of this, it is much easier to run a simple gradient-based optimization algorithm to find the
remaining decision variables. Since the design variables affect different constraints, it is possible to break the optimization into separate discrete and continuous sub-optimization problems. This saves time, and ensures that each topology has been optimally tuned.

The gradient-based search method for the n location variables is accomplished by a Fletcher-Reeves (Fletcher & Reeves, 1964) approach to determining search gradients combined with an arithmetic mean line search (Onwubiko, 2000). These methods were implemented specifically for this application, but the authors have taken care to separate the optimization code into a separate generic toolbox for use in other applications. shows a section of code used to set up the optimization routine. The object-oriented approach that we adopted allows us to easily change the qualities of the optimization routine. In the first line of code, a gradient based optimization framework is initiated and named “optmethod”. In line 2, the arithmetic mean approach to performing line search is created and added to the optimization method (with values defining the local convergence of the arithmetic mean). Line 3 then initiates and adds the Fletcher-Reeves approach to determining the search direction. Finally, line 4 injects a simple process to determine when the algorithm has converged (either after 200 iterations, or if the difference in objective function values is below 0.0001). In addition to arithmetic mean, we have implemented and experimented with Golden Section
(Wilde & Beightler, 1967) and DSC-Powell

(Swann, 1964); these have not been as effective. Additionally steepest descent and BFGS methods have been created for determining search direction, but the decision to use Fletcher-Reeves is due to its robustness and ease of calculation.

When the algorithm reaches one of the stopping criteria the candidate has been fully optimized. This optimized candidate is checked to see if it meets the output location to within one percent and if it does the algorithm is finished and the user is presented with the optimal candidate. If it is not met, then the exhaustive search continues and a new candidate which meets the discrete constraints is optimized and checked against the output location. The reason that the stopping criteria can be based solely on the output location is that if the candidate gets to that stage it already has satisfied the remainder of the constraints.

Figure 9 shows a summary of the three stage search process. In the first step, the user inputs the parameters required for the optimization. The tree search is then performed based on the grammar, followed by the discrete optimization to set the number of teeth, pitch, and face width. Then the first question is asked; Are the pitch, output
speed, and stress constraints met? If the answer is no, then the algorithm goes back to the exhaustive search to try a new combination or if every combination has been attempted it goes back to the tree search to try a new topology. If the answer is yes, then the algorithm goes on to the continuous, gradient based optimization where the location variable is set. At this point, the question is whether the bounding box and output constraints are satisfied. If they are not, then the algorithm goes back to the tree search to try a new topology. If these constraints are all satisfied then the optimal (i.e. lightest and most efficient) solution has been found. This is nearly a guarantee of the global optimum since the objective function and constraints are all monotonic in the three search methods. At this point, the solution is presented to the user.
CHAPTER 6: RESULTS AND DISCUSSION

USER STUDY

To test the validity of the new method, we ran a test using undergraduates who were in a machine elements class. A simplified problem was developed for the class for extra credit which duplicated what the computer is capable of doing and the complete problem is shown in Appendix A. For the purpose of this study we put a limitation on the possible types of gear trains to stay within the limitations of what the students had worked with in class. The students were asked to design a gear train to transform the speed from 100 rpm to 700 rpm within five percent and the location from a given input \{0, 0, 0\} to a target output location \{3, 1, 6\} using simple or compound combinations of gears. The students selected gears from a catalog of thirty-two different spur gears. In the design, the students had to keep in mind the overall objective to minimize mass and the constraints that were discussed earlier namely: stress, location, speed, and bounding box.

The results obtained from the students were extremely varied. In this design experiment we had the students calculate both the bending and wear stress but in the machine elements class the students were in, calculating plastic gear stresses was not emphasized. Despite this, the results of the study were very good. Thirty percent of the students obtained very good results which would be acceptable in a design situation. The best student was within 0.05 pounds of the optimal, if the gear wear stress calculation
was ignored. Thirty percent of the students obtained no feasible results or were unable to meet the speed constraint within the defined parameters. The middle 35% of the students met various constraints, had a hard time doing the bending stress calculation, or developed a gear train that was very far from optimal. The average time spent on this problem by the students was around four hours and a summary of the students results are shown in Appendix B. This problem was provided to our automated tool and the computer spent one minute and twenty seconds solving the problem and came up with a solution that was more optimal solution than the students and considering both stress constraints. The GraphSynth representation of the result is shown in Figure 10.

The graph shows how the gears are connected but the gear parameters are embedded within the nodes and arcs and not shown in the graph. In this instance, the gears on the left of Figure 10 are a mating gear pair because Gear1 has a label of contact.

Gear 14 has the label common which shows that it is on a common shaft with the previous gear and Gear16 is in the same configuration as Gear1.

The results that we have obtained are difficult to understand without some type of graphical representation. A visualization tool has been developed (based on (Holkner, 2008)) to transform the graph output shown in Figure 10 into the three-dimensional rendering shown in Figure 11. The input to the visualizer is an automatically
generated CSV file that includes the location of the gears and shafts and this is interpreted by the visualizer to show the gear configurations. The gear train shown in the graph is imported into the gear visualizer and the output is shown in Figure 11. The gear in the graph with the first label is shown in Figure 11 as the input and the gear in the graph with the new label is shown in the rendering as the output gear. At this point, the visualizer does not yet include bearings and individual shafts other than those which connect two gears.

The input and output gears are labeled and the output gear has a center at (3, 1, 3) and the output shaft moves the output location to (3, 1, 6). Table 1 shows the gears used in the optimization with some of the included parameters. Three of the gears are metal and gear one (48 teeth) is made from plastic. All of the gears satisfy the stress constraints with a safety factor of two and the final weight of all gears is 1.049 pounds. This result is the lightest gear train which can satisfy the stress constraints and none of the students were able to obtain this optimum. The computer was able to meet the speed target and get the output location within tolerances which is something some of the students discussed struggling with. Keeping these constraints in mind, the computer was also able to attain the lowest mass which was simply too complicated for a designer to fully consider when solving this problem.
As discussed earlier, the students had a difficult time calculating the gear stresses so a test was run ignoring this constraint to better judge how close the student subjects could get to the optimal solution. When this constraint was ignored a gear train was developed which was the essentially the lowest mass that would meet the target speed and location. This gear train had an output speed of 720 rpm using gears of 48, 16, 60, 25 and a weight of .4366 lbs and is shown in Figure 12. In this design, the first three gears are plastic while the fourth gear is made of steel. Many of the students were very close to this optimal solution with one student coming within 0.5 pounds of this minimum weight. The main advantage of using this algorithm for this type of problem is being able to eliminate the hours spent by the designer tuning the gear train and performing the required calculations. As the solution begins to get more complicated it would take a user much longer to complete and the computer also begins to take longer to optimize the topologies.

<table>
<thead>
<tr>
<th>Gear</th>
<th>Location (in.)</th>
<th>Speed (rpm)</th>
<th>Number of Teeth</th>
<th>Weight (lbs)</th>
<th>Efficiency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gear 1</td>
<td>0,0,0</td>
<td>100</td>
<td>48</td>
<td>0.376</td>
<td>1.0</td>
</tr>
<tr>
<td>Gear 2</td>
<td>2.49,-0.27,0</td>
<td>400</td>
<td>12</td>
<td>0.222</td>
<td>0.98</td>
</tr>
<tr>
<td>Gear 3</td>
<td>2.49,-0.27,3</td>
<td>400</td>
<td>35</td>
<td>0.340</td>
<td>1.0</td>
</tr>
<tr>
<td>Gear 4</td>
<td>3.00,1.00,3.0</td>
<td>700</td>
<td>20</td>
<td>0.111</td>
<td>0.98</td>
</tr>
</tbody>
</table>

56
MODIFIED USER STUDY

The same test problem was run with all of the gear topologies included except in this iteration the output location had a change in angle. The input location was taken as (0, 0, 0) with input shaft in the x – y plane, rotating about the z-axis. The output location was set at (3, 1, 6) with the output shaft rotated in the X-direction 90 degrees, rotating about the y-axis. The program took a little under half an hour to run and the visualizer rendering is shown in Figure 13. The input gears are a set of bevel gears to change the angle and then two spur gear pairs are used to get the correct x, y, z location.

When the bevel and worm gears are added to the gear library, a completely different visualizer is used. This visualizer is based off of python bindings to the
OpenCASCADE library, an open source CAD API software suite for boundary-representation modeling. Through the use of pythonOCC (Paviot, 2009), an interactive visualizer as well as STEPexporter allows optimized gear trains to be imported into any other CAD system or project. This allows us to show the gears in any 3-dimensional orientation and does not limit us to using only spur gears as in the previous gear visualizer. This visualizer is incomplete so the pictures have been modified to add the shafts to better explain what is happening in the picture. The shafts were not added automatically and is an area for further improvement.

**CHALLENGING GEAR DESIGN PROBLEMS**

To present the computer with more challenging problems two difficult problems were presented and run. The first problem that was presented to the computer is one where the shafts are parallel to each other but the speed ratio goes from 30 to 1. Specifically, the input speed is 3000 rpm and the target output is 100 rpm and the torque on the motor is 1.5 ft-lbs. The input is at the origin (0, 0, 0) and the output is translated to point (3, 1, 4) but there is no change in angle of the output. This problem is interesting because it is unclear if a better solution is to use a worm gear pair to obtain the drastic decrease in speed and then to turn the output with a pair of bevel gears or if the optimal
solution is to have a series of compound gear pairs to obtain the speed reduction. The tradeoff in this case is the efficiency of the process. The worm gear is relatively inefficient when compared to spur gears but the worm is very effective at drastically decreasing the speed.

The algorithm ran for one hour and twenty minutes until it came up with the optimal solution. The solution that was presented is a worm gear pair followed by a bevel gear pair to turn the output and then a spur gear pair to get the output in the correct location. The GraphSynth display of the solution is displayed in Figure 14. The input shaft is labeled input and the output gear has the new label.


This solution was run in the new gear visualizer and the result is shown in Figure 15. The gears have been simplified and are represented by disks and the rods were drawn in after the image was produced to make it easier to understand. The worm is the input off of the shaft and is shown at the right of the image and as Gear 1 in Table 2. Table 2 shows a summary of the gears used and the various properties associated with these gears. The final output speed of the process is 91.67 rpm and the last gear has an output
location of (3.00, 1.02, 2.12). The final shaft translates the output point to (3.00, 1.02, 4.00). The efficiency of the gears is multiplicative so the overall efficiency of the process is 62 percent assuming that there is no energy loss when two gears on the same shaft.

**Table: Gears Used in Speed Reduction Solution**

<table>
<thead>
<tr>
<th>Gear</th>
<th>Location (in.)</th>
<th>Speed (rpm)</th>
<th>Number of Teeth</th>
<th>Weight (lbs)</th>
<th>Efficiency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gear 1</td>
<td>0,0,0</td>
<td>3000</td>
<td>N/A</td>
<td>0.38</td>
<td>1.0</td>
</tr>
<tr>
<td>Gear 2</td>
<td>0,1.335,0</td>
<td>166.67</td>
<td>18</td>
<td>0.56</td>
<td>0.70</td>
</tr>
<tr>
<td>Gear 3</td>
<td>3.96,1.34,0</td>
<td>166.67</td>
<td>24</td>
<td>0.003</td>
<td>1.0</td>
</tr>
<tr>
<td>Gear 4</td>
<td>3.71,1.34,0.2</td>
<td>166.67</td>
<td>24</td>
<td>0.003</td>
<td>0.90</td>
</tr>
<tr>
<td>Gear 5</td>
<td>3.71,1.33,</td>
<td>166.67</td>
<td>11</td>
<td>0.003</td>
<td>1.0</td>
</tr>
<tr>
<td></td>
<td>2.13</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Gear 6</td>
<td>3.00,1.02,2.1</td>
<td>91.67</td>
<td>20</td>
<td>0.111</td>
<td>0.98</td>
</tr>
</tbody>
</table>

The second difficult problem presented to the computer is a problem where the output is not perpendicular to the input but at a skew angle. The input is at the origin and the speed transform is from 200 rpm to 800 rpm. The output location is translated to point (2, 2, 4) and rotated -40° in the Y direction as seen from the origin. The algorithm spent one hour and ten minutes running this problem and the rendering of the solution is shown in Figure 16.
The input to the system is two bevel gears shown to the bottom right of Figure 16 and the output is the final spur gear on the left of the figure. Table 3 details the solution and the gear parameters. The speed has been transformed to 823 rpm which is within tolerances for the solution and the total weight of the solution is 1.55 pounds. All of the bevel gears chosen were made of steel while the two spur gears were made of plastic and are the same gears used in the student’s design problem. The final gear’s output location is (2, 0.94, 2.67) and the output shaft is fixed in the x direction and translates in the y-z plane to (2.0, 2.0, 4.0).

<table>
<thead>
<tr>
<th>Location (in.)</th>
<th>Speed (rpm)</th>
<th>Number of Teeth</th>
<th>Weight (lbs)</th>
<th>Efficiency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gear 1</td>
<td>0,0,0</td>
<td>200</td>
<td>18</td>
<td>0.510</td>
</tr>
<tr>
<td>Gear 2</td>
<td>0.75, -0.01, 240</td>
<td>15</td>
<td>0.298</td>
<td>0.90</td>
</tr>
<tr>
<td>Gear 3</td>
<td>5.0, -0.07, 240</td>
<td>16</td>
<td>0.167</td>
<td>1.0</td>
</tr>
<tr>
<td>Gear</td>
<td>Teeth</td>
<td>Diameter</td>
<td>Reduction</td>
<td>Tooth Width</td>
</tr>
<tr>
<td>------</td>
<td>-------</td>
<td>----------</td>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>0.167</td>
<td>0.90</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>48</td>
<td>0.376</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>14</td>
<td>0.032</td>
<td>0.98</td>
<td></td>
</tr>
</tbody>
</table>
Current research in gear train design involves optimizing the shape of gear teeth and finding ways to manufacture smaller gears. In industry, most effort spent in gear train design is in trying to meet simple constraints and obtain a design that will not fail. As shown in the human studies, a designer can spend up to 11 hours to fully develop a solution to a relatively simple design problem. When the design becomes more complex and the user has a difficult time imagining the gear train in 3-dimensional space, it also becomes more difficult to optimize the parameters and variables required. This difficulty is remedied by the efforts presented in this research.

After the parameters have been set, this fully automated design effort is ready to be used in industry. The results include many meaningful constraints and the gears can be customized and ordered from an OEM supplier. All of the different gear configurations with the corresponding variables are taken into account and solved by the developed optimization method. Every possible configuration of simple, compound, bevel, and worm gears is searched and an optimal solution can be discovered. The generation of these configurations leaves millions of solutions to be searched through. The algorithm is able to consider the best topologies and optimize the variables considering the constraints: mating gears must have the same pitch, the output location required, the bounding box containing the gear train, the stress on each of the gears, and
the output speed of the system. The user is then presented with a 3-D image of the gear train.

It is important to note that the objective function and all of the constraints related to gear choice are nearly monotonic. This fact, taken with the best first and exhaustive search, means that we can guarantee that the global optimum is found. The location constraints \( h_{loc} \) and \( g_{lab} \) and variables \( z \) along shaft, or angle about a gear) appear to produce a smooth and convex search space. Thus, if the gradient-based method is sound then we can claim an optimal topology is found by our process.

**Future Work**

While the results shown here are promising, the design is not able to be implemented without further work. A designer using this tool must take the topology generated by the algorithm and take into consideration lubrication requirements to fully design the system. This could be incorporated into future work in order to completely automate the entire design process. Other limitations are that the types of gears allowed do not include helical or epicyclic gear trains and the system is currently a single output system. This work could be expanded to include these added gear configurations and to allow for multiple outputs or inputs.

In order to make the current work more accurate, the stress constraints may be refined and more accurately calculated instead of presenting a simplified approximation.
This software that has been developed can be used to export a CAD file that includes the gears and the forces on each gear. Finite element analysis software can then be used to more accurately calculate the stresses seen on each gear and predict the failure. Another constraint that can be modified is the bounding box which is at this point is a rectangular prism but can be expanded to a more complicated 3-D shape to represent more varied geometries. All of these additions are possible to the current software however, the optimization must be written to be more efficient so that each topology does not take as long to optimize.
APPENDIX A: EXTRA CREDIT ASSIGNMENT

Design a gear train with an input speed at 100 rpm and a desired output speed of 700 rpm. This output speed must be within 30 rpm of your target. The input torque on the shaft is 50 in-lbs. and a safety factor of two is required for both bending and wear when deciding on which gears to choose. The first gear’s center is at (0,0,0) and the last gear’s center must be within 0.1 inches of the x, y, and z coordinates at (3,1,6). These locations are on a standard Cartesian coordinate system and the units are in inches. At (0,0,0) the gear is in the x, y plane and is rotating about the z axis. The entire gear train must fit within a bounding box where the maximum x and y can be is 5, the minimum they can be is -5, and the maximum z is 10, and the minimum z is 0. The overall objective is to meet these constraints with the lightest (lowest mass) gear train possible. There is no need to design the bearings or shafts in this problem. A graph is shown below helping to explain the locations and the bounding box. For this problem list which gears you used and the location that they are located. You may either choose to show the location graphically or simply list the x, y, and z coordinates of the center of the gear.

Recall:

- The diameter is the number of teeth divided by the pitch
- Mating gears must have equal pitches

Please keep track of the starting and finishing time (this will not affect your grade):

Total Time = ________________

If there are any questions contact Albert Swantner at a.swantner@mail.utexas.edu. Return to Dr. Campbell by 4/10 @ 5pm (outside of office in wooden box).

Gear Catalog¹ (this page)

¹ www.mcmastercarr.com
<table>
<thead>
<tr>
<th>Teeth</th>
<th>(in.)</th>
<th>Plastic Gears:</th>
<th>Steel Gears:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td><em>Sfb = 5000,</em></td>
<td><em>Sfb = 40000,</em></td>
</tr>
<tr>
<td>60</td>
<td>20</td>
<td>0.375</td>
<td>20</td>
</tr>
<tr>
<td>16</td>
<td>16</td>
<td>0.5</td>
<td>20</td>
</tr>
<tr>
<td>24</td>
<td>16</td>
<td>0.5</td>
<td>20</td>
</tr>
<tr>
<td>32</td>
<td>16</td>
<td>0.5</td>
<td>20</td>
</tr>
<tr>
<td>48</td>
<td>16</td>
<td>0.5</td>
<td>20</td>
</tr>
<tr>
<td>14</td>
<td>12</td>
<td>0.75</td>
<td>20</td>
</tr>
<tr>
<td>48</td>
<td>12</td>
<td>0.75</td>
<td>20</td>
</tr>
<tr>
<td>12</td>
<td>10</td>
<td>1</td>
<td>20</td>
</tr>
<tr>
<td>20</td>
<td>10</td>
<td>1</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td></td>
<td><em>Sfc = 50000,</em></td>
<td><em>Sfc = 155000,</em></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.015</td>
<td>0.111</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.035</td>
<td>0.173</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.062</td>
<td>0.250</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.141</td>
<td>0.340</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.032</td>
<td>0.444</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.376</td>
<td>0.666</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.045</td>
<td>0.166</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.125</td>
<td>0.701</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.062</td>
<td>0.375</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.032</td>
<td>0.077</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.376</td>
<td>0.666</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.045</td>
<td>0.166</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.125</td>
<td>0.701</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.062</td>
<td>0.375</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.032</td>
<td>0.077</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.376</td>
<td>0.666</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.045</td>
<td>0.166</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.125</td>
<td>0.701</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.062</td>
<td>0.375</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.032</td>
<td>0.077</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.376</td>
<td>0.666</td>
</tr>
</tbody>
</table>
## APPENDIX B: RESULTS OF STUDENT TEST

| Student 1 | No Valid Solution, 7 hours |
| Student 2 | Good Solution, not lowest mass, 5 hours |
| Student 3 | No Valid Solution, 5 hours |
Student 4
Final solution not optimal, 2.5 hours

Student 5
Incorrectly calculated speed, 3.5 hours

Student 6
Good Solution, Incorrect Geometry, 3.75 hours

Student 7
Exact Speed, Not lowest mass, 4 hours

Student 8
Miscalculated Speed, No Location Optimization 2.5 hours

Student 9
Geometry Not Possible, Too many gear pairs, 4 hours

Student 10
No Valid Solution, 2.5 hours

Student 11
Did not use catalog of gears, miscalculated stress, 3 hours

Student 12
Too many gears included, Incorrect Stress, 2.5 hours

Student 13
Trouble Incorporating speed and location, 9 hours

Student 14
Incorrect stress calculation, too many gears used, 3 hours

Student 15
Not optimal solution, too many gears, .6.5 hours

Student 16
Incorrect stress calculations, 5 hours
Student 17
Good Solution, close to optimal, 3 hours
Student 18
Unable to calculate stress, 11 hours
Student 19
Did not perform stress calculations, 4 hours
Student 20
Incorrect stress, too many gears, 4 hours
Student 21
Too many gears used, 5 hours
Student 22
No valid solution, 4 hours
Student 23
No valid solution, 3 hours
Student 24
Incorrect stress calculations, solution not optimal, 4 hours
Student 25
Incorrect stress calculations, 4 hours
Student 26
Good Solution, too heavy, 3 hours

Belegundu, A., & Chandrupatla, T. Optimization Concepts and Applications in Engineering.


VITA

The author was born in San Antonio, Texas on April 30, 1985. The author’s parents are William and Cheryl Swantner and a sister named Tiffanie. They currently reside in San Antonio, Texas. The author went to high school in San Antonio at John Marshall High School and then attended The University of Texas at Austin where he obtained his Bachelor’s Degree in Mechanical Engineering in May of 2007. He then continued his schooling at The University of Texas where he will obtain his Master’s Degree upon acceptance of this thesis. During his time at The University of Texas, Albert was a teacher’s assistant for ME 366J working with Dr. Seepersad. He also worked at Flextronics Medical – Avail during the summer of 2008 which is where he will begin work full time in July of 2009.

Permanent address: 8507 Skye Heights, San Antonio, Texas, 78254

This dissertation was typed by the author.