AN INTEGRATED ENVIRONMENT FOR CONCEPTUAL DESIGN, SYNTHESIS AND ANALYSIS OF DYNAMIC FRAME STRUCTURES

by

Nathan Daniel Mead

A dissertation submitted to the faculty of
The University of Utah
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

Department of Mechanical Engineering

The University of Utah

August 1998

Copyright © Nathan Daniel Mead 1998

All Rights Reserved

THE UNIVERSITY OF UTAH GRADUATE SCHOOL

SUPERVISORY COMMITTEE APPROVAL

of a dissertation submitted by

Nathan Daniel Mead

This dissertation has been read by each member of the following supervisory committee and by majority vote has been found to be satisfactory.

	Chair: Stephen C. Jacobsen
	Samuel H. Drake
	Richard F. Riesenfeld
	Fraser Smith
	Charles L. Thomas

THE UNIVERSITY OF UTAH GRADUATE SCHOOL

FINAL READING APPROVAL

To the Graduate Council of the University of Utah:

I have read the dissertation of Nathan Daniel Mead in its final form and have found that (1) its format, citations, and bibliographic style are consistent and acceptable; (2) its illustrative materials including figures, tables, and charts are in place; and (3) the final manuscript is satisfactory to the Supervisory Committee and is ready for submission to The Graduate School.

Date

Stephen C. Jacobsen
Chair, Supervisory Committee

Approved for the Major Department

Robert B. Roemer
Chair/Dean

Approved for the Graduate Council

Ann W. Hart
Dean of The Graduate School

ABSTRACT

A computer-aided-design (CAD) environment was designed and implemented for the initial design and analysis of large dynamic structures. The design of these types of structures represents a difficult and time consuming task with little support provided by existing CAD packages. The results of this work include:

- Algorithmic synthesis of complex, dynamic three-dimensional (3-D) structural space frame geometry from initial design specifications.
- Methodology for dynamic analysis of open kinematic chains which is independent of specific joint trajectories.
- Dynamic analysis of the complete structure throughout the workspace, including computation of "worst case" loading conditions.
- Complete algorithmic construction of a Finite Element Analysis (FEA) model for each component of the structure, including conversion of the dynamic loads into a useful form.
- Discrete optimization of each link of the structure, including catalog lookup of existing beam sizes and heuristics to minimize the number of required finite element analyses.

The software algorithms developed during this project provides a unique set of CAD tools to enable design engineers to increase their productivity and allow more extensive exploration of the design space, yet still allow sufficient control to achieve reasonable, reliable, and manufacturable designs. The fundamental tools and support structure provided by these tools should also be highly useful in a much wider range of general mechanical design problems.

For my father, who has provided more love, understanding,
and support than any son could expect.

CONTENTS

LIST OF FIGURES

NOTATION AND SYMBOLS

The interdisciplinary nature and wide scope of this project made the selection of a reasonable and consistent notation scheme a difficult task. The end result of an evolutionary process provides a limited framework for consistency of the major elements; however, the large number of different symbols used precludes the construction of a comprehensive listing. Therefore, a higher level approach has been used to provide a reasonable level of consistency and understanding.

General variables are described with single characters, printed in standard italic font, lower case, with a subscript describing the instance. For example, the variable a_i describe the i^{th} instance of the variable a. Greek characters are used for angular measurements, with appropriate subscripts if needed. The scope of general variables should be considered quite limited and usually do not extend outside the current section.

Vectors and points are single characters, printed in bold font, lower case, again with a subscript describing the location or construction. An example of a possible vector from the point $\mathbf{p}_A$ to point $\mathbf{p}_B$ could be $\mathbf{r}_{A\to B}$. Some characters have special meaning, for example $\mathbf{u}_i$ describes a vector of unit length while $\mathbf{u}_{A\to B}$ would describe a unit vector in the direction of $\mathbf{r}_{A\to B}$. In addition the symbols $\mathbf{x}, \mathbf{y}$, and $\mathbf{z}$ are reserved for the the relevant coordinate axis. If discrimination among different coordinate frames is necessary, subscripts are added for clarity. Accessing vector or point components are described by enclosing the desired component in parenthesizes following the relevant object. Thus, the $\mathbf{x}$ component of the vector $\mathbf{r}_{A\to B}$ would be $\mathbf{r}_{A\to B}(x)$ and the $\mathbf{z}$ coordinate of the point $\mathbf{p}_k$ would be $\mathbf{p}_k(z)$. This component is often treated as a general variable and printed in standard italic font, with the component identifier added to the subscript, thus $\mathbf{p}_k(z) = p_{kz}$.

Single characters printed in upper case, bold font, are used to represent second order tensor variables such as inertia and stress. Included in this category are coordinate frames which, while not tensor quantities, have similarities and are extensively used throughout this project. The inertia tensor of the j^{th} object would be $\mathbf{I}_j$ and the coordinate frame of the same object could be $\mathbf{C}_j$ with other sub or superscripts added as may be necessary for clarity. Components of tensor quantities are described similar to points and vectors but use double component identifiers. Given the inertia tensor $\mathbf{I}_j$, the inertia about the $\mathbf{z}$ axis would be $\mathbf{I}_j(zz) = I_{jzz}$.

A two-character description using bold font has been used for most objects created for this project, with the first character upper case and the second lower. Again the subscript designates the instance, thus the j^{th} instance of a point load object would be $\mathbf{Ls}_j$ and the planar properties of the curve C_n would be $\mathbf{Pp}_n$. The Lisp slot accessor $->$ is used for describing a particular slot of an object type, thus the area of the above curve C_n would be $\mathbf{Pp}_n{->}A$. The scope of object descriptors is consistently maintained throughout the main body, not including the appendices.

ACKNOWLEDGEMENTS

The successful completion of this project required assistance and support from a large number of individuals and research groups. I would like to thank all the people who explicitly or implicitly made this work possible. In particular, thanks goes to:

Steve Jacobsen, Fraser Smith, Sarcos, and the Center for Engineering Design for providing great opportunities for industrial interaction and interesting projects, without which the concepts and development of this project would not have occurred. In addition, I would like to thank them for their continuing financial support over the course of a long and often slow project.

Richard Riesenfeld and the rest of the Alpha_1 group for providing access and support to a wonderful CAD development environment. Without such a foundation upon which to build, this project would not have been possible.

Sam Drake for providing constant assistance and allowing me the opportunity to learn a "useful skill" in addition to providing wonderful feedback.

Charles Thomas for his useful advice and suggestions throughout the limited time I have known him.

And Russ Fish whom I cannot thank enough for his patience, enthusiasm, and willingness to teach a mechanical engineer the fundamentals of software development. Without his constant assistance in understanding the complexities of computational tools, this project would never have been undertaken.

A special thanks to my wonderful wife Nicki and my children, Aubrey and Ryan, for their patience and understanding over a long and difficult period of time.

And finally, I cannot thank my father Carver enough for his continuing love, understanding and support.

CHAPTER 1

INTRODUCTION

There has been significant change in the computation tools available to mechanical engineers, designers and detailers over the past decade. As the cost of computational power has decreased, much more sophisticated software has become available to most individuals working in the field, resulting in improved productivity and shorter design cycles[17, 153, 92]. Today the use of solid modelers is an established part of most organizations involved in performing product design. Although these tools have proven quite useful there still exists a large amount of frustration with their integration into the design process. There is often reluctance to use these tools early in the design process as the modeling effort is substantial and the design quite fluid. There are also problems with integration between the different functional descriptions as manufacturing, stress analysis, and dynamic analysis require radically different descriptions of the same components. This project was undertaken to further identify and provide potential solutions to these problems. To further define this complex problem, a brief examination of both the current state of computational technology and the mechanical design process will be presented.

1.1 Current CAD Technology

The first computer-aided design (CAD) systems, some of which are still in use, were designed to replace the drafting board and served as little more than a drafting assistant. The initial construction of a drawing with these systems often required more time and effort than manual drawings, but the ability to make modifications and plot new drawings was a sufficient advantage to promote their wide acceptance. These systems, like the manual drafting methods, provided

little to no support for insuring that the drawing was correct, requiring skilled individuals to correctly construct and maintain the information. The difficulty greatly increased when a number of components interacted with each other, as in large assemblies. Modification or replacement of a single component could require manual modification of all related parts, a difficult and error prone task. This led to a very high value being placed on a valid or proven design, and a great reluctance to change anything once the drawings were correct.

Despite these shortcomings, these tools have been widely adopted in industry to the extent that it has become quite rare to find drafting boards still used for mechanical drawings. This is understandable as several studies suggests that these tools have provided a substantial increase in productivity. Although the measurement techniques are questionable, the studies suggest an average increase in productivity of between 50%-300% for drawing creation and an average increase of around 800% for drawing revisions[102, 5]. The large difference between drawing creation and drawing modification is not unexpected as it is much easier to modify the electronic version and plot a new drawing than to mount, erase and modify the paper representation.

With the advent of solid modelers the problem of drawing validity was greatly reduced, as correct two-dimensional (2-D) views could be extracted directly from the solid model. Construction of the original model often required more time and effort than using a 2-D CAD system, but the ability to view the objects in a three-dimensional (3-D) representation and determine interference problems as well as correct 2-D view construction were sufficient to justify the additional resources. However, the problem of manual propagation of design changes remained until recently.

Some currently available CAD tools attempt to algorithmically assist the propagation of modifications of the solid models by what is termed *constraint based solid modeling*. In a constraint based system the construction methodology is retained, and when a modification is made the methodology is repeated with the new information. This is a very powerful capability and has greatly reduced the effort required

for propagation of design changes. However, constraint based modeling still suffers from problems with scalability and complexity. These can be best understood by considering each of the constraints as a restriction of the allowable design space. As the number of constraints increase, the probability that a modification will push some parameter outside the allowable design space also increases. Soon the point is reached when multiple modifications of the constraints are required for any significant change of the model. This problem can be reduced with better initial choices of model constraints but this just postpones the inevitable. Despite this fundamental difficulty, these systems are still gaining wide acceptance, for the problems with model validity are at least identified by the system, rather than requiring manual identification before correction.

These issues with scalability and complexity are not unique to mechanical design, and can be viewed as somewhat analogous to that recently faced by VLSI chip designers and software engineers. In these disciplines, the increasing complexity of the systems they were trying to create necessitated a change in the design approach. This generally involved development of a higher level description of the design that could then be algorithmically decomposed into the needed representation. As might be expected, this involved a loss of some low level control, but the improvements in productivity, ease of modification and scalability have resulted in these higher level descriptions and related design systems being widely adopted.

1.2 Design Process

When trying to construct better computation tools for assisting with the design process, it is helpful to examine the entire design process and if possible, identify the source of some of the difficulties. Although the design process can be looked at from many different viewpoints and there are many possible process representations, the following interpretation of the decomposition proposed by Pahl et al.[97] is sufficient to illustrate the fundamental process(Figure 1.1). Using this representation, the design process is broken down into four main components as follows:

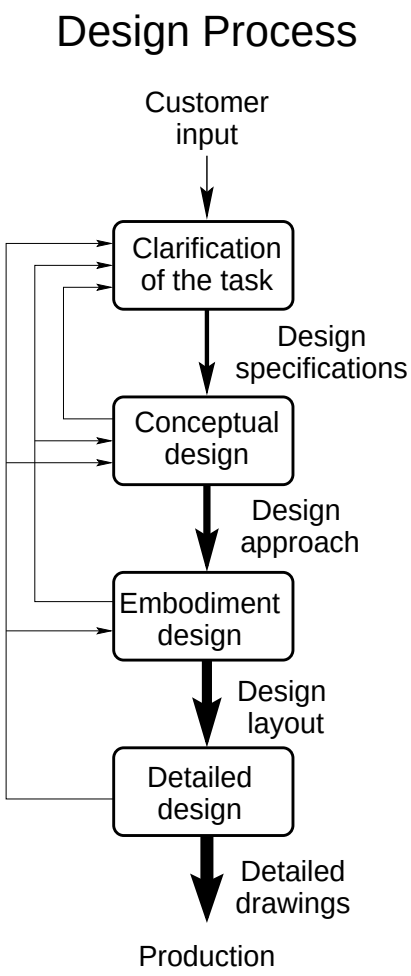

Figure 1.1. A flowchart of the design process. The line width is correlated to the volume of information transfer.

Clarification of the task during which information is collected about the task requirements and constraints. The resulting output of this phase is generally a detailed specification of the constraints and requirements of the design.

Conceptual design during which the fundamental solution approach to the problems posed in the specification are decided. The output of this phase is less well defined than the others, but it is generally an agreement upon an approach or set of approaches to the design problem.

Embodiment phase during which concept solution is sufficiently detailed to test the proposed design against the design constraints. The output of this phase

is usually a "layout" that specifies the critical geometry and components.

Detailed design phase during which sufficient detail is incorporated into the layout to allow for manufacture of the design. Typically, this involves including all catalog components not already specified during the embodiment phase as well as the determination of particular materials and exact geometry. This phase generally concludes with the generation of manufacturing drawings, bills of materials, and other manufacturing documentation.

From this general description, it is possible to make some observations about the design process. These include:

- It is an iterative process during which problems or other difficulties provide feedback to the previous stages. This iterative nature has a considerable affect on the amount of effort required to complete the design, as it involves a great deal of repetitive effort.
- The flow of information is principally one way. Although this seems to contradict the previous statement, examination of the feedback suggests that it mainly involves the inability of certain aspects of the design to satisfy the desired design constraints or requirements.
- The forward flow of information is cumulative. Additional information is added during each phase, much of which is passed on to the next phase.

The preceding decomposition of the design process is often referred to as procedural in that it follows the flow of information. An alternative decomposition can be performed in an orthogonal direction by looking at the various starting points possible for the design process[97]. Although there can be significant overlap between the categories, the different starting points can be described as follows:

Original design in which a novel and unique solution is undertaken to solve the design problem.

Adaptive design in which the design process involves adapting an existing or known solution to a new set of constraints.

Variant design in which variations of existing designs are constructed. Usually these are size variations and are limited such that the solution principle and functionality remain unchanged.

Surveys suggest that about 25% of designs can be considered original, 55% are considered adaptive and the remaining 20% variant[97]. These different starting points have a significant influence on the nature and amount of computational support that can be provided to the design process. For variant design the existing constraint based solid modelers work quite well for most reasonably simple problems. However, if an original design is undertaken the nearly infinite size of the design space greatly reduces the effectiveness of most of the current computational tools. The adaptive designs fall somewhere in between these two extremes, depending upon the complexity of the design and how much it differs from previous designs.

1.3 CAD Problems and Proposed Solutions

Considering the design process with current CAD tools, reasons for the difficulties with early integration of these tools into the design process are apparent[65]. Current CAD tools have evolved from use in the detailing phase and have worked their way into the embodiment phase. This "bottom up" approach was necessary given the requirement for handling a large amounts of detail, but implies that any significant modification to "upstream" information will render all past effort meaningless. Even without upstream modification, it is not unusual for the modeling process to be repeated many times before a satisfactory design is determined. For most reasonably complex systems, each of the individual components must first be individually modeled and then assembled before the whole design can be visualized, or analyzed. For example, before a design can be checked against an allowable stress constraint, the design must be close to finalized so that the geometry and boundary conditions are reasonably well known. This bottom-up philosophy precludes the effective use of current CAD and computer-aided engineering (CAE) tools during the early phases of the design.

This problem is often magnified when the design engineers are faced with an

implicit problem in which possible solutions must be found by repeated iteration. When these problems can be expressed mathematically, there are known solution techniques. However, the problems are generally of sufficient complexity that this approach is not possible. In these cases, the iteration process must be performed manually. Particularly with mechanical design, these types of problems require a large commitment of resources before the viability of any design can be determined.

A second problem arises from the desire to integrate the design and analysis models. There is a great deal of appeal in using the same geometric representation used in the design process for analytical purposes, thus eliminating the need to re-model the components. Despite the claims of many if not all of the CAD and CAE software vendors, the integration of reasonable CAD information into the analysis tools is still a goal, not a reality. The prevalent approach is to exchange low-level geometric information between the design and analysis subsystems via some interchange format, typically IGES. Unfortunately, the graphical information once exchanged is often of little to no value[151, 96]. Either there is too much detail for the desired level of analysis, or the desired information is not present. The former is very common when trying to perform dynamic analysis using CAD data, since the desired associations and rotation axes are either lost or difficult to locate in all the detail. The latter is quite common for nongeometric information, such as material properties and structural loading, but also occurs when nonsurface geometric information is needed. An example of such information is the centerline geometry for shell or beam analysis. There has been improvement in this area over the last few years, but problems are still the rule rather than the exception.

Yet another related problem arises from the poor integration between the various analysis packages. For example, the load information derived from a dynamic analysis cannot be applied to the structural analysis without substantial modification. This is due to the dynamic information representing a state of dynamic equilibrium at an instant in time, although the structural analysis is typically performed assuming static equilibrium. The dynamic load information must be modified to reflect these different assumptions as well as account for any differences

in the abstract representations used in the different analyses.

In working with current CAD and CAE systems, as well as discussions with mechanical designers and structural analysts, several other shortcomings of existing CAD/CAE systems have also become apparent. Some of these shortcomings are summarized as follows:

- Problems involving modification of existing designs. Despite the improvements that parametric and/or constraint based modelers have made in this area, as the complexity of the design grows, the space over which the design is valid decreases, requiring more and more effort to propagate design changes.
- Difficulties moving from conceptual to detailed design. In examination of the design process, the reasons for this are apparent, since the design space is quite large and the design solutions are unknown.

It was to address these specific problems, as well as uncover new problems, that this project was undertaken.

1.3.1 Desirable Features in CAD Systems

Over the years many of these problems have been observed, various system requirements suggested, and many solutions proposed. For example Bond et al.[18] suggest that in addition to the necessary drafting functionality, a good CAD/CAM system should also include the following:

1. Verification the well-formedness of a part.
2. Automatic transformation between different part representations: 2-D, 3-D, flat-pattern, etc.
3. Inclusion of constraints in the design.
4. Verification of the design against applicable standards.
5. Verification of design performance by simulation.
6. Integration of previous and standard designs.
7. Automatic detection of manufacturing features.
8. Automatic generation of process plans and CNC code for part manufacture.

And Akman et al.[2] describe their desire to have the following application programs coupled to a single database:

- Conceptual design systems to handle vague information.
- Consulting and problem-solving for engineering applications.
- Basic/detailed design systems coupled with geometric modelers.
- Engineering analysis programs such as finite element programs.
- Product modelers.

From a slightly different viewpoint, Watson et al.[152] describe the need for a structural engineering design system which includes the following:

- Inclusion of knowledge-based decision support systems.
- Display and detail-level control.
- System extensibility including user-defined parametric features.
- A single uniform data representation to ensure consistency.
- Inclusion of nongeometrical data.
- The ability to work at a higher level of abstraction than mere geometrical elements.
- Support for revision control.
- Integration to structural analysis and detailing applications.

Naoum et al. also describe a set of features desirable in a proposed CAD system[93] including the following:

- Inclusion of "Feature Technology."
- Simulation of all stages after the design phase to allow for optimization feedback to the designers.
- Integration of formal database functionality into the CAD environment.

During the years since these were published, a great deal of work has been performed to address many of these needs. Yet despite this effort, significant problems remain in most, if not all, of these areas.

From the previous problem descriptions, the above "wish-lists," and discussions with design engineers, several conclusions can be drawn. In no particular order, these include:

- Integration of the design specifications into the CAD model, preferably in parametric form, would be highly desirable.
- Data transformation between different types of analysis is often necessary, and is difficult, if not impossible, to perform without geometric information.
- Inclusion of non-geometric data into the system is desirable, both to include this information into the parametric design as well as to ease the integration process.
- Better interfacing between the CAD system and analysis packages is highly desirable.
- The system should be extensible to allow the designer to adapt it to their particular needs.
- Whenever possible, simple abstract representations should be provided to assist the conceptual design process as well as to provide a natural means of level-of-detail control.

1.4 Project Description and Scope

This project was undertaken in an effort to reduce or eliminate many of the previously mentioned problems encountered with current CAD/CAE systems. The first goal was to provide CAD tools that are useful early in the design process. Examination of the design process and the nature of the data being transfered in Figure 1.1 suggests the following:

During the "clarification of the task" stage, there is a high rate of change and a small need for geometric data. This suggests that any CAD tools that will assist with this process should be able to rapidly produce very simple abstract geometry.

Given a set of design specifications there are a very large, if not infinite, number of possible concepts that could potentially provide the desired functionality. One solution to this problem would be to arbitrarily select an initial design approach or domain from which the final design would be selected.

Much of the information in the design specification is not geometrical and may be related to the design indirectly through engineering analysis. Thus, any attempt to include this information in the CAD model must include some engineering analysis functionality.

One of the primary goals of this project was to propose and implement a different computational paradigm that more closely approximated the current design process, and allowed better integration of the CAD system into the early stages of the design process. A second goal was to reduce the engineering effort and related costs associated with design iteration and greatly increase the ability of a designer to explore the design space. Given the success of the higher level descriptions in the VLSI design area, it was felt that a similar approach could be of significant benefit to mechanical designers. It is possible to think of the design specifications and other designer input as a "high-level (source) language" and the geometric description of the system to be a "low-level (target) language"[150]. Then by analogy with computer language compilers, the system would be a "mechanical compiler." One of the primary goals of this project was to design and implement such a high level description of a complex mechanical system, as well as the necessary decomposition (compilation) algorithms.

To this effect, an integrated environment for the design of large, complex 3-D articulated structures was designed and implemented which incorporated existing solid modeling and finite element analysis (FEA) packages. Included in the project scope was automated quasi-static FEA model generation as well as discrete optimization of real systems including both structural members and actuators. Structural design aids including parametric modeling, intuitive design constraints, and intelligent constraint defaults were implemented and refined based upon feedback

from working engineers.

The class of problems selected for this project was the design of large dynamic structures, an example of which is shown in Figure 1.2. These mechanical systems can be described as a series of space frame structures connected by joints at which torques can be applied. Due to the number of potential dynamic load conditions, as well as potential external loads, the design and analysis of these types of systems can be extremely difficult and time consuming. This type of mechanical system occurs quite often in large robotic systems, as well as in other applications.

There were several reasons for this particular problem choice. The principal reasons are listed below:

- There was a need for these types of tools for both current and potential future projects. This provided not only the project area but also provided a baseline for the performance of conventional CAD/CAE tools applied to similar problems.
- There existed a pool of experienced designers who could be solicited for expertise. The probability of producing generally useful results and a useful system was greatly increased by input provided by design engineers actively working on similar problems.
- The problem could be generalized to include a wider domain. Although a certain percentage of the work is necessarily dedicated to a narrow domain,

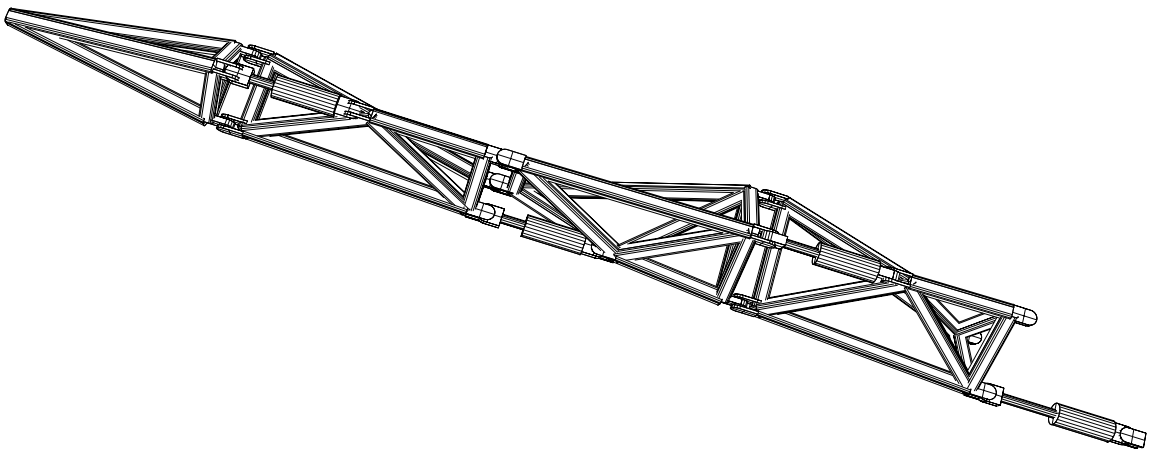

Figure 1.2. An example structure, shown with linear actuators.

much of the underlying structure is applicable to a much wider domain as will be discussed in Section 7.1.

- It was felt that the infrastructure necessary for successful implementation was available. Access to the low-level data and functionality of a solid modeler was crucial to the success of this project. The implementation difficulties inherent in the project would have been much greater without the availability of the Alpha_1[8] modeling system and access to system experts.

However, the problem domain was still too large for the given time and resource constraints. To reduce the problem scope to a more reasonable level, the class of large dynamic structures was further restricted as follows:

1. Structural synthesis and dynamic analysis would only be supported for open kinematic chains. Closed kinematic loops, such as four bar linkages would not be supported. This restriction was imposed because the synthesis of closed kinematic chains is still an area of active research and the dynamic analysis of closed kinematic chains is much more complex.
2. The structures would be limited to space frames constructed from standard beam cross sections. This restriction was imposed to simplify both the structural synthesis and geometric refinement algorithms, and reflects both standard practice as well as common economic constraints.
3. Actuation would be limited to direct drive types of actuators, principally linear fluid actuators, although limited support has been provided for the synthesis of structures using rotary actuators. This restriction was imposed to eliminate problems associated with the synthesis of drive trains, as well as to limit the size of the actuator catalog.
4. The linear actuators would be further restricted to a single mounting type, spherical rod ends at both ends of the actuator, again to limit the size of the actuator catalog. As this is the most common mounting system for this type of application, this is viewed as a minor restriction.

With these restrictions, the problem domain was reduced to a reasonable size without excessive loss of generality. Relaxation of these restrictions provides a

logical starting place for future work.

The DynaFrame (Dynamic Frame) system was constructed to address the above types of problems. An information flowchart for the DynaFrame system is shown in Figure 1.3 showing the principal software modules and related data communication paths.

With construction of high level CAD tools as the principal focus of the project, the foundation upon which the tools are implemented becomes important. During the last decade, the size and sophistication of CAD systems has grown past the point where a single researcher can start from nothing and construct a competitive system. To engage in this type of research, a higher level starting point is necessary. Although many possible CAD tools could be used as a foundation for this project, the Alpha_1 solid modeling system[8], developed by the Computer Science Department at the University of Utah was selected. The Alpha_1 system provided an object oriented, nonuniform rational B-Spline (NURBS) based solid modeler with

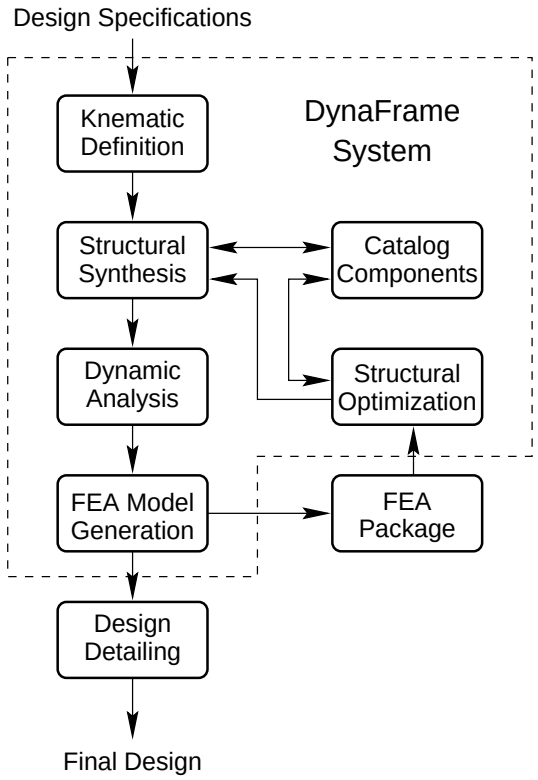

Figure 1.3. Modules and information flow in the DynaFrame system.

the extensibility, geometric tools, parametric modeling, visualization programs and access to low level routines necessary for the successful completion of this project. The project was designed to construct a number of layers of functionality on top of the Alpha_1 system to provide the necessary tools for integration of many of the engineering tasks into the CAD environment. Most of these layers have a wider range of application than the restricted problem set chosen for this project. As a rule, the higher the layer, the more powerful the tool and the narrower the application range [Figure 1.4].

Not all of the necessary tools were constructed in the Alpha_1 system. For the structural analysis requirements, an interface was written to provide two way communication between the Alpha_1 solid modeler and several different FEA packages, including Ansys, Patran and Cosmos/M. This interface was designed to provide relatively easy extensibility to other FEA packages, supporting the changing user needs and available applications.

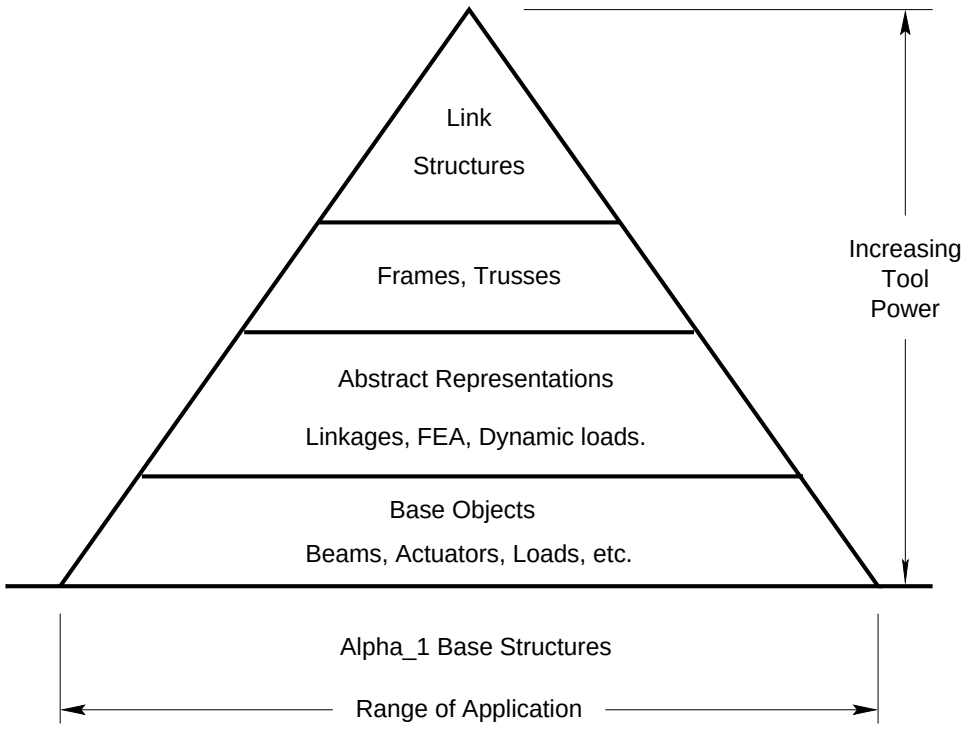

Figure 1.4. A representation of the functional layers and tool power versus the tool application range.

The specific goals for this project were as follows:

1. Design and implementation of a high level description of the desired mechanical system, from which the necessary geometric and analytical models could be algorithmically constructed.
2. Design and implementation of a simple abstract representation useful for conceptual design. This representation would be tightly integrated with the high level description, such that modifications to the fundamental design constraints could be easily accommodated.
3. Algorithmic construction of analytical models for both dynamic and structural analysis, including integration between the different analyses. This would include construction of complete analytical models as well as determination of reasonable boundary conditions.
4. Discrete optimization of the mechanical structures subject to both stress and buckling constraints. Limitations on the type and number of differing beam sizes specified within a single design would also be included in this process.

These goals were specifically selected to address the previously discussed shortcomings in current CAD systems. The high level description allows for radical design changes with a minimal effort, and the automatic construction of analytical models greatly reduces the analysis effort. The abstract representation allows a much greater freedom during the conceptual design phase. Not only are modifications to the abstract representation algorithmically propagated to the structural model, but the abstract representation also provides a very useful tool for initial visualization of the structure kinematics. The discrete optimization and structural geometry refinement allow for much faster design cycle times and greatly reduce the required engineering effort.

1.5 Background

This project is based upon a wide range of previous work, both in mechanical engineering and in computer science. As a complete discussion of all potentially relevant work would require several volumes, the material will be restricted to the main

thrusts of the project: intelligent CAD systems, integrated CAD environments, structural synthesis and structural optimization. The discussion of previous work relating to structural synthesis and structural optimization will occur in the relevant chapters; a brief survey of previous work relating to integrated environments and intelligent CAD systems follows.

1.5.1 Intelligent Design Systems

The need for inclusion of some degree of intelligence in CAD systems has long been apparent and a great deal of work has been performed in this area. Many individuals in the artificial intelligence(AI) community are currently working on applying AI techniques to the design problem. Although there are a number of different approaches to this topic, the major approaches can be divided into several different categories[21]. However, there is a great deal of overlap between them and many areas use approaches from other categories. A short discussion of each of these major categories follows:

Knowledge-based Design is the application of "expert system" technology to the design problem[38]. Expert systems attempt to capture knowledge in *heuristics* or "rules of thumb" and apply these to the problem using some form of *inferencing*. These systems are deterministic and have found application in a wide range of areas[131, 88, 1, 130, 27, 11, 107, 127].

Case-Based Design uses expertise developed in past designs and attempts to extrapolate this knowledge to new situations[89, 44]. Three fundamental issues with these systems are:

1. The representation of the previous design cases.
2. The methodology used to compare past and current design domains.
3. The adaption of past designs to the current design scenario.

Functional Design is a methodology that provides representations for various abstract functions that are of importance to the design and interrelationships between design elements[145]. Using this representation, a reasoning strategy

is applied that determines design function validity. Current implementations provide a purely abstract representation of the design that do not include any geometric representation.

Machine Learning systems use past experience to provide design solutions to current problems[39, 141]. These systems incorporate the designers decisions into the knowledge base such that performance is improved over time. There are a large number of different learning algorithms including agent-based learning, analogical reasoning, genetic algorithms, knowledge compilation, neural networks, and others.

Grammatical Systems are systems that attempt to develop a grammatical representation for the complete design space[22, 110]. Usually this includes a minimal vocabulary set describing the fundamental elements and a set of transformations or rules that define how the elements may be combined into more complex entities.

Analogical-Based Design involves the transfer of knowledge from an expert system or a previous design solution to the current design[49]. Typically this requires a sufficiently *generic abstraction* of the knowledge to allow some determination of relevance to the current situation. Although this approach is not limited to case-based design, analogical-based algorithms have been employed to extract relevant information from past designs.

Design Rationale Systems are systems that attempt to capture the design decisions and design intent for documentation and/or future use[80]. Currently there is a great deal of variability in the amount, type and structure of the design decisions as well as the degree of formality in the system representation. These systems are still in their infancy and a great deal of work needs to be performed before these systems are useful for real world problems.

Multi-agent Systems are systems that try to embed the design knowledge in a set of processes or agents, each of which has expertise in a specific domain[78, 34, 35].

These agents then communicate and work together to determine a solution to the design problem. Difficulties remain in interagent communication, design coherence, and algorithms to prevent thrashing, all of which are active areas of research.

Configuration-Based Design systems are designed to take a set of components and derive a configuration based upon some given criteria[154]. These systems may include a large amount of *domain-specific* knowledge or may be based upon more uniform methods. Most of the structural synthesis approaches can be considered variations of configuration-based design.

Examples of all of the above approaches have been implemented with varying degrees of success. Most systems use some combination of approaches to try to circumvent the difficulties inherent in the various approaches[63, 53]. Several systems have been implemented that reflect the systematic design approach suggested by Pahl et al.[97, 65, 127]. These systems are typically a combination of functional design and knowledge-based approaches. An additional system described by Ward et al.[150] is based upon functional design but includes a formal "compositional" language to describe the functional interactions.

Despite a smaller design space, the knowledge-based system described by Ramaswamy et al.[107] and used for searching the parametric space of possible automotive designs is of some interest. The system quantifies the performance of an automobile using 19 parameters and attempts to determine the optimum configuration of the design parameters. Although geometric information is not included, the discussion relating to problem size and discrete variables was of relevance to this project. Also, the integration between a knowledge-based system and a structural analysis package proposed by Sriram et al.[131] had a similar approach, and was taken into consideration during this project.

1.5.2 Intelligent CAD Systems

There has been a considerable amount of work in recent years relating to adding some degree of intelligence to CAD systems. Most of this work has been related to the organization of the complete CAD system and has been performed in a top down approach [2, 61]. Although intellectually interesting in general, this work is sufficiently advanced from existing CAD technology to provide limited potential for construction of a working system. Most published examples are limited to very simple problems with little or no geometric content. Given the nature of the design process, there appears to be fundamental difficulties with this approach as the additional knowledge to generate the design must come from somewhere.

Other systems have implemented functional design approaches to relatively simple problems, such as the design of electro-mechanical appliances[143], sizing electric motors and transmission components[150], and the design of compound cam mechanisms[157] but the connection to a geometric representation is tenuous at best. The same can be said for the office building configuration system proposed by Bédard et al.[11] and the heuristic object-based system proposed by Biedermann[16] for structural frame design. Along the same line the integration of a knowledge-based system with a structural optimization was performed earlier by Adeli et al.[1] again with quite limited geometry.

An additional system was suggested by Bond et al.[18] that integrated a knowledge-based system(KBS) written in Prolog to the CADAM CAD system. A similar system was constructed by Olhoff et al.[94] in which a structural optimization system (CAOS) was combined with a topology preprocessor on top of the Auto-CAD system. Although an interesting approach, it appears that these systems still suffer from communication restrictions between the various packages and none of these systems attempt to construct geometry of the complexity necessary for this project.

1.5.3 Integrated CAD Environments

The problem of data transfer between various representations has long been understood and a great deal of work has been performed in both academic and commercial areas trying to improve the situation[121]. An architecture for an

integrated analysis and optimization environment was proposed by Prasad[106] who discusses the advantages and disadvantages of a variety of approaches. Current industry standards such as IGES provide means to exchange simple geometric information, that then requires transformation into a useful form. As this often requires more effort than remodeling and must be repeated for each design iteration, the frustration of design engineers is understandable[54, 31, 137]. A common approach to this problem has been to perform direct conversion of the data bases between the different packages; however, the combinatorial nature of this solution precludes wide acceptance. Several current commercial CAD packages have taken a similar approach, including Pro/Engineer and IDEAS. Unfortunately, the exchange of data is often limited to the intersection of the differing data representations and tools for manipulation of nongeometric information are primitive, if available.

The difficulty in accessing and manipulating geometric information is also apparent in the the integrated system described by Reinschmidt et al.[112]. They integrated a commercial knowledge-base system with several commercial CAD systems. Although the results provided an interesting system, the relatively low level of geometric integration limited this approach to relatively simple geometric manipulation. The same is true of a similar system constructed by Bond et al.[18]. Another integrated system is described by Cutkosky et al.[34] in which a variety of different environments and modeling tools were integrated using a multiagent approach. Although promising, the problem domain was so highly restricted that extending this approach to this project domain was impractical.

The integrated system for design of artillery by Seah et al.[126], although quite restricted, suggests some of the difficulties with trying to incorporate a wide range of dynamic loads into the design system. As with most of these types of systems, the correlation to structural geometry is slim at best.

A recent summary paper by Erdman[41] discusses the state of the art in computer-aided mechanism design, with emphasis on the interaction of dynamic analysis and synthesis of kinematic mechanisms. Much of the work is devoted to the attempts to devise reasonable representations from which the mechanisms can be

synthesized and is not particularly relevant to this project. However, the discussions of future directions and the ideal system have many similarities with the structure and functionality of this project.

Although lacking many features expected of a CAD system, the structural frame design system proposed by Biedermann[16] integrates to the SODA structural analysis and optimization software by generating the desired input files directly from the structural model. The analysis results are then read back into the system and the design is heuristically modified and the process repeated until a satisfactory design is achieved. Different analysis packages are algorithmically selected depending upon the required type of analysis. Despite the inherent difficulties of direct data base conversion, this approach appears superior to attempting to integrate and maintain the entire system[136] into a single unified environment. Even though the unified approach has a certain appeal, the resources necessary to construct and maintain such systems rapidly expand past the point where any sort of coherence can be maintained.

Finally, this discussion would not be complete without including the Alpha_1 system that integrates a NURBS-based solid modeling package to a computer-aided manufacturing system[43]. They take an object-oriented, feature-based approach that integrates feature-specific knowledge about the individual objects to generate high level process plans for manufacture as well as generating the instructions for the CNC machine tools. This embedding of the knowledge into the feature manipulation tools provides access to the geometric tools and information. It was felt that using this type of access, information, tools, and system structure as a foundation would provide the greatest possibility for project success.

1.5.4 Background Summary

To date, no previous work has been located that describes a working system with the capabilities of the system constructed for this project. Examination of the AI design approaches and systems and their potential application to this project suggests several potential avenues of approach. The knowledge-based systems are well developed and have a proven record of success with real world problems. The

case-based approaches appear limited and highly dependent upon case representation, as well as very rudimentary for realistic problems. There is an attraction to the abstract representations inherent in the functional design systems. It is not unusual to see designers creating and discussing abstract representations in the conceptual design phase and the inclusion of some form of abstract representation in the project could provide a great deal of assistance to designers. It appears that the machine learning, grammatical, analogical, rational and multi agent systems will require a significant amount of development effort before becoming useful for real-world problems. The configuration-based design systems have a great deal of commonality with this project and together with a knowledge-based system would seem to provide an approach with a reasonable probability of success. As such, this combination of these approaches was selected for implementation.

Most of the systems proposed to provide some degree of intelligence to a CAD system either ignore the geometric problems or try to integrate some form of AI system with an existing CAD package. Both approaches seem to have significant limitations. Ignoring the relationship of geometry to the design problems greatly reduces the utility of these packages and may well render them little more than academic curiosities. Integrating existing AI and CAD systems results in the same data translation and representation problems inherent in the integrated system problem. The separation of knowledge and geometric manipulation tools results in integrating difficulties and again limits the usefulness of these systems. For these reasons the approach chosen for this project was to construct the heuristic reasoning on top of an existing extensible CAD package, thus eliminating the integration difficulties.

1.6 Approach Summary

With an understanding of the problem and general background in mind, the general approach selected for this project is summarized in the following:

- Design and implement a simple abstract representation for open kinematic chains. This representation should have a clear 3-D geometric representation

and provide sufficient feedback to assist the design engineer with the conceptual design of the linkage. This representation along with other data structures are discussed in Chapter 2.

- Integrate the abstract linkage representation into a heuristic-based structural synthesizer. With minimal designer input, the system should be able to proceed from the abstract linkage representation to a structure model. After examination of the various approaches to intelligent CAD systems, a heuristic approach was selected as providing the necessary determinism, fine level of control and scalability. Details of the structural synthesis heuristics and implementation are the subject of Chapter 3.
- Algorithmically construct a abstract model suitable for dynamic analysis. With minimal designer input and using the linkage and structural models, the system should automatically perform the necessary dynamic analysis to allow for reasonable understanding and computation of the dynamic loads acting on the structure. This is discussed in Chapter 4.
- Using the synthesized structure and the dynamic loads, algorithmically construct an FEA model for each individual structure in the linkage. After examination of the various approaches to CAD system integration with analysis packages, a direct data base conversion approach similar to Biedermann's[16] was selected. This allows for control of the exact level of data exchange as well as being extensible as interfaces to new software packages are required. Details of the FEA model generation and interface with external FEA packages is the subject of Chapter 5.
- Using the results of the FEA, perform a discrete optimization on the structures. Once the structural design constraints are satisfied the structural synthesis, dynamic analysis, and structural analysis can be repeated until a satisfactory design is achieved. Details of the discrete optimization algorithm and implementation are the subject of Chapter 6.

CHAPTER 2

REPRESENTATION

From a computer science perspective, the data representation is a fundamental problem, affecting algorithm design, system performance, reliability and maintainability. The current software paradigm is to represent associated data as *objects*, in what is known as *object-oriented* programming[19]. That approach was used for this project, building upon the existing objects and methods present in the Alpha_1 solid modeling system. This chapter contains semi-formal descriptions of many of the object representations constructed during this project as well as descriptions of the common algorithms(methods) used by these objects.

2.1 Linkages

The representation and manipulation of open kinematic chains or *linkages* was fundamental to this project. These objects provide the foundation for the conceptual design, structural synthesis and dynamic analysis. As a primary purpose of this project was to create a set of tools useful to design engineers, their input was gathered and consolidated. In discussion with and observation of several design engineers, several points become apparent. These points are listed below, in no particular order:

- For this kind of problem, designers initially think in somewhat abstract terms, worrying about axis of rotation, ranges of motion and other kinds of what are termed *kinematic* parameters. Much of the initial time and effort is spent on defining these kinematic parameters; therefore the representation should support these efforts as much as possible.

- Early and/or rapid estimates of base loads, required flow rates and actuator sizes may be required for concurrent engineering. Therefore, the representation should allow calculation of these parameters as soon as possible.
- Much of the design effort in establishing the desired kinematics involves defining the travel limits or *workspace* of the linkage. Assistance with this process, in particular multiple views of the travel limits would be very useful.

To address these needs, an abstract geometric representation referred to as a *link* was implemented. A link is defined as a rigid body possessing a single rotational degree-of-freedom(DOF), defined to be about the Z axis of the local coordinate system. A link may have any geometric representation but is usually modeled as a circular cross section swept along the *link axis,* that can be any NURBS curve in any orientation. The rotation axis is usually modeled as a slightly larger cylinder whose center axis is coincident to the rotational axis. Links may be connected to form *linkages,* an example of that is shown in Figure 2.1. Linkages are a recursive data structure with the local coordinate system of each link defining its relationship in terms of the previous links coordinate system and current rotational angle. A

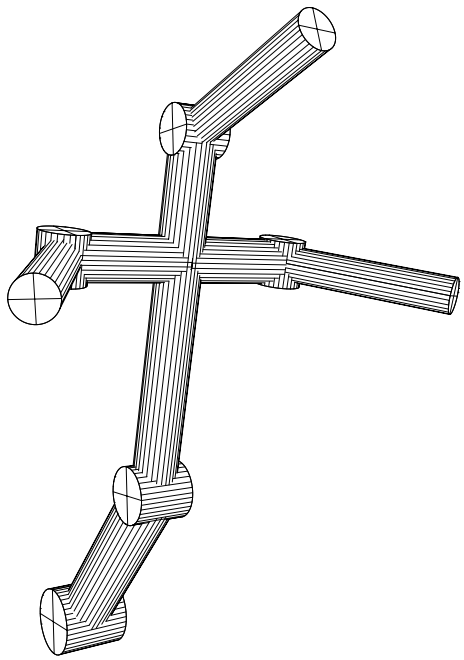

Figure 2.1. A simple linkage composed of five links.

link may have any number of external links connected to it but currently only the first link in the linkage may be fixed to an external reference frame. This constraint restricts linkages to be a type of kinematic structures known as "open kinematic chains" and was included to simplify both the geometric manipulation and dynamic analysis. Linkages are easily and rapidly constructed and manipulated, and serve as an abstract representation of the physical structure. Due to this ease of construction and manipulation, linkages are ideal for use in conceptual design, when the joint location and range-of-motion(ROM) of the structure are being determined.

The representation for the link object $\mathbf{Lk}_i$ used in this project was defined as follows:

$$\mathbf{Lk}_i = \begin{cases} \mathbf{C}_i & \text{the local coordinate frame} \\ Ac_i & \text{the current rotation angle of the link} \\ Ax_i & \text{the maximum rotation angle of the link} \\ An_i & \text{the minimum rotation angle of the link} \\ Ca_i & \text{a curve representing the center of the link} \\ Cs_i & \text{the link cross section geometry} \\ El_i & \text{the length of the end section} \\ Es_i & \text{the cross section geometry of the end section} \\ \omega_i & \text{the local rotational velocity of the link} \\ \alpha_i & \text{the local rotational acceleration of the link} \\ \mathbf{In}_i & \text{the inertia tensor of the link} \\ \mathbf{Ld}_i & \text{the current base loads of the link} \\ \mathbf{Lw}_i & \text{the worst base loads seen by the link} \\ \mathbf{Ls}_i & \text{the external loads acting upon the link} \\ \mathbf{Lk}_{i+1} & \text{any external links that connect to } \mathbf{Lk}_i \end{cases}$$

Linkages are the foundation upon that the physical structure will be constructed and, as such, the inertial properties are usually computed for a synthesized structure. However, there may be occasions when dynamic load information is needed before the synthesis process has been performed. In these cases, an approximation of the inertial properties, usually a point mass, may be constructed for each link and the linkage used directly for dynamic analysis. This analysis may include any external mass and/or loads related to any links in the linkage. Discussion of the representation and manipulation of these nongeometric abstractions, such as point loads, inertia properties, catalog components, and clevises follows.

2.2 Loads

A representation of the load condition on a point is a common and useful engineering approximation. Such a representation is also necessary for both dynamic and structural analytical purposes. As the ability to manipulate load information would be useful for applications outside the scope of this project, the construction of a wide range of general purpose tools seemed appropriate.

The initial point load representation created was broken into two distinct category types, one describing force loads and another describing moment loads. However, after some experimentation, it became apparent that eliminating the distinction between the two objects types would reduce both the number of different methods and storage requirements, at the cost of a slight increase in object size and method complexity.

The current object created to represent a point load contains three items: a local coordinate system, a 3-D force vector, and a 3-D moment vector, with both the force vector and moment vector being referenced in the local coordinate system. A formal definition of the point load $\mathbf{Ls}_i$ would be:

$$\mathbf{Ls}_i = \begin{cases} \mathbf{C}_i & \text{the local coordinate frame} \\ \mathbf{f}_i & \text{a vector representing the force on the point} \\ \mathbf{n}_i & \text{a vector representing the moment on the point} \end{cases}$$

Both the force and moment vector are referenced to the local coordinate system with the components of the moment vector corresponding to the moment magnitude about the related axis.

This object class was extended to include kinematic information for use in the dynamic analysis. A subclass of the point load was created that in addition to the load data also included velocity and acceleration information. Four additional 3-D vectors were added to represent the velocity, angular velocity, acceleration and angular acceleration of the point. The formal definition of the dynamic load object $\mathbf{Ld}_i$ would be:

$$\mathbf{Ld}_i = \begin{cases} \mathbf{C}_i & \text{the local coordinate frame} \\ \mathbf{f}_i & \text{a vector representing the force on the point} \\ \mathbf{n}_i & \text{a vector representing the moment on the point} \\ \mathbf{v}_i & \text{a vector representing the velocity of the point} \\ \mathbf{a}_i & \text{a vector representing the acceleration of the point} \\ \omega_i & \text{the angular velocity of the point} \\ \alpha_i & \text{the angular acceleration of the point} \end{cases}$$

The dynamic load object was not included in the original implementation, but the system was revised for several reasons. Foremost, if structural geometry and inertial properties were defined and constant, the kinematics and loads were dependent quantities. If the kinematics were known, the dynamic loads were calculated from them, or if the dynamic loads and the initial kinematic condition were known, then the resulting kinematic information might be found. The inclusion of the kinematic information with the load data enforced this coupling, insuring synchronization of the data. A couple of additional reasons also prompted this data structure. First, it was desirable when searching for the worst case loads to store both the worst loads as well as the kinematic parameters for those loads. Second, in decomposing the loads for structural analysis, it would be advantageous to use the tools already constructed for manipulation of point loads. Both these representations proved very useful and have been extensively used in the dynamic analysis and FEA model creation.

2.2.1 Manipulation of Loads

In addition to defining load objects, the manipulation of load information is very important to design engineers, as computation of the various 3-D load states necessary for analysis can require considerable effort. To reduce this computational effort, several methods were implemented for addition, decomposition and other manipulation of point loads. The addition of two point loads $\mathbf{L}_i, \mathbf{L}_j$ is accomplished as follows:

$$\mathbf{f}_k = {}^i\mathbf{R}_j\mathbf{f}_i + \mathbf{f}_j \tag{2.1}$$

$$\mathbf{n}_k = \mathbf{t}_{ji} \times {}^i\mathbf{R}_j\mathbf{f}_i + {}^i\mathbf{R}_j\mathbf{n}_i + \mathbf{n}_j \tag{2.2}$$

where ${}^{i}\mathbf{R}_{j}$ is defined as the rotational transformation matrix relating $\mathbf{C}_j$ to $\mathbf{C}_i$ and $\mathbf{t}_{ji}$ is the vector from the origin of $\mathbf{C}_j$ to $\mathbf{C}_i$ referenced in the $\mathbf{C}_j$ coordinate system. By default, the reference coordinate frame for the resulting sum is defined as being $\mathbf{C}_j$. For translation of the load $\mathbf{Ls}_i$ from $\mathbf{C}_i$ to $\mathbf{C}_j$, the force and moment vectors $\mathbf{f}_j$ and $\mathbf{n}_j$ are defined as zero length and the same algorithm applied. The addition of dynamic loads is similar to the above with the restriction that translation is not allowed. This was done because there is insufficient information in the dynamic load object to calculate the new kinematic parameters through a translation. Computation of the kinematic parameters through a rotation of coordinate frames is supported as follows:

$$\mathbf{v}_j = {}^{i}\mathbf{R}_j\mathbf{v}_i \tag{2.3}$$

$$\mathbf{a}_j = {}^{i}\mathbf{R}_j\mathbf{a}_i \tag{2.4}$$

$$\omega_j = {}^{i}\mathbf{R}_j\omega_i \tag{2.5}$$

$$\alpha_j = {}^{i}\mathbf{R}_j\alpha_i \tag{2.6}$$

If calculation of the loads at a different point is required, the kinematic information for the new point is no longer known and the dynamic load replaced with a point load. This prevents possible incorporation of incorrect kinematic information into the analysis process.

Specific types of point load decomposition proved useful and were implemented. In particular, the decomposition of a point load to three points representing two hinge points and an actuator attachment point was necessary for generation of the FEA model. As this configuration is quite common in the kind of problems addressed by this project, further discussion of the problem is appropriate.

The problem may be described in detail using the following variables, shown in Figure 2.2.

$P_{h1,h2}$ = the first and second hinge points

P_a = the actuator point

P_L = the load point

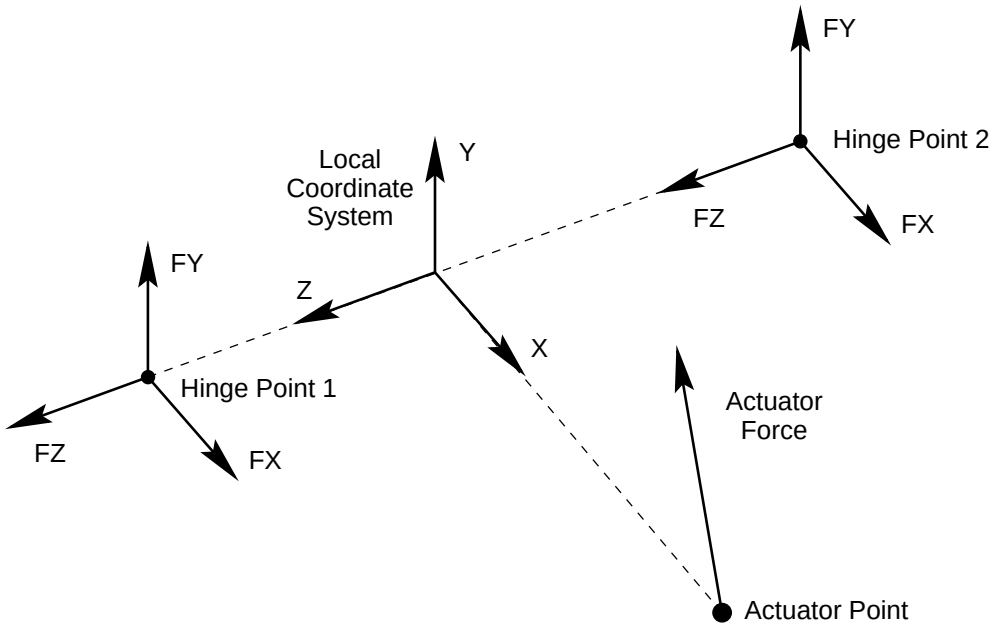

Figure 2.2. Location and orientation of the individual forces when a load is decomposed onto three points.

$\mathbf{u}_a$ = a unit vector in the actuator line of action

$\mathbf{M}_L$ = the moment vector at point P_L

$\mathbf{F}_L$ = the force vector at point P_L

$\mathbf{F}_{(h1,h2,a)}$ = the force vector acting on points P_{h1}, P_{h2} and P_a respectively

$\mathbf{r}_{h1\to a}$ = vector from from P_{h1} to P_a

$\mathbf{r}_{L\to a}$ = vector from P_L to P_a

$\mathbf{r}_{h2\to h1}$ = vector from P_{h2} to P_{h1}

First, the load is calculated at the intersection point of the hinge axis and a plane normal to the hinge axis containing the actuator point. This intersection point is located as follows:

$$\mathbf{u}_{a1} = \text{unit vector in the direction of } \mathbf{r}_{h1\to a}$$

$$\mathbf{u}_{h2} = \text{unit vector in the direction from } P_{h1} \text{ to } P_{h2}$$

$$\theta = \mathbf{u}_{a1} \cdot \mathbf{u}_{h2} \tag{2.7}$$

$$P_L = P_{h1} + \mathbf{r}_{a1} cos(\theta) \tag{2.8}$$

The local coordinate frame is defined as:

$$\begin{aligned} \mathbf{u}_x &= \text{unit vector in the direction from } P_L \text{ to } P_a \\ \mathbf{u}_z &= \text{unit vector in the direction from } P_L \text{ to } P_{h2} \\ \mathbf{u}_y &= \mathbf{u}_z \times \mathbf{u}_x \end{aligned} \tag{2.9}$$

Once the given loads are transformed into the local coordinate frame, decomposition proceeds starting with the actuator force.

$$\mathbf{r}_\alpha = \mathbf{u}_z \times (\mathbf{u}_z \times \mathbf{u}_a) \tag{2.10}$$

$$\mathbf{u}_\alpha = \frac{1}{\mid \mathbf{r}_\alpha \mid} \mathbf{r}_\alpha \tag{2.11}$$

$$r = \mid \mathbf{r}_{L \to a} \mid sin(cos^{-1}(\mathbf{u}_\alpha \cdot \mathbf{u}_x)) \tag{2.12}$$

$$\mathbf{r} = r(\mathbf{u}_\alpha \times \mathbf{u}_z) \tag{2.13}$$

$$\mathbf{F}_a = \frac{M_{Lz}}{r \mid \mathbf{r}_\alpha \mid} \mathbf{u}_a \tag{2.14}$$

The reaction force vector must be added into $\mathbf{F}_L$ to maintain equality. This is easily accomplished, as the rotational orientations of all the local coordinate frames are identical.

$$\mathbf{F}_T = \mathbf{F}_L - \mathbf{F}_a \tag{2.15}$$

This calculation of the actuator force is based upon the assumption that the linear actuator only supports axial forces. This is generally the case, as spherical bearings in the actuator ends was an initial design restriction and significant side loading of linear actuators may greatly decrease their reliability and life expectancy. After the determination of the actuator force has been completed, computation of the hinge point forces may proceed. First, the reduction of the remaining moments into force couples:

$$Fmx_{h1} = \frac{-M_{Lx}}{\mid \mathbf{r}_{h2 \to h1} \mid} \tag{2.16}$$

$$Fmx_{h2} = \frac{M_{Lx}}{\mid \mathbf{r}_{h2 \to h1} \mid} \tag{2.17}$$

$$Fmy_{h1} = \frac{M_{Ly}}{\mid \mathbf{r}_{h2\rightarrow h1} \mid} \tag{2.18}$$

$$Fmy_{h2} = \frac{-M_{Ly}}{\mid \mathbf{r}_{h2\rightarrow h1} \mid} \tag{2.19}$$

Next, the distribution of the in-plane forces to the hinge points, determined by the need not to add any additional moment terms.

$$\mathbf{r}_{L\rightarrow h1} = \text{vector from } P_L \text{ to } P_{h1} \tag{2.20}$$

$$\mathbf{r}_{L\rightarrow h2} = \text{vector from } P_L \text{ to } P_{h2} \tag{2.21}$$

$$t_{h2} = \frac{\mid \mathbf{r}_{L\rightarrow h1} \mid}{\mid \mathbf{r}_{h2\rightarrow h1} \mid} \tag{2.22}$$

$$t_{h1} = \frac{\mid \mathbf{r}_{L\rightarrow h2} \mid}{\mid \mathbf{r}_{h2\rightarrow h1} \mid} \tag{2.23}$$

$$Ffx_{h1} = F_{Tx}t_{h1} \tag{2.24}$$

$$Ffx_{h2} = F_{Tx}t_{h2} \tag{2.25}$$

$$Ffy_{h1} = F_{Ty}t_{h1} \tag{2.26}$$

$$Ffy_{h2} = F_{Ty}t_{h2} \tag{2.27}$$

The above is correct as long as P_L lies between P_{h1} and P_{h2} but a correction of sign is needed if this is not the case.

$$Ffx_{h1} = \begin{cases} Ffx_{h1}, & t_{h2} \leq 1.0 \\ -Ffx_{h1}, & t_{h2} > 1.0 \end{cases} \tag{2.28}$$

$$Ffy_{h1} = \begin{cases} Ffx_{h1}, & t_{h2} \leq 1.0 \\ -Ffx_{h1}, & t_{h2} > 1.0 \end{cases} \tag{2.29}$$

$$Ffx_{h2} = \begin{cases} Ffx_{h2}, & t_{h1} \leq 1.0 \\ -Ffx_{h2}, & t_{h1} > 1.0 \end{cases} \tag{2.30}$$

$$Ffy_{h2} = \begin{cases} Ffx_{h2}, & t_{h1} \leq 1.0 \\ -Ffx_{h2}, & t_{h1} > 1.0 \end{cases} \tag{2.31}$$

The last component that must be distributed is the axial force. How this is to be distributed is determined by the specific implementation of hinge bearings used and how and where this load is transferred. If one of the hinge points always supports the axial load the solution is trivial:

$$\left.\begin{array}{rcl} Ffz_{h1} &=& F_{Tz} \\ Ffz_{h2} &=& 0 \end{array}\right\} \text{ if } P_{h1} \text{ supports all the axial load} \tag{2.32}$$

$$\left.\begin{array}{rcl} Ffz_{h1} &=& 0 \\ Ffz_{h2} &=& F_T \end{array}\right\} \text{ if } P_{h2} \text{ supports all the axial load} \tag{2.33}$$

Another trivial solution, although very difficult to achieve in practice, is equal distribution of the axial load between the two points.

$$\left.\begin{array}{rcl} Ffz_{h1} &=& F_{Tz}/2 \\ Ffz_{h2} &=& F_{Tz}/2 \end{array}\right\} \text{ if } P_{h1} \text{ and } P_{h2} \text{ support the axial load} \tag{2.34}$$

The cases often found in practice are that all of the axial load is supported by a single hinge point, but the point that supports the load does change with the direction of the axial load. This can be thought of as the load being carried on the *inside* or *outside* of the hinge. Mathematically, these are described as follows for the inside case:

$$\begin{array}{rcll} Ffz_{h1} &=& F_{Tz} & \left.\begin{array}{l} \\ \end{array}\right\} \quad F_{Tz} \geq 0 \\ Ffz_{h2} &=& 0 & \\ Ffz_{h1} &=& 0 & \left.\begin{array}{l} \\ \end{array}\right\} \quad F_{Tz} > 0 \\ Ffz_{h2} &=& -F_{Tz} & \end{array} \tag{2.35}$$

and for the outside case:

$$\begin{array}{rcll} Ffz_{h1} &=& 0 & \left.\begin{array}{l} \\ \end{array}\right\} \quad F_{Tz} \geq 0 \\ Ffz_{h2} &=& F_{Tz} & \\ Ffz_{h1} &=& -F_{Tz} & \left.\begin{array}{l} \\ \end{array}\right\} \quad F_{Tz} > 0 \\ Ffz_{h2} &=& 0 & \end{array} \tag{2.36}$$

Now that all the components have been found, the hinge force vectors can be assembled as follows:

$$\mathbf{F}_{h1} = (Ffx_{h1} + Fmx_{h1},\ Ffy_{h1} + Fmy_{h1},\ Ffz_{h1}) \tag{2.37}$$

$$\mathbf{F}_{h2} = (Ffx_{h2} + Fmx_{h2},\ Ffy_{h2} + Fmy_{h2},\ Ffz_{h2}) \tag{2.38}$$

For some applications, or for analysis purposes it may be desirable to have a single hinge point and allow the hinge point to carry the nonaxial moments. Rather than

perform a separate derivation and implementation for this case, an alternative solution was implemented using the two hinge point algorithm and the load calculation tools.

The approach involves finding an additional dummy hinge point, calculating the forces on the three points and then combining the two hinge loads. As the hinge axis is not defined in this case, an axis vector $\mathbf{u}_{axis}$ must be provided. With the axis vector, the dummy point P_d is defined as:

$$P_d = P_{h1} + \mathbf{u}_{axis} \tag{2.39}$$

After the load calculation is performed using the dummy hinge point, the hinge loads are combined using equations 2.1 and 2.2.

2.3 Inertia Properties

This project required the representation and determination of inertial properties, both of closed planar curves and solid objects. As neither the representations or the ability to perform these calculations was implemented in the Alpha_1 modeling system, providing this functionality became part of this project. As some unique approaches to both representation and calculation of these properties were implemented, a short discussion of the details is included.

2.3.1 Properties of Planar Curves

It is not immediately apparent why the representation and calculation of the properties of planar curves is necessary. After all, these are an abstract representation, not required for determination of the inertia of a solid body. However, the effort was undertaken for several reasons, some of immediate value and others potentially valuable over a longer term.

First, it was assumed that many of the structures constructed during this project could be represented as *beams*, which are often described as objects with constant cross sectional properties. Knowledge of the properties of the beam cross section is required before a beam type FEA model can be constructed. For most of the commercially available cross sections, this information is tabulated and available, so

this alone is insufficient reason to undertake the implementation of computational algorithms. However, it is not uncommon for designers to combine common cross sections, or to design custom cross sections. These are generally not tabulated and would require manual computation without computational support. An additional reason for implementation was to reduce the size and complexity of the information stored in the beam catalog. As the cross sectional geometry was required for proper visual display, and planar properties are determined by cross section geometry, inclusion of this information in the catalog is needlessly redundant.

A longer term reason for implementing the computation of planar properties was that this information is commonly used by engineers for analytical purposes. Despite its usefulness, it can be quite difficult and time consuming to manually calculate these properties. In many cases, either approximations are made or the design is modified to ease the calculation of this information. Neither approach is desirable: in the first, the uncertainty is increased, in the second, the design is needlessly constrained. For these reasons, derivation and implementation of computational algorithms for the properties of planar curves were undertaken.

The first step was to define what information was needed as well as a representation for that information. After examination of the beam analysis data requirements for some common FEA systems as well as some considerations for potential future applications requiring this type of information, the following data structure describing the planar properties of a cross section was implemented. The formal definition of the planar properties object $\mathbf{Pp}_i$ used for the project were:

$$\mathbf{Pp}_i = \begin{cases} \mathbf{C}_i & \text{the reference coordinate frame} \\ c_i & \text{a point located at the centroid} \\ A_i & \text{the area contained inside the cross section} \\ \mathbf{m}_i & \text{a vector representing the moment of inertia} \\ \mathbf{s}_i & \text{a vector representing the section modulus} \\ \mathbf{g}_i & \text{a vector representing the radius of gyration} \\ Ip_i & \text{a number describing the polar moment of inertia} \\ Tc_i & \text{a number describing the torsional constant} \\ sc_i & \text{a point located at the shear center} \\ wc_i & \text{a number describing the warp constant} \\ Mp_i & \text{a bounding box describing the limits of the cross section} \end{cases}$$

All of the information in the planar properties object is referenced to the local coordinate frame $\mathbf{C}_i$ that is restricted to be on the plane of the cross section and to have its $\mathbf{x}$ and $\mathbf{y}$ axis in the plane.

In general form, the centroid of an area $(\mathbf{x}_c, \mathbf{y}_c)$ lying in the $\mathbf{xy}$ plane can be found by:

$$x_c = \frac{\int_A x \, \mathrm{d}A}{\int_A \mathrm{d}A} \tag{2.40}$$

$$y_c = \frac{\int_A y \, \mathrm{d}A}{\int_A \mathrm{d}A} \tag{2.41}$$

and the moment of inertia about the $\mathbf{x}$ and $\mathbf{y}$ axis as well as the product of inertia I_{xy} are defined as:

$$I_x = \int_A y^2 \, \mathrm{d}A \tag{2.42}$$

$$I_y = \int_A x^2 \, \mathrm{d}A \tag{2.43}$$

$$I_{xy} = \int_A xy \, \mathrm{d}A \tag{2.44}$$

The section modulus can be defined as:

$$S_x = \frac{I_x}{c_y} \tag{2.45}$$

$$S_y = \frac{I_y}{c_x} \tag{2.46}$$

where c_i is the maximum distance from the axis to a point included in the cross section in the i direction. The radius of gyration also can be defined as:

$$R_x = \sqrt{\frac{I_x}{A}} \tag{2.47}$$

$$R_y = \sqrt{\frac{I_y}{A}} \tag{2.48}$$

The difficulty in computation of these quantities is in performing the integration, that is calculating the $\int_A f(x,y)\ \mathrm{d}A$ terms. A traditional approach is to use a polar representation rather than Cartesian for the area integration. Thus

$$\int_A f(x,y)\ \mathrm{d}A \ = \ \int\int f(r,\theta)\ \mathrm{d}r\ \mathrm{d}\theta \tag{2.49}$$

In the case of Alpha_1, as well as other boundary representation modeling systems, the cross sectional information provided is a NURBS curve or list of NURBS curves. Several approaches can be taken to perform this integration numerically, a number of which were explored and will be discussed.

The first method is to evaluate the NURBS at many locations and perform what is essentially a trapezoidal integration from these points similar to the approach proposed by [158]. The curve is evaluated at points of constant distance in parameter space, which rarely translates into constant distance in Euclidean space. Because of this unequal spacing of the evaluation points it is not possible to use higher level numerical integration schemes such as Simpson's rules. Although the results of this process are often adequate for many purposes, it suffers from two different problems, a systematic accuracy problem and a computational efficiency problem. The systematic accuracy problem results from the fact that the trapezoidal integration under estimates the effect of convex curves and overestimates the effect of concave curves. This results in a systematic error in the cross sectional properties, but as engineering cross sections may not have either convex or concave curves the size and direction of the error is very difficult to predict. On the positive side, as all of the cross sections being integrated are closed, and assuming an equal distribution of convex and concave areas, this process will tend to underestimate the area and other area dependent properties of the cross section [Figure 2.3]. This is attractive to design engineers, who desire to err on the side of being conservative. The computational problem results from the fact that evaluation of a NURBS at a fixed parametric location is an iterative process and as such is computationally

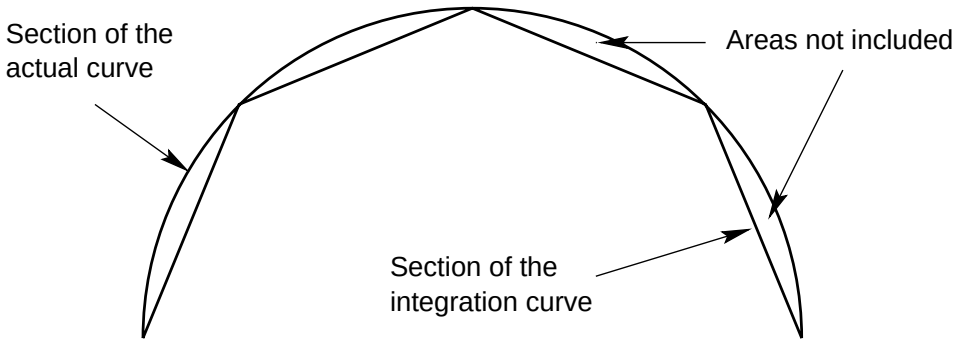

Figure 2.3. Example of areas not included in the integration process when the curve evaluation algorithm is used.

expensive. As many evaluations are desired to reduce the systematic error, the cost of this iterative process rapidly becomes significant.

The second approach is to refine the control polygon of the NURBS until it is close enough to approximate the curve. Once the control polygon is refined, it may be integrated directly, thus eliminating any need for curve evaluation. This is the approach used to display NURBS, and fast and efficient algorithms have been devised and implemented to perform this task. Unfortunately, this procedure also has a systematic error opposite that of trapezoidal integration in that it overestimates the effects of convex sections and underestimates the effects of concave sections [Figure 2.4]. Using the above assumption of equal distributions of convex and concave areas this process tends to overestimate the area properties, a behavior that bothers design engineers. An additional difficulty with this approach

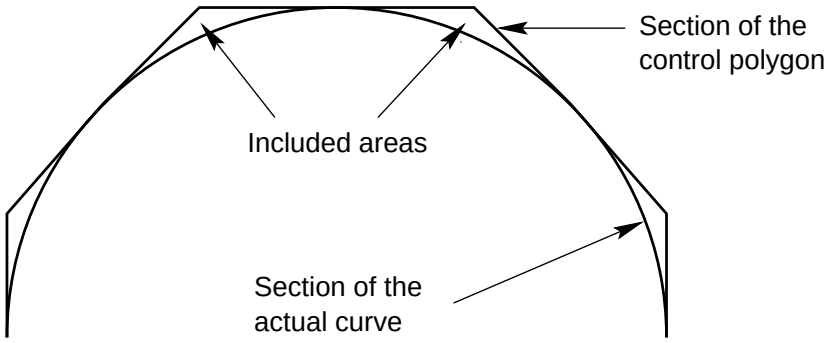

Figure 2.4. Example of areas included in the integration process but outside the boundary curve when the curve refinement algorithm is used.

is that the refinement takes place in parametric space and, as such, it is difficult to determine how accurately the control polygon approximates the curve. The obvious solution to this problem is to try to use a higher order numerical integration algorithm such as Simpson's Rules[25]. It is a relatively trivial matter to check the distance between the evaluation points and whenever enough are equally spaced, use Simpson's Rules integration in that segment. However, this was tried and little to no improvement was found, due to the fact that it is a rare occurrence when sufficient evaluation points are equally spaced to allow higher order integration. Therefore, investigation into a new approach was undertaken.

The approach used was to convert the NURBS into a higher order approximation that is easily integrated, rather than to try to create an accurate linear approximation of the NURBS curves. The approximation used was an arc-line approach, which had been implemented and extensively used for the generation of NC code from NURBS[8]. As very few NC machine tools can utilize higher order curves, algorithms to reduce NURBS to arcs and lines that the NC controller are capable of interpreting has been necessary. Most of these arc-line approximation algorithms are limited to planar curves, but the cross sections have the same restriction. An additional benefit is that the accuracy of the arc-line approximation is easily controllable within meaningful and measurable tolerances. Also, most cross sections created by engineers for structural purposes are composed of arcs and lines and therefore can be represented exactly by an arc-line approximation.

For numerical integration, the integration process is replaced by summation, thus:

$$\int\int f(r,\theta)\ \mathrm{d}r\ \mathrm{d}\theta \quad = \quad \sum_{i=1}^{n}\sum_{j=1}^{m} f(r,\theta)\Delta r\Delta\theta \tag{2.50}$$

With the arc-line approximation, the integration process is reduced to summation of the desired properties of two types of shapes, triangles and circular segments[Figure 2.5]. The properties of these geometric shapes can be easily determined and summed to provide a total for the cross section. The choice of a center point for the construction of the triangles is arbitrary but as a significant

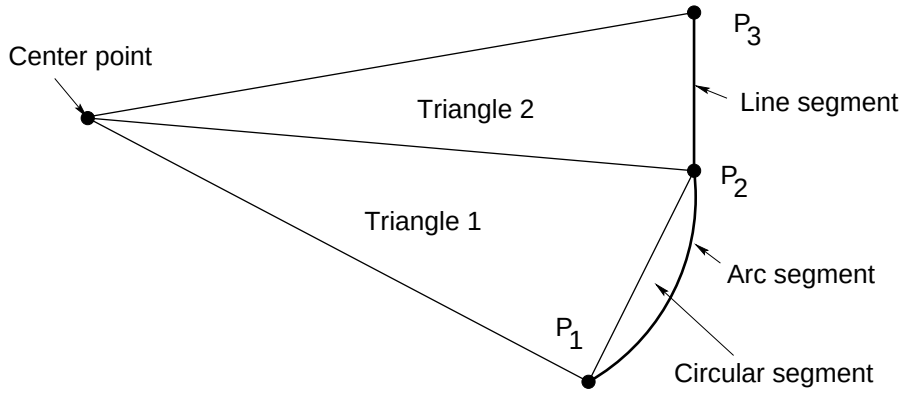

Figure 2.5. Example of triangles and circular segments created from an arc and line segments

amount of trigonometric calculations are required, selection of a center point close to the centroid will reduce the numerical error inherent in these computations. An approximation of the centroid useful for the initial center point c_{ic} can be determined by averaging the control polygon points p_j as follows:

$$c_{ic}(x) = 1/n \sum_{j=1}^{n-1} p_j(x) \tag{2.51}$$

$$c_{ic}(y) = 1/n \sum_{j=1}^{n-1} p_j(y) \tag{2.52}$$

where n is equal to the number of points on the control polygon. Once an initial center point has been selected, the integration process can commence. The centroid ($\mathbf{c}_t$) and area (A_t) of an arbitrary triangle with one vertex located at the origin was determined as follows[Figure 2.6]:

$$\mathbf{c}_t = 1/3(\mathbf{r}_i + \mathbf{r}_{i+1}) \tag{2.53}$$

$$A_t = \mid (\mathbf{r}_i + \mathbf{r}_{i+1}) \mid^2 sin(\theta/2) \tag{2.54}$$

This would work without problems for convex cross sections but the solution for general cross sections requires that the area be a signed quantity. The sign of the area is determined by:

$$t = \mathbf{u}_z \cdot (\mathbf{r}_i \times \mathbf{r}_{i+1}) \tag{2.55}$$

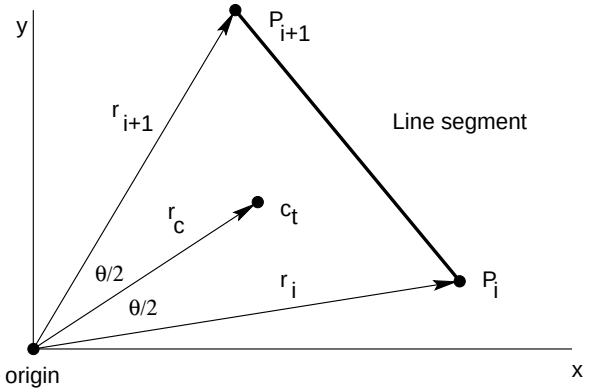

Figure 2.6. Graphic description of the variables used in determination of the centroid, area and inertial properties of an arbitrary triangle.

$$A_t = \begin{cases} A_t & t \geq 0 \\ -A_t & t < 0 \end{cases} \tag{2.56}$$

The centroid (c_{cs}) and area (A_{cs}) of an arbitrary circular segment[Figure 2.7] are determined by first computing the properties of the triangle created by the origin, start and endpoints of the arc. Then the area and centroid of the remaining circular segment are determined as follows[48]:

$$\mathbf{r}_{ct \to P1} = \text{a vector from the center of the arc to the first point} \tag{2.57}$$

$$\mathbf{r}_{ct \to P2} = \text{a vector from the center of the arc to the second point} \tag{2.58}$$

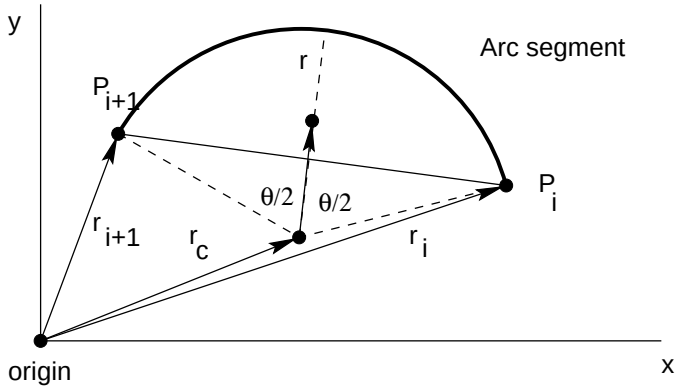

Figure 2.7. Graphic description of the variables used in determination of the centroid, area and inertial properties of an arbitrary circular segment.

$$\mathbf{u}_{mid} = \frac{\mathbf{r}_{ct\to P1} + \mathbf{r}_{ct\to P2}}{\mid (\mathbf{r}_{ct\to P1} + \mathbf{r}_{ct\to P2}) \mid} \tag{2.59}$$

$$\mathbf{c}_{cs} = \mathbf{r}_c + \frac{2r\ sin(\theta)\mathbf{u}_{mid}}{3\theta} \tag{2.60}$$

$$A_{cs} = r^2(\theta - sin(\theta)cos(\theta)) \tag{2.61}$$

Again, as in the triangle case, the area of the circular segment needs to be a signed quantity in the general case. The sign of the area of the circular segment can be determined by:

$$t = \mathbf{u}_z \cdot (\mathbf{r}_{ct\to P1} \times \mathbf{r}_{ct\to P2}) \tag{2.62}$$

$$A_{cs} = \begin{cases} A_{cs} & t \geq 0 \\ -A_{cs} & t < 0 \end{cases} \tag{2.63}$$

A running total is kept of the area (A_{rt}) and the centroid vectors scaled by the area ($\mathbf{c}_{rt}$) as follows:

$$A_{rt}(i+1) = A_{rt}(i) + A_t + A_{cs} \tag{2.64}$$

$$\mathbf{c}_{rt}(i+1) = \mathbf{c}_{rt}(i) + A_t\mathbf{c}_t + A_{cs}\mathbf{c}_{cs} \tag{2.65}$$

After all of the segments of the cross section have been evaluated, the centroid can be determined by:

$$\mathbf{c} = \frac{\mathbf{c}_{rt}}{A} \tag{2.66}$$

Once the centroid and area have been determined, the inertial properties of the cross section about the centroid are evaluated. The same arc-line approximation is used to create the same triangles and circular segments. Calculation of the inertial properties of triangles (I_t) proceeds as follows using the variables defined in Figure 2.6[48]:

$$b = \mid \mathbf{r}_i \mid \tag{2.67}$$

$$h = \mathbf{r}_{i+1} \cdot \frac{(\mathbf{u}_z \times \mathbf{r}_i)}{b} \tag{2.68}$$

$$c = b - \frac{\mathbf{r}_{i+1} \cdot \mathbf{r}_i}{b} \tag{2.69}$$

$$I_{t1} = \frac{bh^3}{12} \tag{2.70}$$

$$I_{t2} = \frac{bh}{12}(3b^2 - 3bc^2) \tag{2.71}$$

$$I_{t12} = \frac{bh^2}{24}(3b - 2c) \tag{2.72}$$

It is important to note that it is possible for h to be negative, that will create negative moments of inertia. Although these cannot exist in a physical sense, they exist computationally to serve the same function as the negative areas, compensating for nonconvex cross sections.

With the inertial properties of the triangle computed with respect to itself, it is necessary to convert the inertial properties to the desired coordinate system. This is accomplished as follows[104]:

$$\phi = 2cos^{-1}\left(\frac{\mathbf{u}_x \cdot \mathbf{r}_i}{b}\right) \tag{2.73}$$

$$I_{tx} = \frac{1}{2}\{(I_{t1} + I_{t2}) + (I_{t1} - I_{t2})cos(\phi) - 2(I_{t12})sin(\phi)\} \tag{2.74}$$

$$I_{ty} = \frac{1}{2}\{(I_{t1} + I_{t2}) + (I_{t1} - I_{t2})cos(\phi) + 2(I_{t12})sin(\phi)\} \tag{2.75}$$

$$I_{txy} = \frac{1}{2}(I_{t1} - I_{t2})sin(\phi) + I_{t12}cos(\phi) \tag{2.76}$$

that are combined into an inertial vector $\mathbf{m}_t$ defined as follows:

$$\mathbf{m}_t \equiv [I_{tx}, I_{ty}, I_{txy}] \tag{2.77}$$

For the circular segments, computation of inertial properties proceeds as follows, using the variables defined in Figure 2.7. The inertial properties are first computed about the center of the circular segment and an axis through the centroid[48]:

$$I_{cs1} = \frac{r^4}{4}\{\theta - sin(\theta)cos(\theta) + 2\,sin^3(\theta)cos(\theta)\} \tag{2.78}$$

$$I_{cs2} = \frac{r^4}{12}\{3\theta - sin(\theta)cos(\theta) - 2\,sin^3(\theta)cos(\theta)\} \tag{2.79}$$

$$I_{cs12} = 0 \tag{2.80}$$

Again, the inertial properties must be transformed into the proper coordinate frame. This is done using equations 2.74, 2.75 and 2.76 with the angle of rotation ϕ determined by:

$$\phi = cos^{-1}(\mathbf{u}_y \cdot \mathbf{u}_{mid}) \tag{2.81}$$

Once the orientation is correct, the transformation is performed using the parallel axis theorem, modified to use vectors. First, the following definitions of a planar inertia vector and vector multiplication are made:

$$\mathbf{m}_i \equiv [I_x, I_y, I_{xy}] \tag{2.82}$$

$$\mathbf{v}_i\mathbf{v}_j \equiv [v_{ix}, v_{iy}, v_{iz}] \begin{bmatrix} v_{jx} \\ v_{jy} \\ v_{jz} \end{bmatrix} \tag{2.83}$$

The parallel axis theorem can be stated as:

$$I_x = I_{xc} + Ad^2{}_x \tag{2.84}$$

$$I_y = I_{yc} + Ad^2{}_y \tag{2.85}$$

$$I_{xy} = I_{xyc} + Ad_yd_x \tag{2.86}$$

where I_{xc} and I_{yc} are the moment of inertias about the $\mathbf{x}$ and $\mathbf{y}$ axis through the centroid of the area. Using the above definition of vector multiplication and the parallel axis theorem, the computation of planar inertias about an arbitrary location follows:

$\mathbf{r}_t$ = a vector from the desired location to the centroid of the area

$$\mathbf{r}_i = \mathbf{r}_t\mathbf{r}_t \tag{2.87}$$

$$\mathbf{r}_{iz} = \mathbf{r}_{tx}\mathbf{r}_{ty} \tag{2.88}$$

$$\mathbf{m}_{it} = A_i\mathbf{m}_{ic}\mathbf{r}_i \tag{2.89}$$

For the circular segment, the inertial properties about the desired axis m_{csx}, m_{csy} and m_{csxy} are determined by:

$$\mathbf{i}_t = [I_{cs1t}, I_{cs2t}, I_{cs12t}] \tag{2.90}$$

$$\mathbf{r}_t = \mathbf{r}_{t\to c_{cs}}\mathbf{r}_{t\to c_{cs}} \tag{2.91}$$

$$r_{tz} = r_{t\to c_{cs}x}r_{t\to c_{cs}y} \tag{2.92}$$

$$\mathbf{m}_{cs} = A_{cs}\mathbf{i}_t\mathbf{r}_t \tag{2.93}$$

As with the area and centroid calculations, a running total is kept for the inertial properties $\mathbf{m}_{rt}$ as follows:

$$\mathbf{m}_{rt}(i+1) = \mathbf{m}_{rt}(i) + \mathbf{m}_t + \mathbf{m}_{cs} \tag{2.94}$$

Once the area, centroid and inertial properties are determined for the desired location and orientation, the radius of gyration and polar moment of inertia are determined by:

$$\mathbf{u}_{2d} = [1,1,0] \tag{2.95}$$

$$\mathbf{g}_i = \sqrt{\frac{\mathbf{m}_i\mathbf{u}_{2d}}{A_i}} \tag{2.96}$$

$$I_p = m_{ix} + m_{iy} \tag{2.97}$$

Calculation of the section modulus requires determination of the distance from the centroid to the extremes of the cross section. The extremities of the cross section Mp_i are found using bounding box routines provided in the Alpha_1 environment in terms of a maximum and minimum point. From this information the extreme distances are determined and the section moduli calculated as follows:

$$\begin{aligned}
\mathbf{r}_{max} &= \text{vector from centroid to maximum point} \\
&\quad \text{of the bounding box } Mp_i \\
\mathbf{r}_{min} &= \text{vector from the minimum point of the} \\
&\quad \text{bounding box } Mp_i \text{ to the centroid} \\
r_{ext}x &= \begin{cases} r_{max}x & r_{max}x \geq r_{min}x \\ r_{min}x & otherwise \end{cases} \\
r_{ext}y &= \begin{cases} r_{max}y & r_{max}y \geq r_{min}y \\ r_{min}y & otherwise \end{cases} \\
r_{ext}z &= 0 \\
\mathbf{s}_i &= \frac{\mathbf{m}_i\mathbf{r}_{ext}}{|\,\mathbf{r}_{ext}\,|^2}
\end{aligned} \tag{2.98}$$

The sign of the resulting planar properties has a directional dependence, as positive results will be obtained from cross sections that have a counter-clockwise or positive angular orientation. Cross sections that have a clockwise or negative angular orientation will return negative area and inertial properties. This was intentional, as computation of cross sections composed of multiple curves, such as hollow sections, becomes a problem of computation of the properties of the individual sections, finding the centroid and area of the combined cross sections and then transforming and summing of the inertial properties as previously described. This provides a very powerful tool for determination of the planar properties of combinations of very complex cross sections. To date, implementation of accurate algorithms for computation of the torsional constant Tc_i, the shear center sc_i and the warping constant wc_i have not been implemented or tested. Algorithms for these properties are more complex and are seldom implemented in most commercial CAD packages. This functionality was not needed for this project but could become valuable for future work.

2.3.2 Properties of Solids

The inertial properties of solid objects are necessary for dynamic analysis and as such, were implemented for this project. Existing support for calculation of the inertial properties for general objects created with the Alpha_1 modeling system exists as a separate, stand alone program. However, no representation for inertial properties existed in the Alpha_1 modeling system or tools for manipulation of inertial properties. As such, the following representation for the inertial properties of solids was implemented.

$$\mathbf{In}_i = \begin{cases} \mathbf{C}_i & \text{the reference coordinate frame} \\ \mathbf{I}_i & \text{the inertia tensor } (3 \times 3 \text{ matrix}) \\ m_i & \text{a number representing the mass} \\ \mathbf{p}_{com} & \text{a point at the center of mass} \end{cases}$$

The inertia tensor $\mathbf{I}_i$ is defined as a 3×3 matrix as follows:

$$\mathbf{I}_i = \begin{bmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{xy} & I_{yy} & I_{yz} \\ I_{xz} & I_{yz} & I_{zz} \end{bmatrix}$$

As is the case for the planar properties object, the relevant slots (inertias tensor $\mathbf{I}_i$ and center of mass $\mathbf{p}_{com}$) are referenced in the local coordinate frame $\mathbf{C}_i$.

The above representation is useful for the general case and a limited set of tools for computation and manipulation of inertia objects were implemented. For this project, the inertial properties of interest were typically combinations of beams. For a straight beam of general cross section, the components of the inertia tensor about the centroid of the beam ($\mathbf{p}_{cent}$) is calculated as follows[128]:

$$\begin{aligned} L &= \text{the length of the neutral axis of the beam} \\ \mathcal{I}_{ij} &= \text{second moment of area of cross section} \\ A &= \text{area of cross section} \\ I_{xx} &= m\left(\frac{\mathcal{I}_{xx}}{A} + \frac{L^2}{12}\right) && (2.99) \\ I_{xy} &= m\frac{\mathcal{I}_{xy}}{A} && (2.100) \\ I_{yy} &= m\left(\frac{\mathcal{I}_{yy}}{A} + \frac{L^2}{12}\right) && (2.101) \\ I_{zz} &= m\left(\frac{\mathcal{I}_{xx} + \mathcal{I}_{yy}}{A}\right) && (2.102) \\ I_{xz} &= I_{yz} = 0 \quad (\text{ by symmetry }) && (2.103) \end{aligned}$$

The inertia properties of nonlinear beams are calculated by subdividing the beam axis using a linear approximation and calculating the inertia properties of the individual segments. These inertia properties are then transformed and summed for determination of the total beam inertia. As the transformation and summation operations are useful for the general case, their implementation will be discussed in greater detail.

The rotation of an inertial tensor from one coordinate frame to another is accomplished as follows[69]

$$\mathbf{C}_{a\to b} = \text{the } 3\times 3 \text{ rotation matrix between coordinate frame } \mathbf{C}_a$$

$$\begin{aligned} & \text{and coordinate frame } \mathbf{C}_b \\ \mathbf{C}^T_{a\to b} &= \text{the transposition of matrix } \mathbf{C}_{a\to b} \\ \mathbf{I}_b &= \mathbf{C}^T_{a\to b}\mathbf{I}_a\mathbf{C}_{a\to b} \end{aligned} \tag{2.104}$$

and the inertia about a different location is found by first rotating the inertia to the desired orientation as shown above and then performing a translation as follows[128]:

$$\begin{aligned} \mathbf{r}_{b\to a} &= \text{a vector from the origin of coordinate frame } \mathbf{C}_b \\ & \quad \text{to the origin of coordinate frame } \mathbf{C}_a \\ \mathbf{I}_b &= \mathbf{I}_a + m \begin{bmatrix} (r_y^2 + r_z^2) & -r_x r_y & -r_x r_z \\ -r_x r_y & (r_x^2 + r_z^2) & -r_y r_z \\ -r_x r_z & -r_y r_z & (r_x^2 + r_y^2) \end{bmatrix} \end{aligned} \tag{2.105}$$

The summation of inertia tensors is accomplished by first transforming them to the same coordinate frame and then by simple addition of the matrices. The addition of the more general inertia objects were implemented as follows:

$$\begin{aligned} \mathbf{In}_a,\ \mathbf{In}_b &= \text{the inertia objects to be added} \\ \mathbf{In}_{b2} &= \mathbf{In}_b \text{ transformed to coordinate frame } \mathbf{C}_a \\ \mathbf{In}_c, &= \text{the new inertia object} \\ \mathbf{C}_c &= \mathbf{C}_a \qquad (2.106) \\ \mathbf{I}_c &= \mathbf{I}_a + \mathbf{I}_{b2} \qquad (2.107) \\ m_c &= m_a + m_b \qquad (2.108) \\ \mathbf{p}_c &= \frac{m_a\mathbf{p}_a + m_{b2}\mathbf{p}_{b2}}{m_c} \qquad (2.109) \end{aligned}$$

It should be noted that the resulting sum is referenced to coordinate frame $\mathbf{C}_a$. If the inertia about a different frame is desired, then the transformation will have to occur either before or after the summation.

The implementation of the inertia object and transformation and summation tools provides an expanded ability to perform engineering analysis inside the Alpha_1 modeling system. Future work should include integration of the existing inertia computation program with the inertia object representation as well as extension of inertial calculations to other data types.

2.4 Catalog Components

The selection of available components is a necessary component of the designers task. These components impose significant constraints on the design and it is almost impossible to design and manufacture anything without having to use materials and/or components produced by other manufacturers. In addition, the discrete nature of these available components implies a discrete optimization process where new components are selected and replaced as the design is refined. Unfortunately, support for catalog components is limited in most CAD systems. At best the CAD system will support manually constructed geometric information, but most, if not all, engineering information is excluded. Thus, the designer is still reduced to digging back through the catalogs and manually performing and documenting the same engineering checks every time a component is changed. This process is both time consuming and error prone, as it is not unusual for some of the engineering checks to be forgotten or their documentation to be missing or incomplete. The problem becomes even worse when modifying old designs, as the individuals who performed the original engineering checks may no longer be involved, leaving the current designers to try to reconstruct the original thought process.

The catalog work performed during this project was an initial attempt to integrate both geometric and engineering information about catalog components into the CAD environment. It was based upon previous catalog work performed in the Alpha_1 modeling system that involved integrating bearings and fasteners into the modeling and manufacturing environment. In addition, algorithmic selection routines were implemented to reduce, if not eliminate, the necessity for manual intervention during refinement of the design. Although no attempt is made to suggest that this is a comprehensive solution to the problem of the proper selection of discrete components, it is significantly better than other currently available solutions and contains properties of over 2500 common beam cross sections as well as linear actuators and material properties. The catalog extensions are divided into logical sections, each of which will be discussed in detail.

2.4.1 Material Information

The first catalog required for this project was a collection of the engineering information about some commonly used materials. This was in no way a comprehensive list of all the material information that may be needed. In particular, the inclusion of machineability data and heat treatment condition would be very useful. However, sufficient information is included to allow the algorithmic creation of FEA models as well as provide some support for some simple engineering analysis. A material information object was created to organize this information. The current implementation includes the following slots:

$$\mathbf{Mi}_i = \begin{cases} Desc & \text{a text description of the material} \\ MatGrp & \text{a material group identifier} \\ MatID & \text{a unique material identifier} \\ E & \text{the elastic modulus of the material} \\ G & \text{the shear modulus of the material} \\ \nu & \text{Poisson's ratio for the material} \\ \sigma_y & \text{the yield stress of the material} \\ \sigma_u & \text{the ultimate stress of the material} \\ \rho & \text{the mass density of the material} \\ \alpha & \text{the coefficient of thermal expansion} \end{cases}$$

The $MatGrp$ identifier was included to allow for restricted algorithmic selection of different materials from a common group. Although this feature was not implemented during this project, it was felt that the capability would be useful in the future. An additional problem that occurs when integrating material information into a CAD environment is that the issue of units can no longer be ignored. Typical CAD systems can ignore the units until such time as external communication (drawings, IGES, etc.) are necessary. However, for material information, the choice of units needs to be known. As it is undesirable to both enter and store the material information in different units, the decision was made to create the catalog using only SI units and provide the necessary conversions to other systems. It should also be noted that the density ρ used in the material information object is a mass density and as such will have to be converted to a weight when that information is needed. The current catalog contains materials sufficient for implementation and integration into this project but in no way pretends to contain all materials that

may be needed or desired. Additional materials may be added by the user as the need arises.

2.4.2 Beams

The selection and manipulation of standard beam members was an integral part of this project. From a geometric construction standpoint, beams may be defined as a shape generated by a sweep with a constant cross section. This was reflected in the representation of beams implemented during this project, that separates the information related to the cross section and material from information related to the sweep. The definition of the beam object $\mathbf{Bm}_i$ used for this project was as follows:

$$\mathbf{Bm}_i = \begin{cases} \mathbf{C}_i & \text{the reference coordinate frame} \\ C_i & \text{the centroidal axis curve} \\ \mathbf{o}_i & \text{a vector describing the orientation of the } \mathbf{x} \\ & \text{axis of the cross section} \\ \mathbf{Bi}_i & \text{a beam information object} \end{cases}$$

It should be noted that the beam object contains information related to the sweep such as the sweep axis C_i and the orientation of the cross section $\mathbf{o}_i$ but that all the cross sectional information is stored in the beam information object $\mathbf{Bi}_i$. This makes it very easy to link several beam objects to a single beam information object and thus allows very simple change propagation.

The definition for the beam information object used for this project contains the following information:

$$\mathbf{Bi}_i = \begin{cases} Cs_i & \text{a curve describing the cross sectional geometry} \\ \mathbf{Pi}_i & \text{a planar properties object} \\ \mathbf{Mi}_i & \text{a material information object} \\ ty_i & \text{a unique identifier list} \\ si_i & \text{a unique numerical identifier} \end{cases}$$

The beam catalog created for this project contains information sufficient to generate typical cross sections such as square and round tubing. From this information, the planar properties for the cross section are calculated. This approach greatly reduced the amount of catalog information that had to be entered, but not without imposing some computational penalties. Even with efficient algorithms for

planar integration, this process currently may take a few seconds to complete. The migration of the system to a faster language and hardware should greatly reduce this delay and integration of the beam information objects into a persistent object data base would eliminate the need for recomputation each time a beam is selected from the catalog.

A larger problem with the computation of the planar properties from cross section curves is that the information generally available is insufficient to accurately define the cross section. For example, for the channel section shown in Figure 2.8, the catalog supplies values for d, b_f, t_f and t_w as well as the angle θ that is the same for most C and S sections[3]. However the radii r_1 and r_2 and the location where the flange width is measured t_{fx} are not available. This insufficient definition necessitated experimental derivation for the unknown variables. A starting value was thus selected for each unknown variable. Integration and computation of the resulting planar properties was performed and the results compared with the catalog values. The unknown variables were then modified and the process repeated until close agreement was reached with the catalog values. For the example shown in Figure 2.8, the following relationships for the unknown variables were determined:

$$r_1 = \frac{t_w}{2} \tag{2.110}$$

$$r_2 = \frac{t_w}{8} \tag{2.111}$$

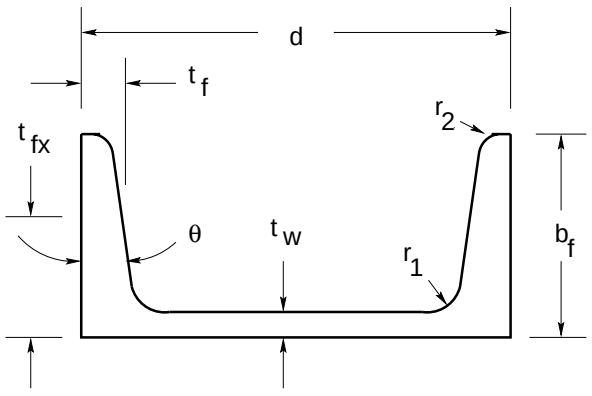

Figure 2.8. Channel section drawing detail showing the relevant variables for complete definition of the cross section.

$$t_{fx} = b_f(1 - 0.42t_f cos(\theta)) \quad (2.112)$$

For the cross sections included in the beam catalog, agreement was reached within 2% of the catalog values for the complete range of sections, with most sections being within 0.5% of the catalog values.

The current catalog has a complete range of the following sections: HP, channel(C), square tubing, rectangular tubing, and round tubing totaling about 2500 different sizes. Additional types of sections, including the popular W and angle sections, are being added as time permits.

2.4.3 Actuators

The class of actuators implemented was restricted to linear fluid actuators for several reasons. First, these types of actuators are directly coupled to the mechanism, so the difficulty of designing a mechanical amplification system is unnecessary. Second, the power/weight ratio of these types of actuators is quite high, such that remote mounting of the actuator is seldom necessary. Third, similar sizes and types of these actuators are readily available from a large number of suppliers for reasonably low costs making them economically attractive. Last, the industry has settled into a limited number of common sizes, with direct substitution possible between several different manufacturers. Thus, construction of a manufacturer independent catalog is a realistic possibility.

The representation used for this project for the linear fluid actuator $\mathbf{Af}_i$ was defined as follows, with the relevant geometric parameters identified in Figure 2.9.

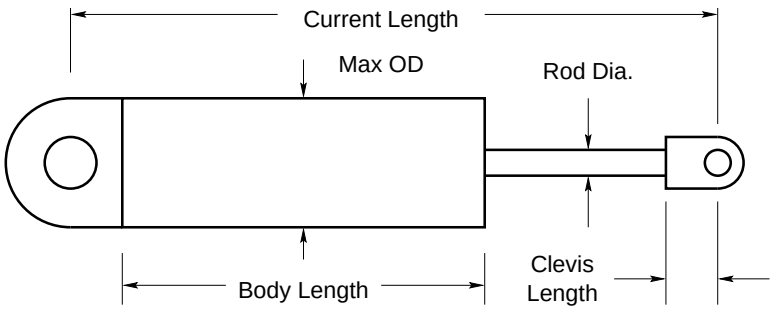

Figure 2.9. Fluid actuator and relevant geometric parameters.

$$\mathbf{Af}_i = \begin{cases} \mathbf{C}_i & \text{the local coordinate frame} \\ D_i & \text{a text description of the actuator} \\ b_i & \text{the actuator bore} \\ s_i & \text{the actuator stroke} \\ rd_i & \text{the rod diameter of the actuator} \\ MaxP_i & \text{the maximum allowable actuation pressure} \\ WrkP_i & \text{the designed actuation pressure} \\ MinL_i & \text{the minimum actuator length} \\ MaxL_i & \text{the maximum actuator length} \\ CurL_i & \text{the current actuator length} \\ MaxOD_i & \text{the maximum outer diameter of the actuator} \\ BL_i & \text{the length of the actuator body} \\ CL_i & \text{the length of the actuator clevises} \end{cases}$$

Actuators were defined as having the base pivot located at the origin of $\mathbf{C}_i$ and extending in the $\mathbf{x}$ direction a distance $CurL_i$ to the second attachment point. With this representation reasonable geometric modeling and analysis of these types of actuators is possible and has been implemented into the system.

This representation was conceived and implemented very early in the project, and although performing its intended function reasonably well, significant modifications are recommended for future implementations. One of the problems with the actuator representation is that it is includes some redundancy. For example, once the minimum length $MinL_i$ and stroke s_i are known the maximum length must be $MaxL_i = MinL_i + s_i$. A second problem is that differing clevis lengths are not allowed, nor are noncircular body cross sections. As both of these occur in practice, support for this functionality should be added. Another nonobvious problem with the current representation arises from how the actuator is defined in the local coordinate frame. Most applications for these types of actuators involve rotation of one component in relationship to the other. As the current representation defines the actuator to be in the direction of the $\mathbf{x}$ axis, correctly modeling the actuator with respect to the mating structures requires reorientation of the local coordinate frame as a function of the angular rotation between the structural components. It is also difficult to determine the correct position and orientation when given the current actuator length ($CurL_i$). For these reasons the following modifications to the actuator representation are suggested for future implementations:

- Eliminate the maximum length ($MaxL$) slot and just perform the $MinL + s_i$ arithmetic whenever needed.
- Modify the maximum diameter ($MaxOD$) slot to contain a curve or a number. This would allow for representation of noncircular body cross section.
- Include a new center of rotation ($\mathbf{p}_{rot}$) slot, that would contain a 2-D point in the local **xy** plane that describes the local center of rotation in the local coordinate frame.
- Include a new moment arm ($\mathbf{r}_m$) slot, that would contain a vector representing the end position of the actuator when $CurL = MinL$.
- Redefine the location of the actuator as being from the origin of $\mathbf{C}$ to the intersection of two circles, the first described by a radius $CurL$ about the origin and the second described by a radius $\|\mathbf{r}_m\|$ about the point $\mathbf{p}_{rot}$. Which of the two intersection points should be used would be determined by which side of the line from the origin to $\mathbf{p}_{rot}$ the vector $\mathbf{r}_m$ lies on. If no $\mathbf{r}_m$ then it would assume the actuator lies along the $\mathbf{x}$ axis.
- Redefine the existing clevis length slot CL_i to be the end clevis length CLe_i and define a new base clevis length slot CLb_i. These slots should also be modified to allow clevis object data types as well as numerical data, to allow for more flexibility in modeling the actuators.
- Provide support for different actuator base mounting systems such as trunnion or flange.

With the above modifications, the actuator representation would become easier to use and better represent the variety of actuators available. Future improvements could include the modeling of the fluid attachments, as well as flexible elements such as hoses and wires.

For the current project a limited set of high pressure actuators has been integrated into catalog form. These range in size from 3/8" to 4" bore and can include built-in position sensors, larger rod sizes and load sensors. The pressure,

specification is left up to the user although the actuator geometry is defined for actuators with a working pressure of 3000 psi. Future implementations should also include a wider range of components, particularly some of the ISO and stainless steel body pneumatic actuators, that are widely used and readily available.

2.4.4 Clevises

The final catalog object describes clevises. During implementation it became apparent that a representation for some form of hinge would be necessary for the project. For this purpose a representation for clevises was implemented and proved very useful, including being integrated into the actuator representation. The definition of the clevis object $\mathbf{Cv}_i$ is as follows:

$$\mathbf{Cv}_i = \begin{cases} \mathbf{C}_i & \text{the local coordinate frame} \\ Gm & \text{a geometric representation of the clevis} \\ Lc & \text{the length from the base to the pivot point} \\ Wc & \text{the width of the clevis} \\ Tc & \text{the thickness of the clevis} \\ Sc & \text{the space between the centers of a female clevis} \\ Ty & \text{a text description of the clevis type} \\ \mathbf{Pc}_i & \text{the planar properties of the clevis} \\ \mathbf{Ic}_i & \text{the inertial properties of the clevis} \\ \mathbf{Mc}_i & \text{the material properties of the clevis} \end{cases}$$

By default, the user may include any shape in Gm to represent the geometry; however, the planar and inertial properties are computed assuming rectangular cross sections and the values stored in the Lc, Wc, Tc, and Sc slots. Thus, some care must be taken to ensure reasonable values in these slots if a nondefault geometry is desired. The default geometry and relevant parameters of a clevis object are shown in Figure 2.10. To ease the geometric construction and increase drawing speeds, no shaft hole or bearing bore were included in the default clevis geometry. As computer systems get faster and CAD systems evolve to support variable levels of detail, a step-bore feature should be added to the geometric representation to better reflect the actual clevis geometry.

A large number of algorithmic routines for the construction of clevises were implemented allowing a wide selection of input options. Included among these,

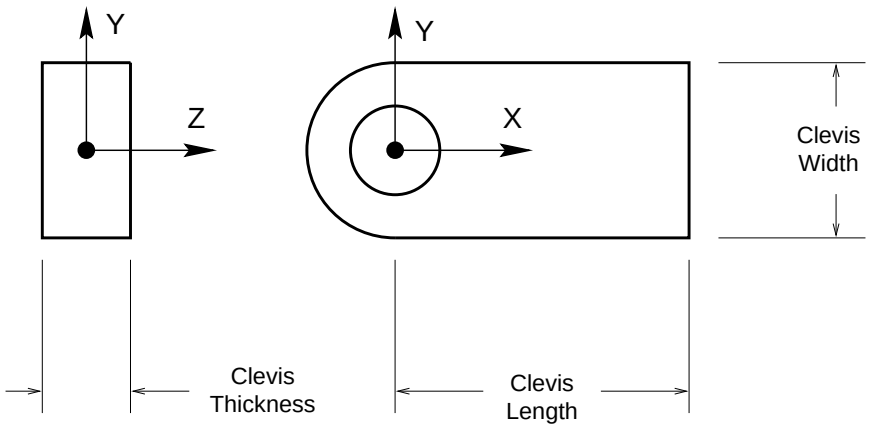

Figure 2.10. Default clevis object and relevant geometric parameters.

are algorithmic routines that construct mating clevises for either male or female clevises. These routines have proven extremely useful during the course of this project and specific details are available in the DynaFrame User's Manual.

2.5 Link Structures

Having defined data structures and methods for the representation and manipulation of linkages, loads, inertial properties and catalog components large aggregate objects can be defined. The principal goal of this project was the design of large dynamic structures, therefore a representation for these structures was necessary. This representation contains all of the individual components, beams, clevises and actuators, relating to the complete structure arranged in a manner to allow easy access and modification of individual components. The representation used in this project for the link structure object $\mathbf{Ls}_i$ was defined as follows:

$$\mathbf{Ls}_i = \begin{cases} \mathbf{Ln}_i & \text{the link defining the kinematics of the structure} \\ Pb & \text{the base hinge points of the structure} \\ Pe & \text{the end hinge points of the structure} \\ Mb & \text{the beam members that form the base of the structure} \\ Me & \text{the beam members that form the end of the structure} \\ Ms & \text{the beam members that join the base and end} \\ Cb & \text{the base clevises} \\ Ce & \text{the end clevises} \\ \mathbf{Af}_i & \text{the fluid actuator that moves the structure} \\ Pa & \text{the actuator mount points} \\ Ca & \text{the actuator mount clevises} \\ De & \text{a text description of the structure} \\ La & \text{the link structures attached to this link structure} \end{cases}$$

It should be noted that the link structure representation, like the link representation, is recursively defined with each link structure containing any external structures attached to it. A further description of the different sections and components of the link structure and how the link structures are constructed is the subject of the next chapter.

CHAPTER 3

STRUCTURAL SYNTHESIS

The principal purpose of adding the structural synthesis capability to this project was to allow a design engineer to quickly construct a reasonable structure for analysis. Currently, designers spend weeks of effort modeling and sufficiently detailing structures for analysis, only to find that they cannot carry the required loads, or meet the dynamic performance requirements. In either case, the designer is faced with the task of repeating a significant percentage of the work, hoping that the modifications will be proven sufficient to eliminate the problems.

The structural synthesis algorithms that were developed and implemented during this project are designed to rapidly construct simple reasonable structures that can be used for dynamic and structural analysis and could be manufactured with minor detailing modifications.

3.1 Background

Most of the past work performed in the area of structural synthesis would be better defined as structural optimization, although the difference is neither clearly defined nor absolute. In general, the structural synthesis problem can be broken down into three major areas:

Topological Synthesis during which the fundamental structural geometry is generated. During this process, the number of structural members and the related connections are determined.

Shape Optimization during which the location of the connection point is modified to provide improved performance.

Sizing Optimization during which the cross sectional properties of the structural members is modified to meet the design requirements.

There is a great degree of coupling between these three areas, and a good deal of work has been performed with the intent to integrate two or more of these areas. For purposes of this dissertation, structural synthesis is equivalent to *Topological Synthesis* and can be defined as: *The initial definition or major modification of structural geometry.* The intent is to separate out algorithms that construct geometry from those that just modify existing geometry. Thus, constructing the center axis of truss members would be considered structural synthesis, and changing the sizes of the truss members to minimize structural weight would be considered structural optimization. The *Shape Optimization* and *Sizing Optimization* problems will be discussed in the chapter on structural optimization.

The initial effort at structural synthesis is generally credited to Mitchell[91] who started from Maxwell's [90] work that stated that an optimal structure is one in which all of the members undergo an equal stress. Given that assumption, Mitchell derived several types of equal stress structures for different load and boundary conditions. Most of these were 2-D structures; however, an example of a 3-D axially symmetric structure subject to torsional loading was also derived. In Mitchell structures, the members are only subject to axial loads; bending stresses are neglected, even in curved members. The axial load assumption is quite common in engineering, and structures designed using this criterion are referred to as *truss structures.* The neglect of bending stresses and buckling are more serious and reduce the practical application of this approach. Mitchell's work has been extended from the initial static loading assumptions to include both creep and frequency[62]. This work also included a proof of a unique optimum layout for a given set of boundary conditions. This leads to Mitchell structures often being referred to as *least weight trusses* and examples of this type are often used as test examples for other approaches to topology optimization. Mitchell structures typically are quite dense, being composed of a very large number of members. This density, as well as

the restricted set of boundary conditions, mathematical complexity, and fabrication difficulties, result in Mitchell structures being primarily of academic interest.

An additional methodology for determining optimal truss topology which has been developed are known as *Prager structures* [115, 113, 114]. These structures are similar to Mitchell structures, with the exception that the load on all members in the structure have the same sign, i.e., all members are either in compression or in tension. A further restriction in Prager structures is that although the line of action of all loads is defined, the point of application to the structure is not and is a variable in the optimization process. With these constraints, Prager structures form two basic types of structures, arch-grids for compressive structures and cable networks for tension loaded structures. Prager structures also differ from Mitchell structures in that they invariably are "surface structures" in which the axes of all members are contained in a single surface in space. In contrast, 3-D Mitchell structures tend to form dense space frames. The density in both types of structures is typically reduced by application of some optimal criteria [105]. Prager structures can be found for a wider range of load conditions and support locations than Mitchell structures, and the self-weight can be included. These extensions provide practical advantages over Mitchell structures but there still remain significant restrictions on allowable boundary conditions as well as fabrication difficulties that preclude the application of this work to this project.

With the advent of digital computers and mathematical programming approaches to optimization, this technology was rapidly applied to the problem of truss topology optimization. There has been a considerable amount of work performed in this area that can be divided into three fundamental approaches[144]:

1. The *Ground structure approach*[15]. In this approach a grid of points is constructed that includes all of the joints, supports and loading positions. Typically interconnection of all points is assumed and various members are removed during the optimization process. An alternative formulation is to start with a minimal structure and add elements and optimize[15].

2. The *Geometric approach*[59]. In this approach, an initial topology is assumed and the joint locations are considered design variables along with the member sizes.

3. The *Hybrid approach*[76, 14]. These approaches generally divide the problem into two groups, one consisting of the topology and the other the member sizes. Either the ground structure or geometric approach can be used initially, followed by a number of possible mathematical optimization methods.

Almost every mathematical optimization approach has been tried, with a large amount of work going into the problem formulation and various schemes for computation of the various search parameters. An excellent summary of the current state of the art of this field was written by Topping in 1983[144]. Other relevant summary papers that concentrated on the use of finite element approaches to the optimization of solid shapes include Haftka et al.[57] and Ding[37]. Despite a great deal of effort, none of these approaches have proven particularly popular. Most are very sensitive to initial conditions and local minima, a situation that is exacerbated by the discrete nature of member sizes.

With the advent of computational analysis tools such as finite element analysis (FEA), a couple of general topology optimization schemes have been developed that take advantage of this technology. The first is known as the *homogenization method* in which the allowable design space is meshed for FEA, assuming a composite material consisting of "material" and "void"[13, 99, 94]. Load and boundary conditions are then applied to this mesh, a FEA is performed and the results are used to modify the density of the material. Areas that are highly stressed have additional material added, while areas of low stress have material removed. Typically the amount of available material is expressed as a percentage of the total volume and is held constant during the optimization process. Variation of this percentage produces differing structures, and experimentation is usually necessary before satisfactory results are achieved. This approach has been shown to produce optimal shapes, such as Mitchell structures[132, 77], or Prager structures[36] when

the appropriate boundary conditions are provided. A discussion of an integrated structural optimization system using this approach is described by Papalambros et al.[99, 30]. The system uses the homogenization method for topology generation, together with an image interpolation module that uses some modified vision algorithms to extract a reasonable topology from the density array. This output is then fed into an optimal shape and sizing algorithm using the method of moving asymptotes[134]. A similar system is described by Olhoff et al.[94] that also tries to extract boundary information from the homogenization mesh. Other FEA based techniques have been proposed[98], with the principal difference being how the structure is inferred from the FEA mesh. Despite differences in the topology extraction methods the results from these systems are similar to those produced by the homogenization method.

Although the homogenization method shows promise, there are still formidable obstacles that must be overcome. The process is computationally expensive, using between 2000 to 6000 material-void elements for reasonably simple 2-D problems. No 3-D work using the homogenization method was located, although there seems to be no apparent reasons other than scale why the work could not be extended to 3-D. In addition, the resulting structure is dependent upon the average density of the design volume, a constant parameter selected at the start of each optimization process. Several optimization iterations are often required using different average densities before a satisfactory design is determined. Even the proponents of this approach view it as a conceptual design tool and suggest rough mesh densities and additional computational methods for extraction of structural geometry from the micro-void mesh[132]. The example problems discussed also use quite simple boundary conditions and analysis criteria, typically a single load case and mean compliance. It is not unreasonable to assume that using this approach for real world loading, stress, and stability constraints would require a substantial increase in the computational complexity. Manufacturing constraints could also prove difficult to incorporate, as they can be hard to quantify, and restrictions on shape could be difficult to incorporate into the process. It is also not unusual to have areas of

variable density in the resulting geometry and interpretation of these areas presents additional difficulties.

Another class of approachs to the problem of topology design are known as *genetic algorithms.* These algorithms try to imitate the biological selection process[66, 24] to arrive at an optimal topology. In these approachs the design space is discretized and each discrete area assigned a binary string representing the "state" of the area. This state may be as simple as "zero" indicating a void and "one" indicating material, or a longer string including more complex information. These binary strings are then joined in some fashion to form a "chromosome" that can be analyzed and checked against the design constraints. Typically a number of chromosomes are randomly constructed to serve as the initial "gene pool." After analysis, chromosomes that are deemed "good" are used as parents for the next generation and "poor" chromosomes are removed from the gene pool. After several "generations" (iterations) have been tested, the design "evolves" from the gene pool. The genetic algorithms consist of a set of rules controlling the method of chromosome reproduction, mutation probability, gene pool pruning, and other parameters of the evolutionary process.

A slightly different genetic approach to truss topology optimization was suggested by Hajela et al.[58] in which the design space was represented by a collection of nodes or points in space that were usually completely connected by a grid of members. The problem was further divided into two parts: the topological optimization problem in which all members were considered to be the same size, and the sizing problem in which the individual size of each member was optimized. The structural geometries generated during the topological optimization were used as the starting "seeds" for the member resizing optimization. For topological optimization, the absence or presence of a member was indicated by a zero or one respectively, and a four digit string was used to represent the size of each member during the resizing process. This subdivision of the optimization process greatly reduced the number of potential permutations, that in turn greatly decreased the required search time. A similar but more complex approach has been tried for the

general mesh approach. Search performance can be further improved by elimination of the kinematically unstable structures. Unfortunately, this representation also restricts the design space to trusses with connection points at the specified locations, thereby eliminating potential optimal solutions.

One of the advantages of the genetic algorithms is that they are discrete by definition and can easily map into discrete catalogs. Unfortunately, in general there are severe computational difficulties with genetic algorithms. The number of possible permutations is 2^n where n is the length of the chromosome. For many problems this can be large enough to impose a severe computational cost that is prohibitive in many applications. The computational requirement is further increased by the large number of "generations" that are required before the reasonable designs evolve. Simple 2-D truss problems using the member grid approach converged in 33 generations[58], compared to the design space mesh approach, that required 250 generations for a 10x16 grid[24]. It is important to realize that each member of each generation must be evaluated every generation, further increasing the computational requirement. It is also important to realize that the genetic algorithm is fundamentally a stochastic method with no guarantee of locating the optimal solution or providing repeatability. Some attempts have been made to decrease the computational complexity by integration of heuristic systems and fuzzy logic[129] but the improvements, although noticeable, appear inadequate to sufficiently reduce the complexity of most real world problems. Soh et al.[129] demonstrate their approach on several 3-D truss structures using a large amount of symmetry and reduction of the number of allowable member sizes to reduce the design space to a manageable size.

Another relatively new approach is *shape annealing* [111, 110]. The shape annealing process typically starts with existing geometry primitives called *shapes* and maps them through a set of rules called *shape rules* to existing geometry that provide scale and location of critical points. For truss optimization, the shapes are usually triangles and the shape rules usually involve division of an existing triangle or addition of a new triangle. The resulting structure is then checked

against the previous optimum. Depending upon the results it may have additional topology modifications, or may be optimized by modifying the connection locations and member sizes using a *simulated annealing* [73] optimization scheme that is loosely analogous to material annealing. In the early stages, the probability of modifications to topology is quite high, and decreases to zero as the optimization process progresses. The set of shape rules also includes a full set of inverse functions so that it is possible for the algorithm to undo modifications to the topology during the optimization process. Unlike some of the other methods, this approach can support multiple types of constraints including stress, buckling and geometric obstacles[110, 26]. It is also possible to use a different methodology for generation of the initial geometry. For example, Anagnostou et al.[4] describe a system using the homogenization method for shape representation and a simulated annealing approach for optimization. In all cases, the example problems discussed in the literature are 2-D although there does not appear to be any significant obstacles, other than computational requirements, to extending the work to 3-D.

Unfortunately, there are still several problems with the shape annealing approach. Like the genetic algorithms, shape annealing is a stochastic optimization process and suffers similar problems with repeatability and nonoptimal solutions. It can be quite sensitive to local minima and the optimization parameters. For example, if the possibility of topology modification is rapidly decreased, the chances of finding the optimal structure are also substantially reduced. If this probability is slowly reduced, a great deal of computational effort may be spent on marginal topologies which are then discarded.

Although all of these approaches are interesting and may prove useful in other circumstances, none proved attractive for this project. The addition of the dynamic loading conditions, and their dependency upon structural mass coupled with the variable actuator size, greatly increase the already large computationally complexity approaches. In addition to the limited set of boundary conditions, the theoretical approaches pioneered by Mitchell also tend to produce very dense unmanufacturable structures. Other mathematical optimization approaches require an initial struc-

tural geometry, avoiding the primary purpose of the structural synthesizer to this project. Extension of the homogenization method to 3-D would greatly increase its already high computational requirements. Trying to couple that with variable loading does not appear promising. The genetic and shape annealing algorithms suffer from similar computational problems, even assuming a connectivity based representation. These methods are also stochastic and so pose the problem of not being repeatable, a potential source of frustration to designers. As even the example problems in the literature do not appear to be optimal without significant manual correction, these approaches were also discarded.

A final approach suggested by Rule[116] starts with a predefined topology with a known number of support and load application points. Using a heuristic approach coupled with member resizing, the truss topology is then "grown" until the support and load points are at the locations specified by the designer. The results of this approach are dependent upon the initial truss configuration. There is no discussion of 3-D structures or multiple load cases, although the extensions do not appear to be difficult. Even though differing significantly from the structural synthesis approach selected and implemented for this project, this work is the most relevant found to date.

3.2 Implementation Details

The approach to structural synthesis that was finally decided upon for this project was a heuristic or expert system approach. In this approach, a set of rules are determined that control how the structure is constructed. The heuristic approach was settled upon for the following reasons:

- The restricted class of problems provided sufficient information to allow a heuristic approach to be successful. Currently the domain is primarily restricted to general 3-D open kinematic chains with linear actuators.
- The nature of the heuristic approach allows a simple implementation and incremental improvement as additional functionality is needed. For example, an extension has been provided for rotation about the linkage axis.

- The heuristic approach is deterministic: given the same input parameters the system will provide the same results. Determinism is an highly desirable attribute to many design engineers.
- Addition of fundamentally different synthesis algorithms is possible with the heuristic approach. It is often desirable to provide structures that are simpler and less expensive to manufacture, at the penalty of some performance.
- It is relatively easy with this approach to include reasonable default behavior. Thus initial designs can be produced with minimal input from the designer and "tweaked" until a satisfactory result is achieved.
- The heuristic approach allows the designers reasonably fine control of the synthesis process. It is possible to provide external control over each of the rules, thus allowing the designer to override the usual behavior.
- With the heuristic approach, it is always possible to provide feedback to the designer. This allows the designer to provide correction for unreasonable designs, reducing the amount of required error checking and related computation.

The heuristics used in this project for structural synthesis can be divided into three classes, the general space frame case, the single beam case, and special cases. The heuristics used for the general space frame case will be discussed first, as the other cases can be viewed as exceptions to some parts of the general algorithm. The approach implemented in the system proceeds as follows:

1. The system determines the location of the base hinge and base actuator mount points.
2. From the base hinge and actuator mount points, the system construct the base hinge clevises as well as the actuator clevis.
3. The system then joins the base hinge clevises and the actuator clevis with the appropriate size beams.
4. The location of all of the end hinge and related actuator mount points is then determined.

5. As with the base clevises, the end hinge clevises and the external actuator clevises are constructed from the end hinge and related actuator mount points.
6. The end hinge clevises and the external actuator clevises are joined with appropriate size beams.
7. The base and end sections are then joined using the minimum amount of the desired beam material.
8. Bracing is added to the structure as is appropriate.

Which heuristics are used is dependent upon the linkage geometry but this alone is insufficient to completely define the structure. Therefore, *attributes* are assigned to each link to further control the synthesis process. These attributes are accessible to the designer and allow them to "tweak" the design until they are satisfied with the result. As implemented, the structural synthesis system will always construct a structure, even if it is a very bad design. This is intentional, as in discussions with design engineers, they expressed a desire for visual design feedback rather than error messages. A simple example structure is shown in Figure 3.1.

This synthesis algorithm can also be viewed by considering the linkage as consisting of a number of *joints* separated in space. The algorithm goes through and constructs the joint details, then finishes up by building the structure that connects the joints. This *joints* view will be taken for a more detailed discussion of the synthesis algorithm.

3.2.1 Joint Synthesis

The term *joint* is used to refer to the interface between two links in a linkage. At this interface, the hinge points must be located, both sides of the hinges constructed, the actuator properly located in relationship to the hinge axis, and the actuator

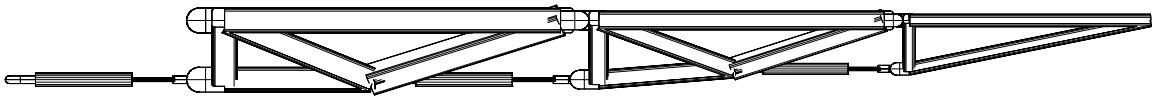

Figure 3.1. A simple three joint example structure.

mounting clevises constructed. Each of these tasks will be discussed in turn.

The construction of the hinge is controlled by the *HingeWidth* attribute that determines the spacing between the hinges and the *HingeOffset* attribute that determines the distance from the link coordinate system to the center of the hinge [Figure 3.2]. If the *HingeWidth* attribute is not defined then by default it is set to the value stored in the end length slot in the link($\mathbf{Lk} : El$) and the default value for the *HingeOffset* attribute will be set to zero, which will center the hinge on the link coordinate frame. The positioning of the actuator mount point is determined by the actuator stroke, link range of motion and the angle between the connecting links. By definition the actuator mount points lie in the link **xy** plane; the remaining problem is the proper positioning of the mount points in this plane. The first piece of information needed is the actuator moment arm r, that is typically calculated from the actuator stroke length and the link range-of-motion as follows:

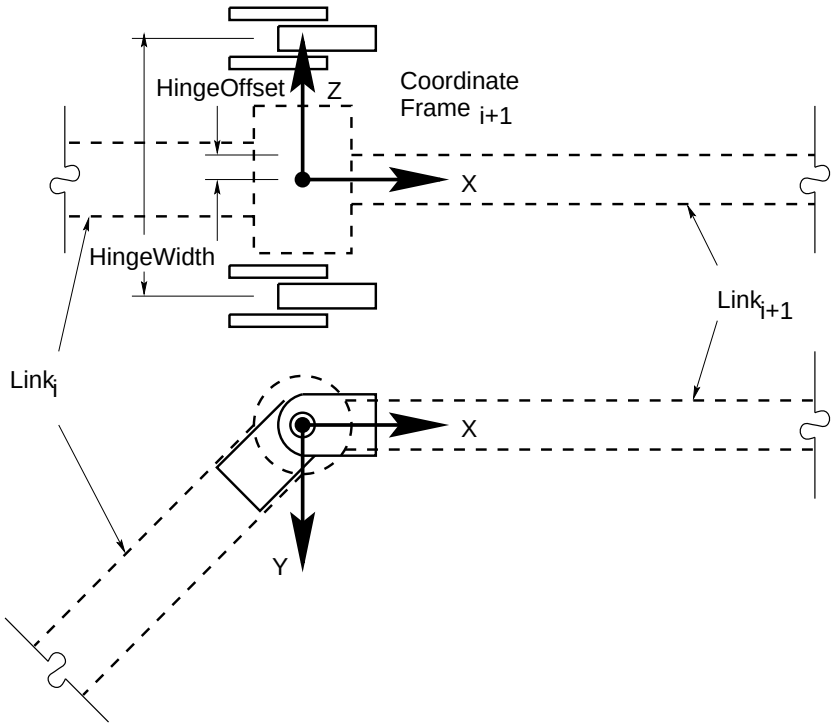

Figure 3.2. Joint showing the *HingeWidth* and *HingeOffset* attributes.

$$r = \frac{A_{stroke}}{2\,sin(ROM/2)} \tag{3.1}$$

$$L_m = \text{actuator minimum length + half the stroke length} \tag{3.2}$$

This provides the correct moment arm if the actuator line of action is perpendicular to the moment arm at the center of the range of motion, which is the default case shown in Figure 3.3. An example of the default hinge and actuator placement geometry produced by these heuristics is shown in Figure 3.4.

However, there are times when it is desirable to position the actuator at a different angle than the default. This is easily accomplished by using the *CylArmAng* attribute, that defines a rotation away from the default case [Figure 3.3]. The default value of this attribute is usually zero, which is parallel with the link for space frame structures. In the single beam case this attribute defaults to an angle designed to allow sufficient distance offset from the structural beam for the necessary mounting clevis.

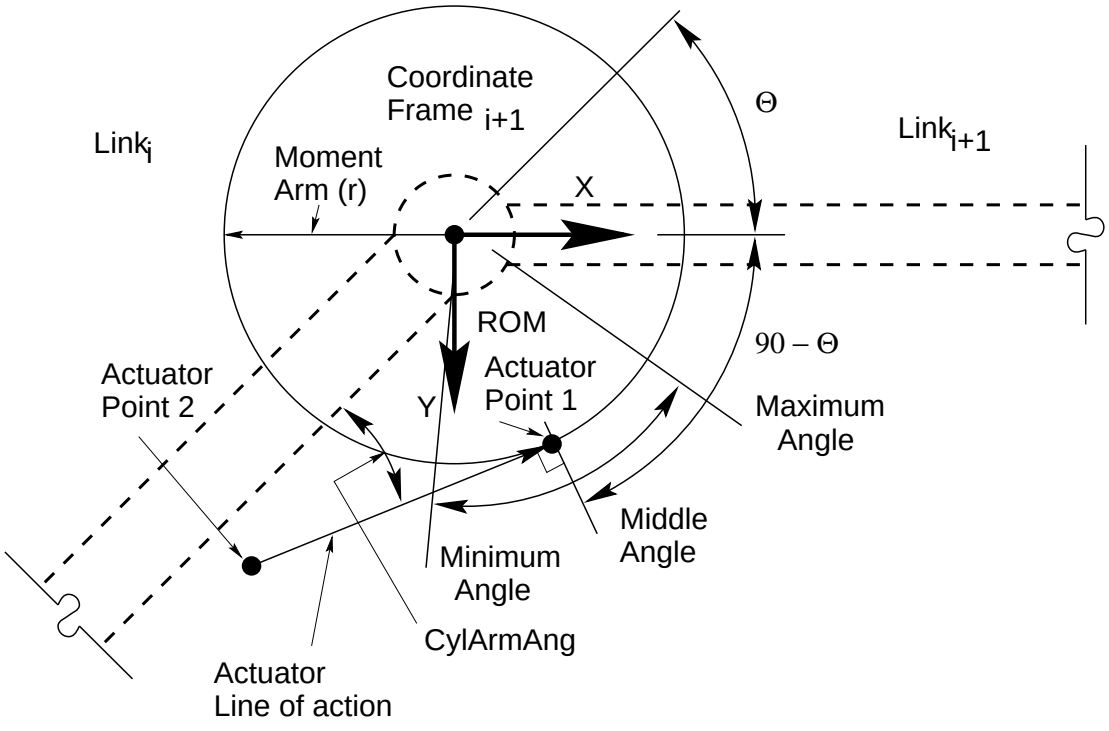

Figure 3.3. Joint showing the actuator point construction and relevant quantities for the default case.

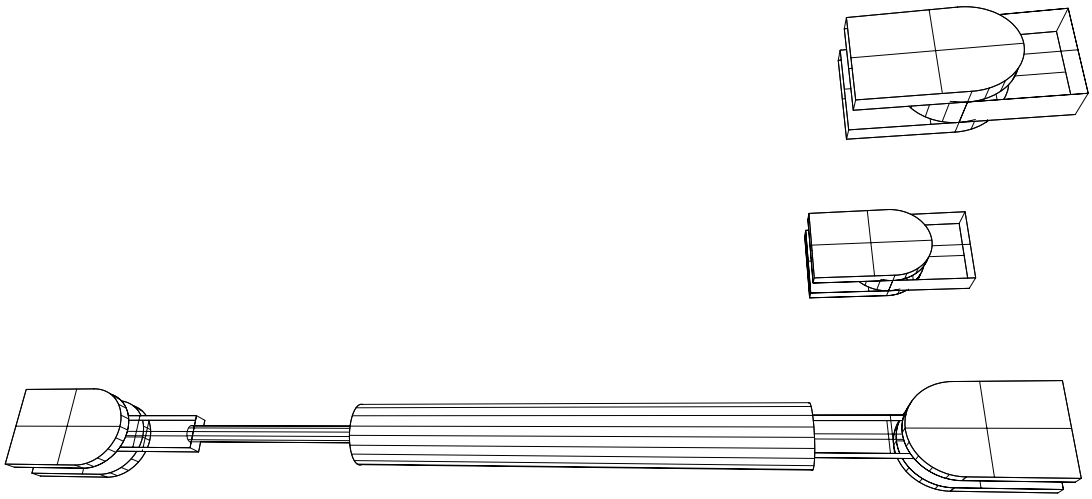

Figure 3.4. The synthesized hinge and actuator mount clevises along with the associated actuator for the second joint in Figure 3.1

The *CylOffsetAng* attribute is used to offset the maximum moment arm position from the center of the range of motion. This is often done to compensate for the maximum required joint torque occurring at one end of the joint range of motion. For this case the moment arm calculation is quite a bit more complex as shown in Figure 3.5. For this case the known quantities are L_1 and L_2 which are the minimum and maximum length of the actuator as well as the joint range of motion(ROM), and the *CylOffsetAng* shown in Figure 3.5 as the variable ν. With these known

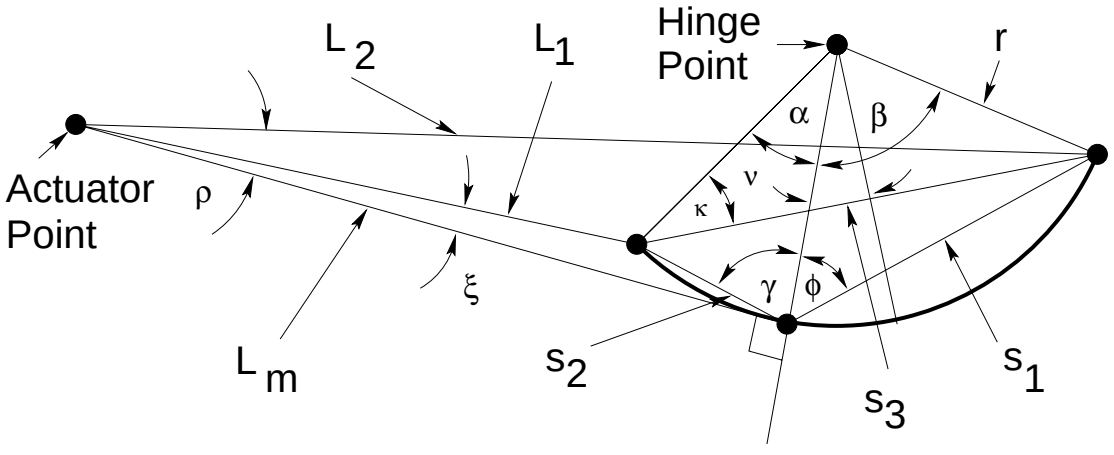

Figure 3.5. Joint showing the actuator point construction and relevant variables for the nondefault case.

quantities the computation of the actuator radius r may be accomplished as follows:

$$\alpha = \frac{ROM}{2} - \nu \tag{3.3}$$

$$\beta = \frac{ROM}{2} + \nu \tag{3.4}$$

$$\gamma = \frac{\pi - \alpha}{2} \tag{3.5}$$

$$\phi = \frac{\pi - \beta}{2} \tag{3.6}$$

$$\kappa = \frac{\pi - ROM}{2} \tag{3.7}$$

$$s_2 = L_1 \left(\frac{sin(\pi - 2\gamma - \phi + \kappa)}{sin(\pi/2 - \gamma)} \right) \tag{3.8}$$

$$s_3 = L_1 \left(\frac{sin(\gamma + \phi)}{sin(\pi/2 - \gamma)} \right) \tag{3.9}$$

$$r = \frac{s_3}{2\, sin(ROM/2)} \tag{3.10}$$

$$\xi = sin^{-1} \left(\frac{s_2\, sin(\pi/2 - \gamma)}{L_1} \right) \tag{3.11}$$

$$L_m = s_2 \left(\frac{sin(\pi/2 - \xi + \gamma)}{sin(\xi)} \right) \tag{3.12}$$

Once the moment arm r is known, the position of the fixed actuator point can be found. There are an infinite number of possible points that would produce the desired actuator line of action, so the default is to place the actuator parallel to its closest link. If, in the center of the range of motion the links meet at an angle[Figure 3.3], the angle at which they meet is bisected and the fixed actuator point is placed perpendicular to the bisection line at a distance of L_m.

Two additional degrees of freedom in positioning the actuator have not yet been discussed. The first is on which side of the link the actuator is to be placed and the second is on which link the actuator will be placed. These various options are shown in [Figure 3.6]. Which side of the link the actuator is placed is controlled by the boolean attribute *CylSide* attribute. The default is to place the actuator on the side with the acute angle of intersection between the links, but this behavior can be overridden by changing the *CylSide* attribute. The other potential location

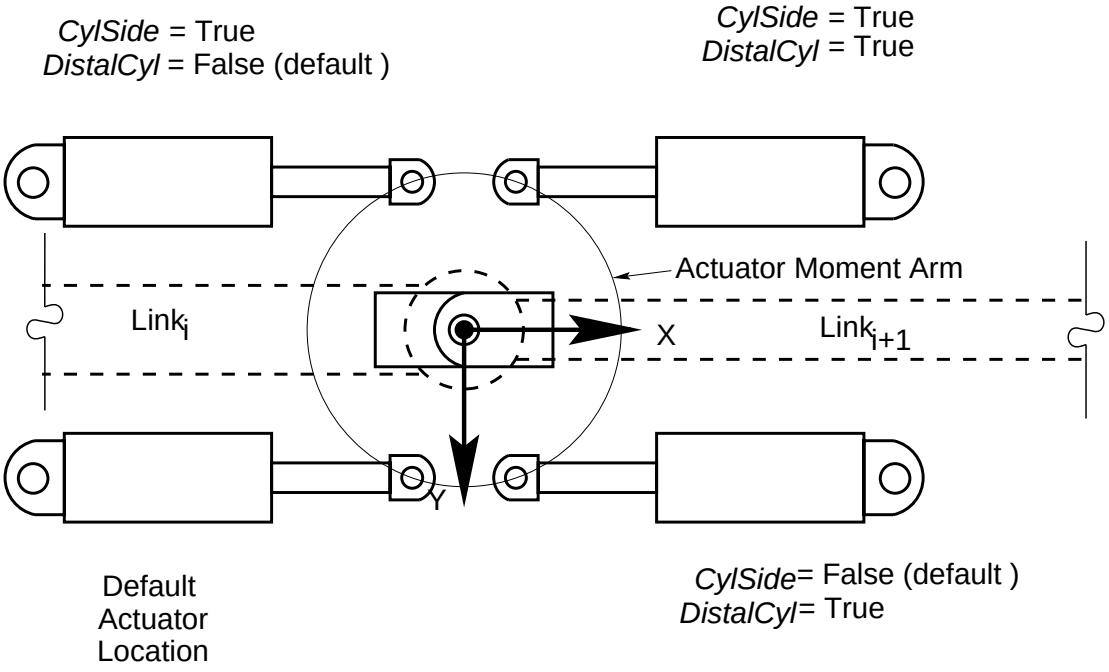

Figure 3.6. Possible actuator locations and the attribute states for the various locations.

is on the distal link. This location is specified using the *DistalCyl* attribute and is generally undesirable as it increases the inertia of the linkage. However, there are situations where the kinematics require this location, as when the cylinder length is shorter than the distance between the hinge axes. In this case, the heuristics will set this attribute, but it can also be specified by the designer whenever they feel that is desirable.

3.2.2 Structural Construction

Once the hinge points and actuator mount points have been determined, they provide the starting points for the structural synthesis. However, before the synthesis process can continue, it is desirable to define the beam types to use in the initial construction. This can be done by the default heuristic, that will select a round tube whose size depends upon the length of the link. However, the optimization process can be much faster when provided with an improved starting point.

The link structure is internally divided into four different sections: the base, the end, the structure and the bracing[Figure 3.7]. Each of these sections may have

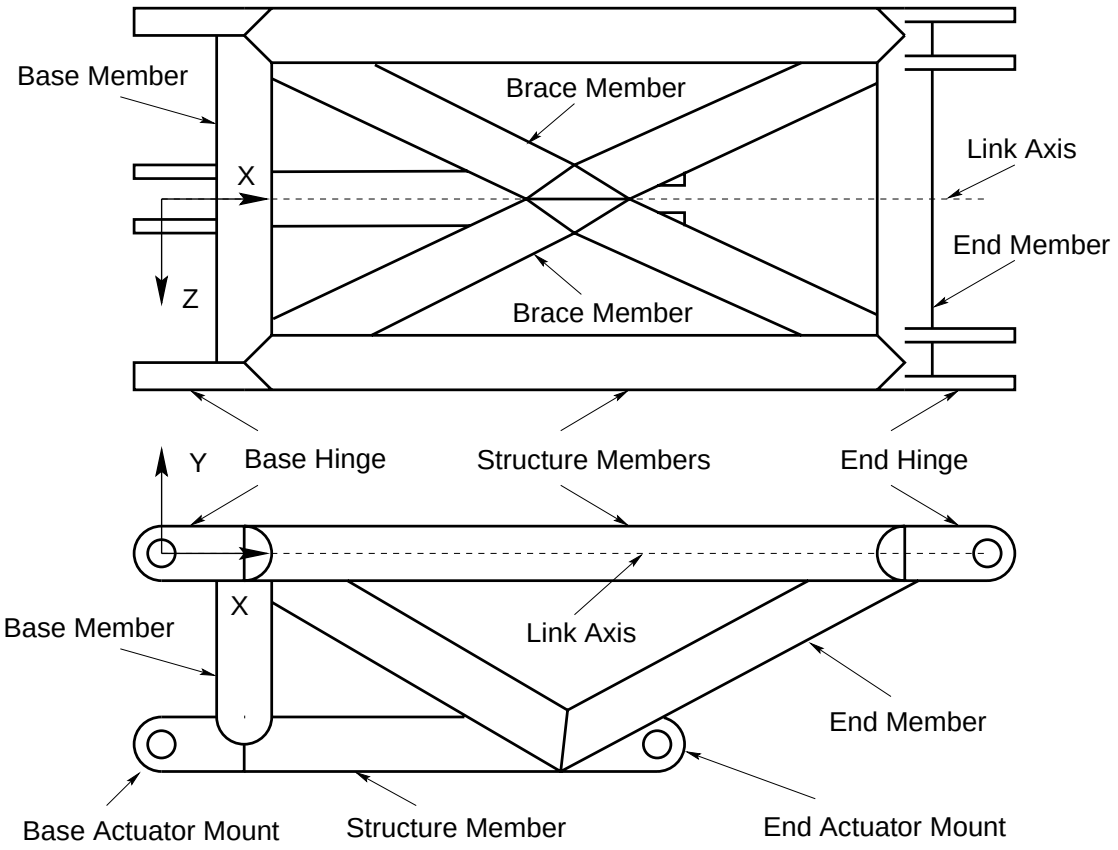

Figure 3.7. A simple link structure showing the relationship between the link and structure as well as the structural component identification.

its own beam type, that is determined by its attributes. The relevant attributes are *BaseBeam, EndBeam, StructBeam* and *BraceBeam*. It is not necessary for the designer to specify all of these. If any single one is specified, all of the others will be set to the same beam type. In the event that more than a single beam type is desired, the designer may specify any beam type for any section of the link structure they wish. The default behavior for unspecified attributes other than *StructBeam* is to be set to the same beam type as *StructBeam*. If *StructBeam* is not specified, it will be set to the first beam type it finds among the *BaseBeam, EndBeam* and *BraceBeam* attributes. If none of the attributes are set, the default round tube will be used.

Once the hinge and actuator points are determined, the hinge and actuator clevises are then constructed. These may be specified by the designer by setting

the *BaseClevis, EndClevis* and *CylClevis* attributes to the desired clevises. If they are not specified by the designer, the sizes of the clevises will be calculated from the size of the base beam or end beam, whichever is relevant. The default is to build the clevis the same width and height as the relevant beam. The default thickness of male clevises is about 1/3 the height, and the default thickness of female clevises is about 1/6 of the beam width. There is no check for interference between the clevises and the beams. These clevises are intended to be used as a rough approximation and need to be checked and detailed by the designer before fabrication. An example of the clevis geometry created by these heuristics is shown in Figure 3.8.

By default, the base hinge clevises are male, but this behavior can be modified with the boolean attribute *FemaleBaseHinge*. There is no corresponding attribute for the end hinges, as they are determined by having to fit with the next link. In the current implementation, the actuator mount clevises are always female, to fit with the male clevises on the actuators. This will have to be modified when support for additional actuator mounting styles is included in the system.

After the hinge and actuator clevises are constructed, the base and end substructures can be synthesized. This is done by simple triangulation of the ends of the hinge clevises and the relevant actuator mount clevis. This triangulation is performed by building three straight beams, one between the hinge clevises and the other two between a hinge clevis and the actuator mount clevis. For the base, the beam type used is specified with the *BaseBeam* attribute, and the triangulation of

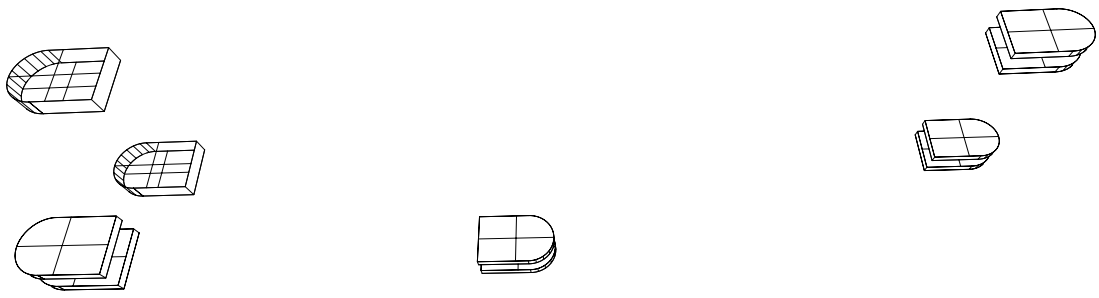

Figure 3.8. Synthesis of base and end hinge clevises as well as the actuator attachment clevises for the first link structure of Figure 3.1

the end substructure uses the *EndBeam* beam type. Orientation of the cross sections may be specified by setting the *BaseBeamOrVec* and *EndBeamOrVec* attributes to the desired orientation vector. However, this vector affects all of the beams in the relevant section, which may produce undesirable results. If modification of the orientation of the cross section of a single beam is desired, it is recommended that the orientation vector of the particular beam be replaced. This will have to be performed every time the synthesis process is repeated but is easily accomplished and is usually unnecessary. An example of this triangulation is shown in Figure 3.9.

Once the base and end structures have been constructed, the synthesis of the connecting structure is performed. This is accomplished by connecting the ends of the three base beams with the ends of the three end beams using the least amount of material. Using the constraint that each end of each base beam must connect to a unique end of an end beam, the minimum total distance is found. Once these point pairs are determined straight beams are constructed between them using the *StructBeam* beam type. As with the base and end sections, the orientation of the beam cross sections may be modified using the *StructBeamOrVec* attribute with the same restrictions and recommendations. In the event that there is no end structure, which occurs at the most distal links in the structure, the end point of the link axis curve is used and the structural members are constructed from this point to the base points. However, it is recommended in most circumstances that the single beam structural algorithms be used, as it is less expensive to manufacture

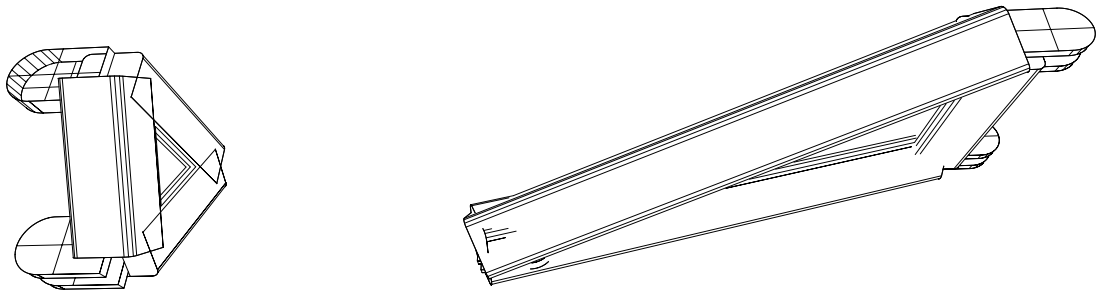

Figure 3.9. Synthesis of the base and end clevis connections for the first link structure of Figure 3.1

and usually strong enough for the last link of a chain. A simple example of the main structural beam generation heuristic is shown in Figure 3.10.

Examination of the structures resulting from this process reveal that they are constructed of quadrilaterals, an inherently unstable structure. In some cases the frame may support the specified loads without any additional members, but in most cases additional members are required to triangulate these quadrilaterals. The default heuristic for this is to use the same minimum distance constraint used in the structural synthesis but disallow the point pairs previously used. This produces reasonable triangulation in many cases but may require modification for cases when the structure is very short or very long. As with the previous heuristics, sample geometry is shown in Figure 3.11.

The short case is defined as when one of the structural members becomes sufficiently short as to not require triangulation. In this case, the member may be

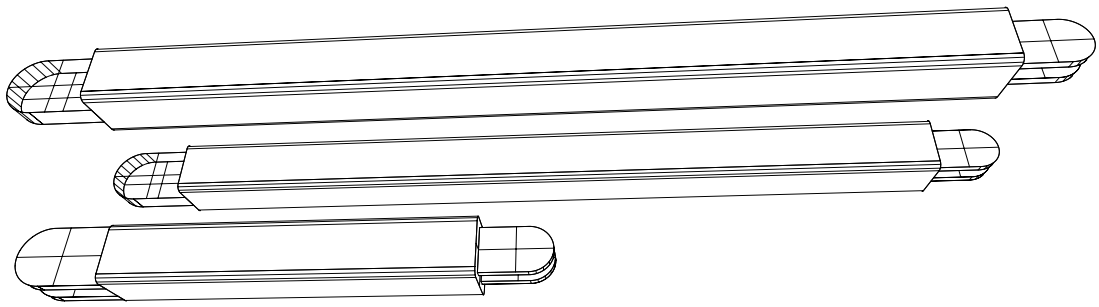

Figure 3.10. Synthesis of principal structural members for the first link structure of Figure 3.1

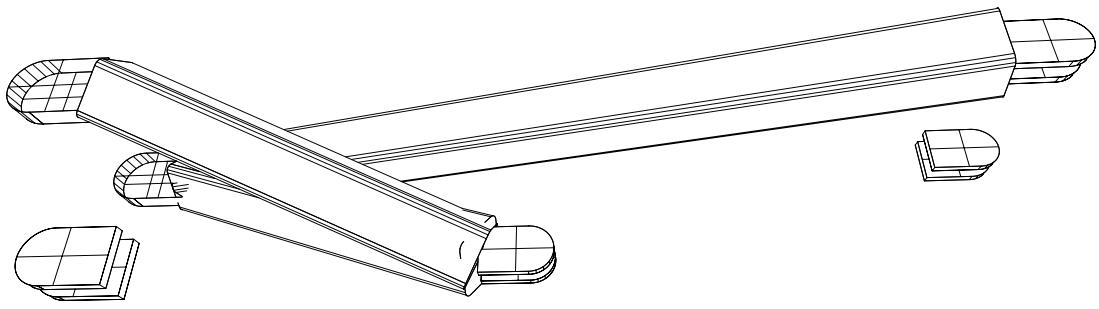

Figure 3.11. Synthesis of bracing members for the first link structure of Figure 3.1

stiffened with local gussets, or may require no stiffening at all. The point at which bracing is discontinued is determined by the *BraceFact* attribute, that defaults to a value of three times the height of the *StructBeam* cross section. As with all of the structural synthesis attributes, this can be modified by the designer to achieve the desired structure. A complete link structure produced by the structural synthesizer is shown in Figure 3.12.

The long case arises when the structural members become sufficiently long that buckling becomes a possible failure mode. When this occurs, rather than running the braces in a single straight line from the base to the end, the braces are run to intermediate locations on the structural beams and additional members are added as required. This stiffens the structural members by reducing their critical length, but requires significantly more members leading to higher fabrication costs. The fundamental heuristic used for this synthesis is to divide the longest structural member l_l into the desired number of subdivisions. The number of subdivisions n_{SD} is determined by the integer attribute *BraceSubDiv*, whose default value is one. If the length of the other two members is within δ_{SD} of the subdivided length of the longest member, they are subdivided into the same number of subdivisions as shown in Figure 3.13. If all of the structural members are equally subdivided, then the related subdivision points are connected with bracing members. After the interconnection has been accomplished, diagonal bracing is added to each section using the minimum distance heuristic to determine brace location. Otherwise, they are subdivided into fewer sections, usually $n_{SD} - 1$, but $n_{SD} - 2$ subdivisions are

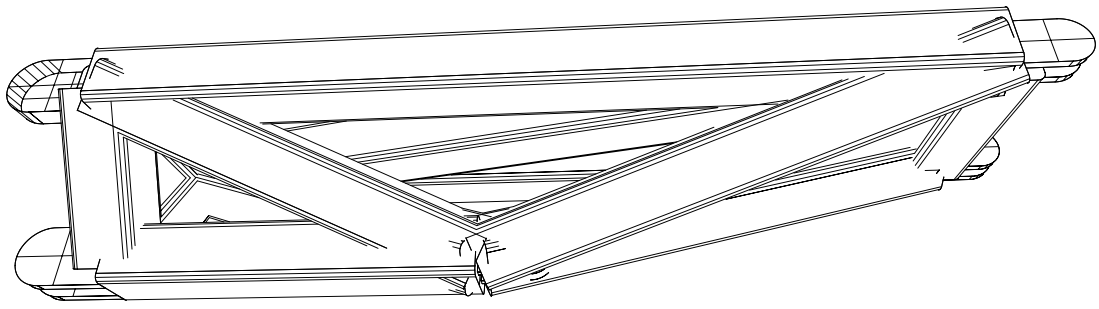

Figure 3.12. Completed synthesis of the first link structure of Figure 3.1

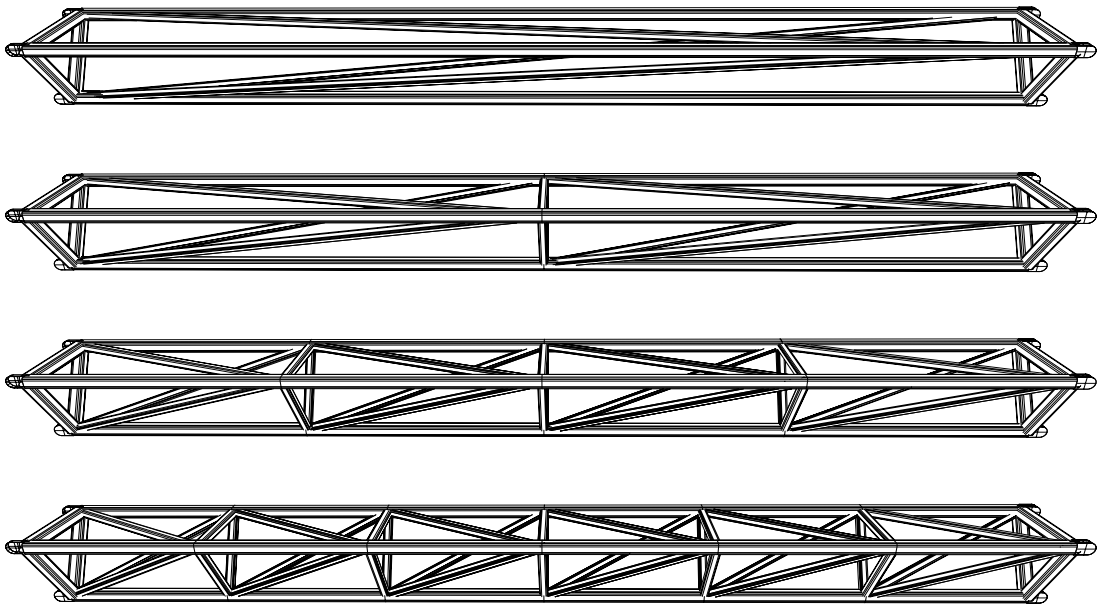

Figure 3.13. Link structure showing the normal bracing as well as subdivision bracing into two, four and six sections respectively, all using normal beam subdivision.

possible in some circumstances. This situation is usually a result of the desired actuator placement and either one or two of the structural members may be affected at each end of the link structure. Control of δ_{SD} is provided by the *BraceSubCorrFact* attribute n_{cf} that defaults to one. Using n_{cf} the computation of δ_{SD} is as follows:

$$\delta_{SD} = \text{ round } \left(\frac{l_l n_{cf}}{n_{SD}}\right)$$

In the case where there is not equal subdivision of the structural members, the structural synthesis heuristics are more complex and requires detailed examination of one or both ends of the structure. There are two possibilities for each end, either one or two of the structural members are more than δ_{SD} shorter than the longest member. In the first case, the standard interconnection is performed between the end of the short member and the related subdivision points of the longer members, after which an additional diagonal member is added between the two longer members. An example of this case with differing levels of subdivision is shown in Figure 3.14.

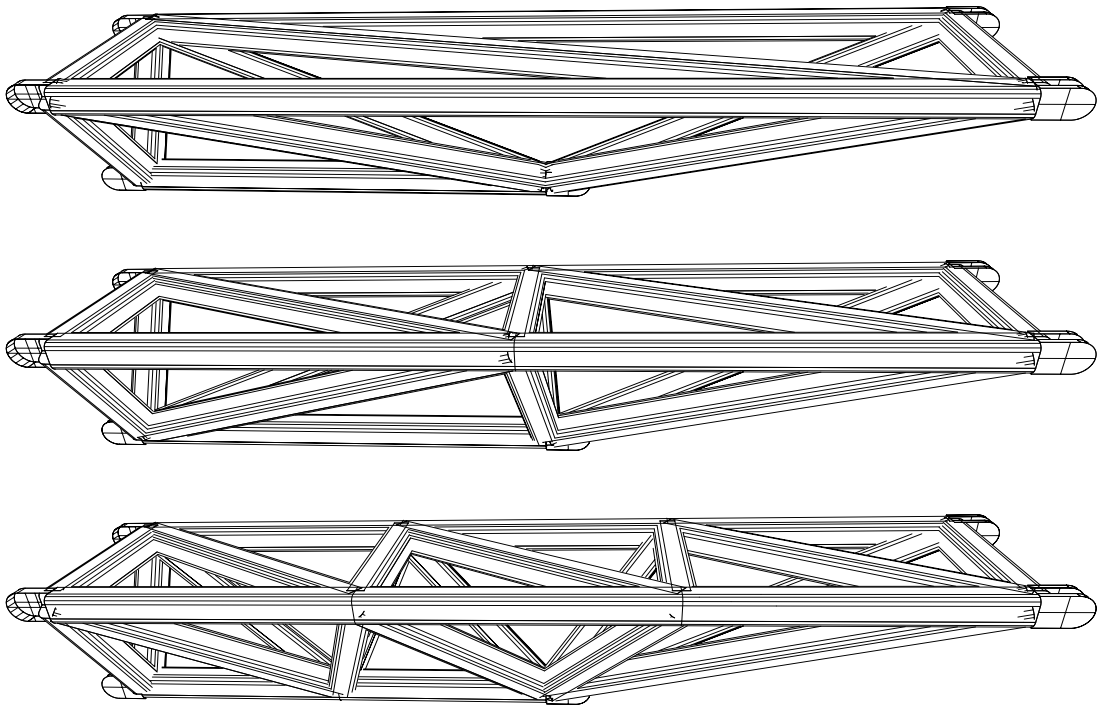

Figure 3.14. Link structure showing the normal bracing as well as subdivision bracing into two and three sections respectively, all with single short beam end condition.

If two members are δ_{SD} shorter than the longest member, they will already be connected by other beams so only two interconnection beams are needed. This usually results from unusual actuator placement; an example of this condition is shown in Figure 3.15. The additional members are added between the end points of the short members and the subdivision point. In this case, additional diagonal bracing is not required, as the necessary triangulation will be performed by the other beams forming the hinge.

3.3 Single Beam Cases

The term *single beam* used in this project should not be taken to mean the structure is composed of a single beam, although that may be the case. The intended implication is that the principal load bearing component is carried upon a single beam, but additional beams and bracing may be present to provide for hinge and actuator attachment points. These algorithms were implemented at the request

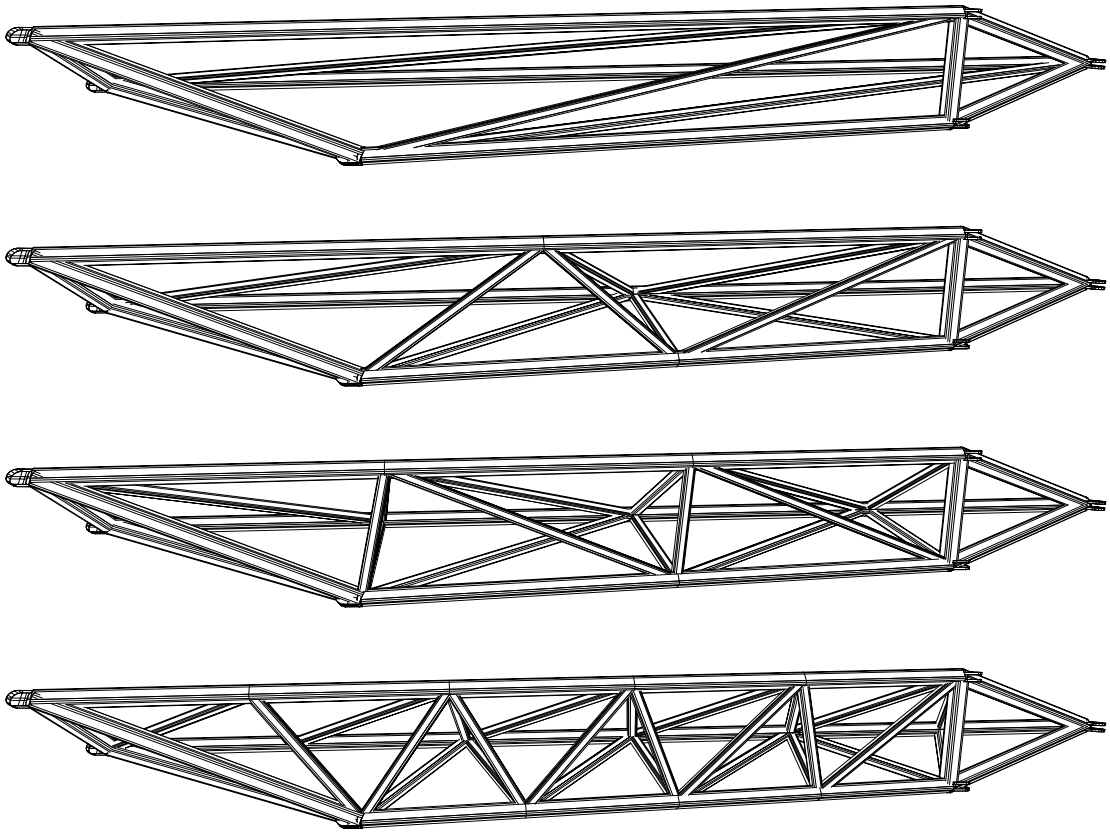

Figure 3.15. Link structure showing bracing for variable length beams as well as the single long beam end condition. From top to bottom, examples are subdivided into two, four and six sections. It should be noted that each of the structural beams in the bottom view is subdivided differently.

of design engineers, and often provide the simplest and lowest cost structure when the loads are low. To synthesize a single beam structure, the boolean attribute *SingleBeam* should be set. An example of a single beam structure is shown in Figure 3.16.

The synthesis of a single beam link structure proceeds in a similar fashion to the general algorithm. First, the joint geometry is calculated and then the structure is constructed from that geometry. The hinge construction is slightly different in that if the *HingeWidth* attribute is not set, the default value of the hinge width is the width of the beam. This allows the construction of a lap style hinge that is commonly used in this kind of application. If the *HingeWidth* attribute is set, then the hinge construction is the same as the general case, including the functionality

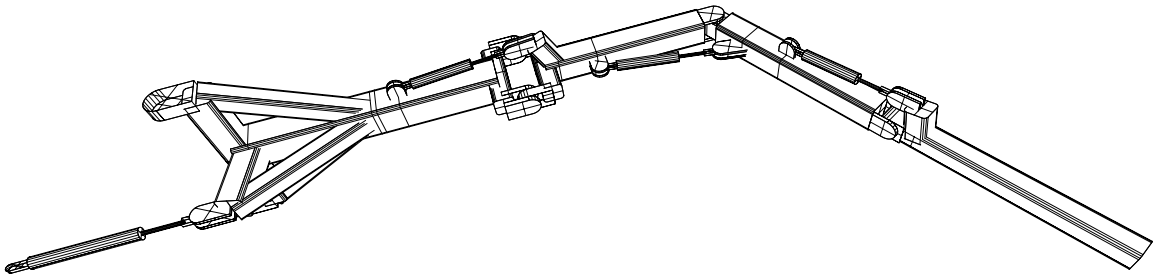

Figure 3.16. An example four link single beam structure showing most of the potential design permutations.

of the *HingeOffset* attribute.

Another difference in the default behavior of the synthesizer is in the placement of the fixed actuator mount. If the *CylOffsetAng* attribute is not set or is zero, then the synthesizer will try to place the actuator mount as close to the beam as possible, as shown in Figure 3.17. The default offset of the fixed actuator mount point from the center of the beam is $1.1(height/2 + actuatorOD/2)$, that leaves a small space for backing plates. If a different behavior is desired, the *CylOffsetAng* attribute may be used, as in the general case, to position the actuator wherever the designer wishes.

Once the joint geometry has been determined, the synthesis of the structure can commence. The fundamental backbone of the single beam structure is constructed from a single beam of type *StructBeam* that runs down the link axis. As the link

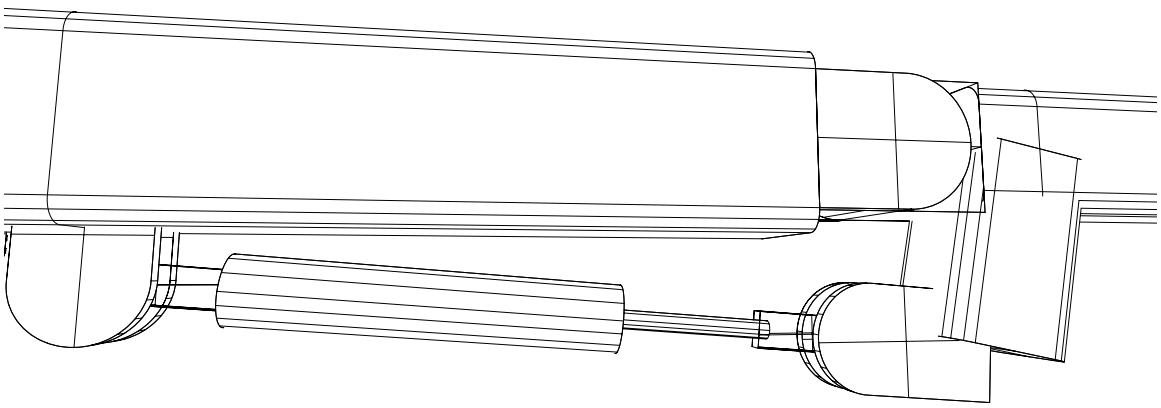

Figure 3.17. A default single beam joint.

axis is a NURBS curve and not constrained to a straight line, it is possible to construct single beam structures that twist or turn by defining the desired link axis. However, care should be taken to provide sufficient straight length of the link axis to allow for reasonable mounting of the actuator. Currently, the curved link axis is not considered by the space frame synthesizer and structures that require this behavior should be created using the single beam heuristics. The ability to create these types of structures can be very useful in certain situations.

If the *HingeWidth* attribute is set, then the hinges are joined together and to the structural beam with a single member of beam type *BaseBeam* . This same type of beam is used to connect to the actuator mount point if the mount point is further away from the beam than the beam height [Figure 3.18]. The fixed actuator mount point of any external actuators and end hinges will be attached to the structural beam with a member of beam type *EndBeam* if necessary. An example of this type of synthesis is shown in Figure 3.19. Increasing the hinge width is often desirable to reduce the bearing loads due to off-axis moments.

It may also be desirable to provide bracing from either the hinges or the actuator mount points to the structural beam. This bracing is controlled by several different

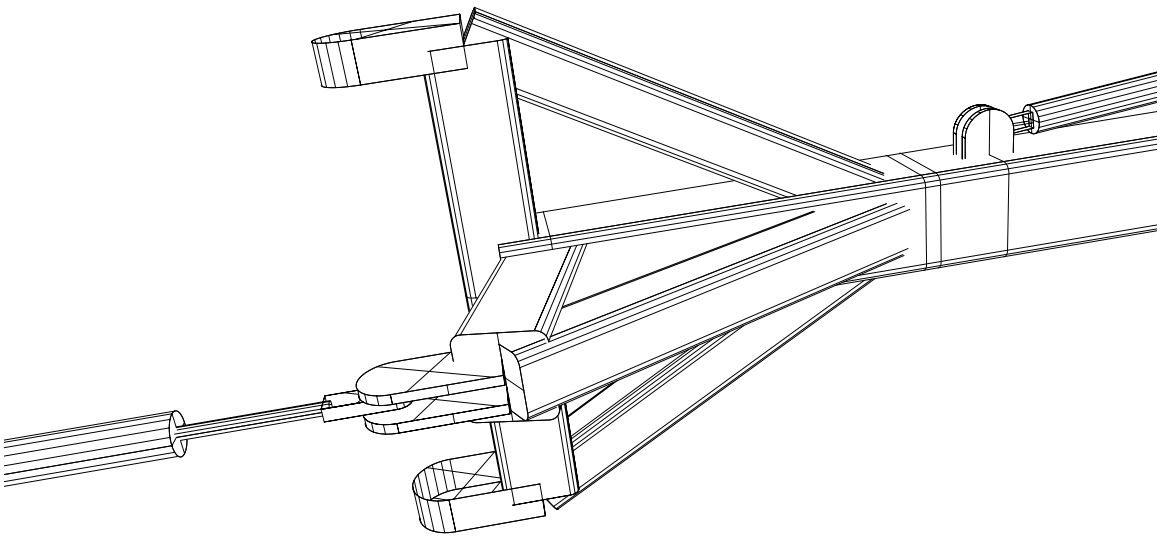

Figure 3.18. A complex single beam hinge and actuator mount with a wide hinge, distant actuator mount and related bracing.

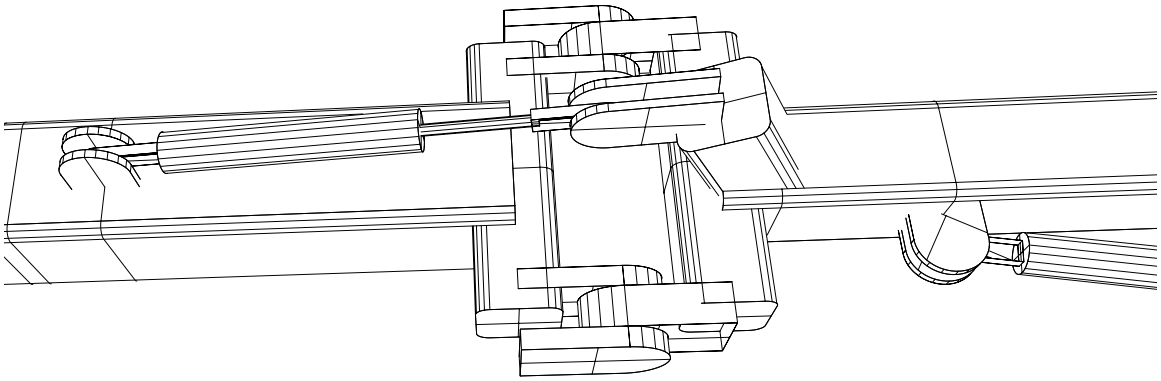

Figure 3.19. The end joint of a single beam structure with a large hinge width. It should be noted that both the hinge width and actuator moment arm were insufficient to require bracing.

parameters that define when and how the bracing is constructed. For the hinges the relevant attributes are *SingleBeamHingeBraceParm* and *SingleBeamBraceAngle.* The *SingleBeamHingeBraceParm* is a factor that controls the minimum distance the hinge clevises need to be from the structural beam before the braces are added. If the distance from the structural beam to the hinge is greater than the width of *BaseBeam* multiplied by *SingleBeamHingeBraceParm,* a brace is added from the hinge to the structural beam at an angle of *SingleBeamBraceAngle*. The default value for *SingleBeamHingeBraceParm* is two and the default value for *SingleBeamBraceAngle* is 60 degrees. The *SingleBeamClevBraceParm* attribute performs the same function for controlling the minimum distance necessary to brace the actuator mount and also defaults to a value of two. An example of this type of structural synthesis is shown in Figure 3.18.

3.4 Special Cases

The heuristics previously described provide the basic structural synthesis capability necessary for most common linkage configurations. However, there are cases when they prove inadequate. These difficult cases can be separated into two different types, the cases when the joint angle gets larger than around 60 degrees and the cases where the joint hinge axis gets within about 30 degrees of the link axis curve. Possible solutions to the first case can be to use the single beam algorithms

that handle these cases much better, or to play with the actuator placement options and actuator angle until a satisfactory structure is constructed. The single beam heuristics can also be useful for the second case as long as space allows and the range of motion is not too large. However, there are instances of the second case when a true rotary or twisting motion is desired, or when multiple joints are attached to a single structure. These cases are discussed in more detail below.

3.4.1 Rotary Joints

The case where a twisting motion is desired presents an unusual synthesis problem in that there are a number of different possible approaches. For light duty applications, it is not unusual to use a rotary vane actuator and attach the distal links directly to the actuator. In larger applications, the structural loads often exceed the allowable loads on the rotary vane actuator and additional external bearing support is required. For even larger applications, the torque requirements exceed the capabilities of available rotary vane actuators and a linear actuator operating on an eccentric crank or other high torque source is required.

These different actuator configurations and wide variation among potential solutions make structural synthesis for these types of joints a difficult task. In the first case, the size and mass of the actuator is critical to reasonable dynamic analysis. In the other cases, the details of how the joint load is transferred to the structure affects the structural analysis.

What has been implemented during this project is a set of simple heuristics that synthesizes a structure that connects to two planes normal to the joint rotational axis[Figure 3.20]. These planes are sized and spaced dependent upon the size of a cylinder that the designer specifies. The space is equal to the height of the cylinder and the triangle formed by the beam axes of the base and end beams is defined to include the circle described by the cylinder. This cylinder can be sized to describe a rotary vane actuator or many different types of support bearings, whatever the designer deems appropriate. This provides the designer with sufficient flexibility to perform the dynamic and structural analysis with reasonable structures but does not require the inclusion of heuristics for all of the potential cases. For this case,

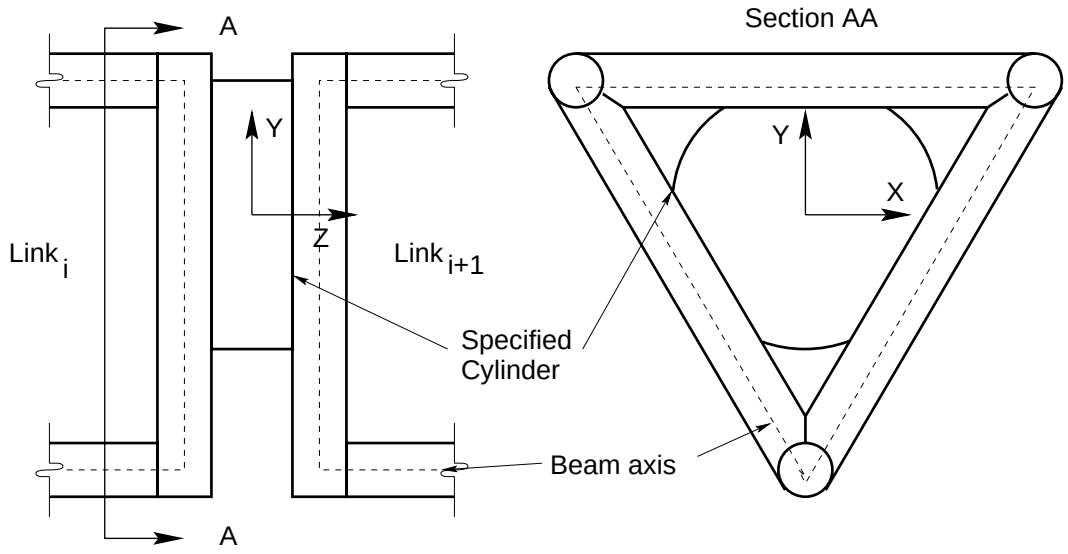

Figure 3.20. An example structure for a rotary joint showing the defining cylinder and related link structure.

the load decomposition for the structural analysis will be accomplished by simply dividing the loads by three and applying them directly to the joints between the base and end members. It is up to the designer to ensure that sufficient structure is included to render this load transfer assumption valid.

3.4.2 Multiple Joints

The second special case is when multiple joints are attached to a single structure. In this case, the structure synthesize heuristics need to be extended to construct a higher complexity structure. The range of multiple joint cases can be further divided into two subcategories by the magnitude of the loads on the external link structures. The principal heuristic implemented in the current system defines a *principal structure* that is expected to carry most of the loads. The remaining structures are referred to as *secondary structures* and are attached to the primary structure during the synthesis process. In the event the magnitude of these secondary loads is significantly less than the loads carried by the primary structure, these secondary link structures can be attached to the primary structure with relative ease. In the event that the loads from the secondary structures are of equal or greater magnitude than the primary structure, the structural synthesis

becomes more complex and heuristics for this case have not yet been implemented. The first step in the process is to construct a linkage with multiple attached links, an example of which is shown in Figure 3.21. Once the linkage has been constructed, then the structural synthesis for the light load case proceeds using the following heuristics:

1. Locate the distal joint closest to the end of the link axis. In the event that there is no joint close to the end of the link axis, then select the most distal joint.

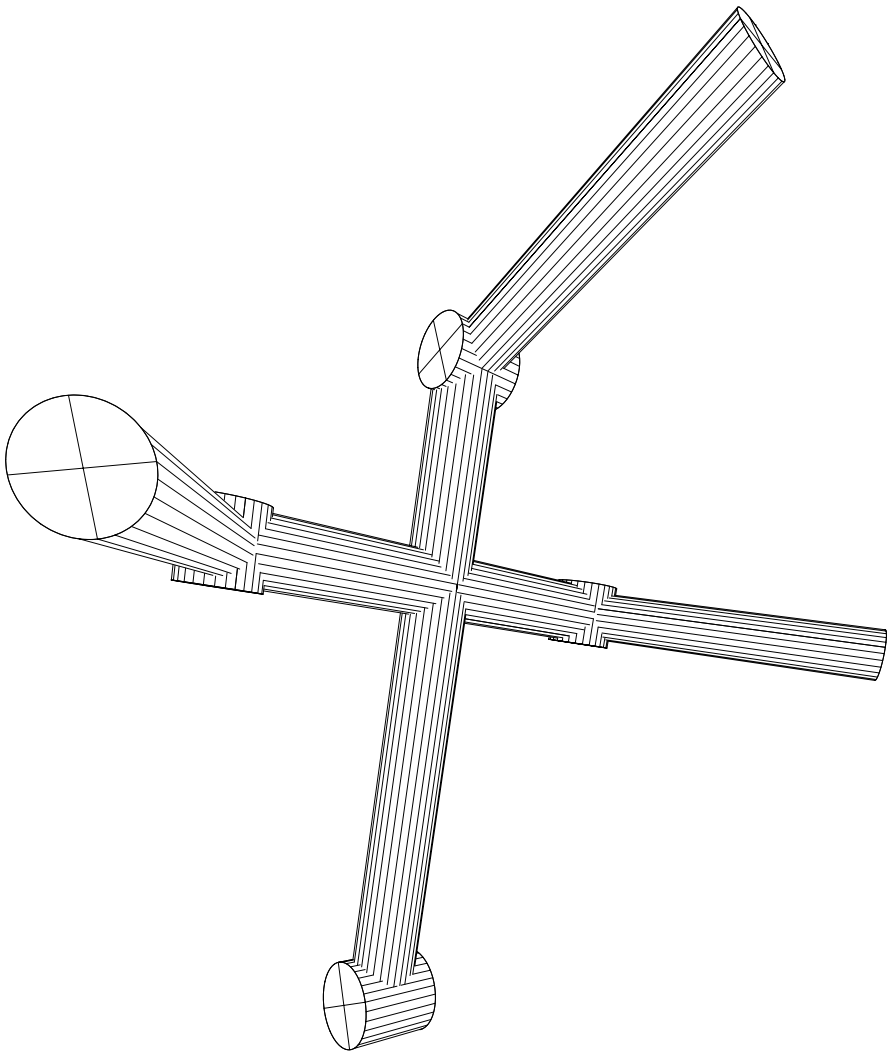

Figure 3.21. An example linkage with multiple attached links.

2. Construct the base and selected end substructures using the heuristics for the single distal joint cases.
3. Connect the base and end substructures again using the heuristics for the single distal joint cases.
4. Connect the additional end substructures to the link structure using the multiple joint heuristics.
5. Construct the substructures for the remaining joints using the same synthesis algorithm as the distal link. For example, if the distal link has the *SingleBeam* attribute, then the joint is constructed using that algorithm.

The principal difficulty with synthesis of these types of structures is attachment of the hinge and actuator mounts. Using the secondary structure load assumption, the approach taken is to attach the hinge points to the closest structural member. Then, the actuator mount point is triangulated to the next closest member. An example of these heuristics applied to the example linkage [Figure 3.21] is shown in Figure 3.22.

This approach and heuristics provide usable structures for most cases, but as the complexity of the structures and attachments grow it is possible to get marginal and/or unrealistic geometry. Heuristics have also been devised to allow for multiple attached structures for link structures synthesized using the single beam approach [Figure 3.23].

These heuristics work reasonably well for most cases fitting the load assumptions, but may not provide acceptable structural geometry for other cases. There is no structural reason why the heuristics could not be devised to fit these cases. However, to date this has not been done. If a designer is faced with such a circumstance it is relatively easy to manually add the necessary structural members and proceed with the structural analysis and optimization.

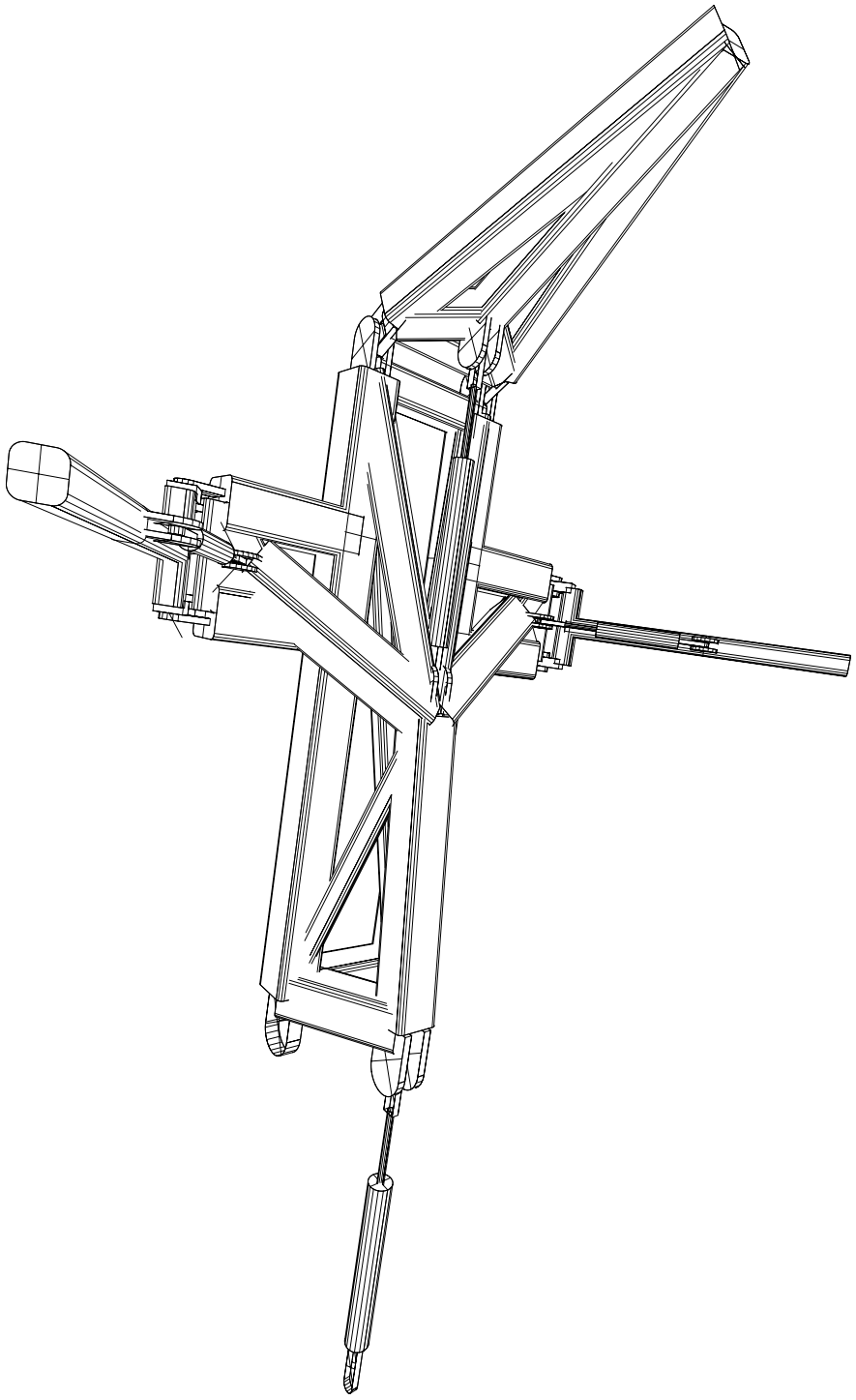

Figure 3.22. An example link structure with three-link structures attached to a single-link structure. In this case the upper link structure is the primary structure and the two side structures are secondary.

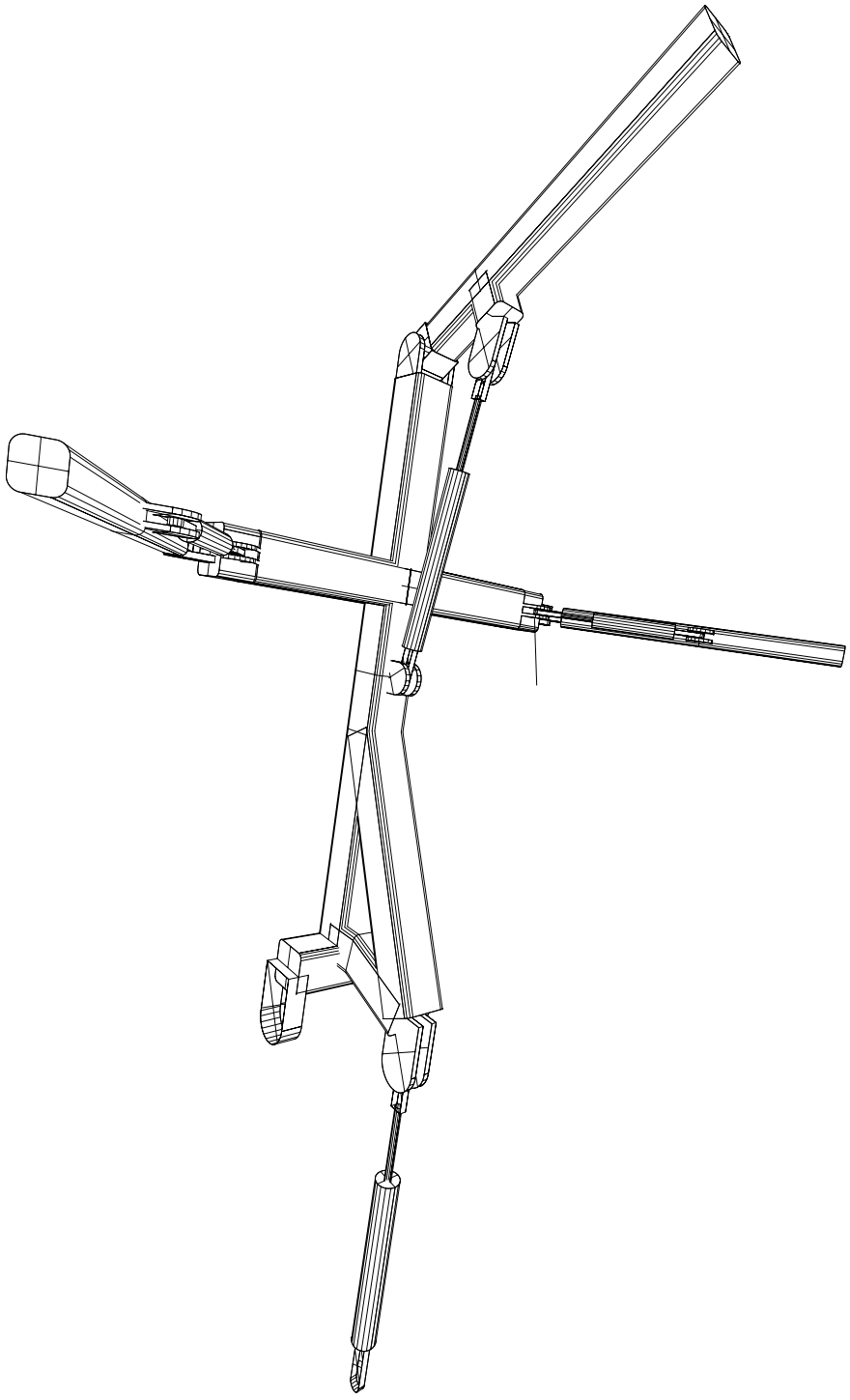

Figure 3.23. An example link structure with three-link structures attached to a single-link structure. All of the link structures were synthesized using the single beam approach.

CHAPTER 4

DYNAMIC ANALYSIS

The dynamic analysis of open kinematic chains is a well-understood field with several different possible approaches. These are documented in most of the common robotics texts [33, 47, 68]. In general, there are two complementary problems, the *forward dynamics* problem in which the joint torques/forces are known and the resulting joint trajectory, velocities and accelerations are calculated, and the *inverse dynamic* problem in which the joint kinematics are known (positions, velocities and accelerations) and the joint torques/forces are calculated. Many different approaches have been worked out in detail, including the Lagrange-Euler, Newton-Euler, d'Alembert's, and Kane's method. Although differing in approach and computational requirements, equivalent results are produced by all of these methods. Therefore, the "best" choice for a particular application is dependent upon the problem definition, data structures, data representation, and personal preference.

An additional problem facing structure designers is deriving useful loading information for structural analysis. In mechanisms of this type, the loads imposed upon the base hinge and actuator are dependent upon many varying parameters. These include link velocity and acceleration, loads imposed on the link by outer links, and link orientation with respect to gravity. Reconfiguration of the mechanism generally has a significant effect on the base loads. This leads to a combinatorial number of joint loadings as changes in configuration are propagated down the kinematic chain and makes defining reasonable *"worst case"* loads a difficult and time consuming task.

4.1 Kinematic Parameters

Before any dynamic analysis can be attempted, the problem must be sufficiently well defined. Although the geometric configuration of the linkage provides some of the necessary information, additional specification of the system performance is required before analysis can be performed. Typically, this involves specification of a position trajectory, that is then differentiated and evaluated at intervals for the necessary position, velocity and acceleration parameters. This approach is reasonable when the position trajectory is known with reasonable accuracy, but in many cases this is not a valid assumption. In many, if not most cases, the actual position trajectory is unknown and impossible to determine. Or a position trajectory may be known but it is not possible or reasonable to limit the system performance to the existing trajectory. In these cases, the designer is often given scalar values of joint velocity or acceleration, or time constraints (stop-to-stop) on how fast each joint must perform. From these vague constraints, the designer needs to define some reasonable velocity and acceleration constraints before further analysis can be performed. For this reason, it is desirable to pursue a different approach for the specification of joint velocity and acceleration constraints.

The approach developed for this project was to look at the joint range of motion (ROM), velocity and acceleration specifications for each link and from this information derive an envelope that defines the links maximum designed performance. This envelope can be represented by a *velocity trapezoid* (VT) that specifies the maximum designed joint velocity as a function of joint position [Figure 4.1].

The velocity trapezoid representation is based upon the assumption that, under normal use, the system should not be impacting the components, usually the actuator, that limits the links range of motion. Thus, the maximum allowable joint velocity at the limits of the link ROM must be zero. Somewhere inside the links ROM the velocity must reach a maximum, that is represented as the top section of the velocity trapezoid. Between the zero velocities at the ROM limits and the peak velocity, the allowable velocity varies linearly with position. From the slope of this linear-varying portion of the velocity trapezoid, accelerations can

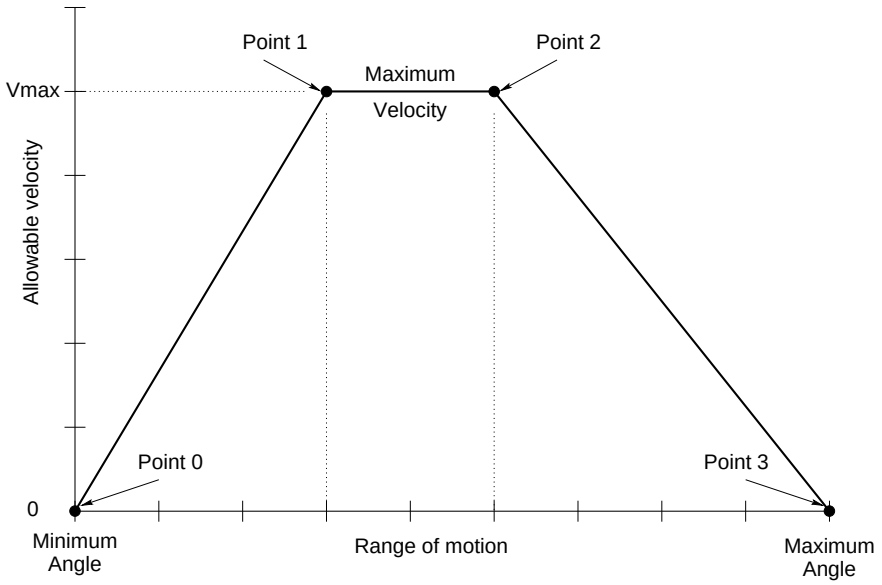

Figure 4.1. Velocity trapezoid showing parameters and defining points.

be computed, that are assumed to be the maximum designed accelerations the system will experience. Due to uncertainties in final structural loading, structural mass, actual actuator performance and corresponding compensations in the design, it is probable that the actual system performance will exceed that described by the velocity trapezoid. This should be compensated for by the designer, either by increasing the safety factors used for structural analysis, or by providing external limits on the system performance, such as acceleration and velocity limits in a feedback control system. An additional assumption, fundamental to the velocity trapezoid, is that the magnitude of both the velocity and accelerations are independent of direction. This is not a bad assumption when using fluid actuators but does increase the uncertainty of the dynamic analysis. This direction independence could be eliminated by creating a second velocity trapezoid with negative velocity values and different accelerations, but it was felt that this additional complexity would be of little value, as the overall uncertainty inherent in the velocity trapezoid representation generally exceeds that in the direction independence assumption.

One of the advantages of the velocity trapezoid representation is the ease with which maximum values for acceleration and velocity can be determined for any

position inside the ROM. This is accomplished as follows.

Given the following definitions:

$$\begin{aligned}
P &= \text{array of 4 points defining the trapezoid.} \\
P_i &= \text{the } i^{th} \text{ point. } 0 \leq i \leq 3 \\
P_{ix} &= X \text{ value of the } i^{th} \text{ point.} \\
P_{iy} &= Y \text{ value of the } i^{th} \text{ point.} \\
V_{max} &= \text{Maximum designed velocity.} \\
V_x &= \text{Maximum designed velocity at position } x. \\
A_{max} &= \text{Maximum designed acceleration.} \\
A_{min} &= \text{Minimum designed acceleration.}
\end{aligned}$$

Then it follows from the velocity trapezoid that:

$$V_{max} = P_{iy},\ 1 \leq i \leq 2 \tag{4.1}$$

$$A_{max} = \frac{P_{1x} - P_{0x}}{2V_{max}} \tag{4.2}$$

$$A_{min} = \frac{P_{2x} - P_{3x}}{2V_{max}} \tag{4.3}$$

The maximum and minimum designed accelerations are constant throughout the allowable ROM, but the allowable velocity varies. For a particular position x inside the link ROM the maximum designed link velocity can be found by:

$$V_x = \begin{cases} V_{max}\frac{x - P_{0x}}{P_{1x} - P_{0x}} & P_{0x} \leq x < P_{1x} \\ V_{max} & P_{1x} \leq x \leq P_{2x} \\ V_{max}\frac{P_{3x} - x}{P_{3x} - P_{2x}} & P_{2x} < x \leq P_{3x} \end{cases} \tag{4.4}$$

It should be noted that the velocity calculated from equation 4.4 is always positive, yet it is possible that the actual velocity is in the negative direction. The absolute value nature of the link velocity derived from a velocity trapezoid is compensated for during the dynamic analysis.

An additional advantage to the velocity trapezoid is in its relative ease of construction from the design specifications. If scalar values for acceleration are provided, the X location of the two center points may be determined by:

$$P_{1x} = P_{0x} + 2A_{max}/V_{max} \tag{4.5}$$

$$P_{2x} = P_{3x} + 2A_{min}/V_{max} \tag{4.6}$$

and the Y location of these points is defined by equation 4.1.

Often the joint acceleration is not specified, leaving the designer free to specify a percentage of the ROM for the joint to reach the desired velocity. Different percentages can be used for each side of the velocity trapezoid as long as the sum of the percentages used is less than 100%. For this case, the Y component of the velocity trapezoid center is again defined by equation 4.1 and the X components are found by:

$$P_{1x} = P_{0x} + p_{start}ROM \tag{4.7}$$

$$P_{2x} = P_{3x} - p_{stop}ROM \tag{4.8}$$

where p_{start} and p_{stop} are the start and stop percentages respectively and ROM is the joint range of motion. Once the trapezoid is defined, the accelerations are determined by equations 4.2 and 4.3.

It may be that the designer is given stop-to-stop times, with little or no restriction on either velocity and acceleration. For this case, the X components of the velocity trapezoid may be found using equations 4.7 and 4.8. Then the maximum velocity can be determined as follows:

$$V_{max} = \frac{2ROM - P_{2x} + P_{1x}}{\Delta t} \tag{4.9}$$

where Δt is the stop-to-stop time. Once the maximum velocity has been determined, the accelerations can be determined by equations 4.2 and 4.3.

In conclusion, the velocity trapezoid representation provides several advantages over the conventional position trajectory representation. These can be summarized as follows:

- Velocity trapezoids are much easier to construct and/or modify then position trajectories.
- Velocity trapezoids define reasonable limits on velocity and acceleration for all allowable positions.
- Several different fabrication methods from well-understood parameters are available for trapezoid construction.
- No curve differentiation and corresponding stability problems are necessary when using velocity trapezoids.

4.2 Recursive Newton-Euler Formulation

Given the recursive nature of the linkage representation, the recursive Newton-Euler Formulation was the logical approach for dynamic analysis. Not only is it one of the most computationally efficient dynamic algorithms, it also fits quite well with the recursive nature of the linkage representation used in this project. The recursive Newton-Euler formulation is based upon a *forward propagation* and a *backward propagation*. During the forward propagation, the kinematic terms (velocity and acceleration, both linear and angular) are calculated, starting from the base and working outward along the linkage, summing in the contribution from each link motion until reaching the end. After all the kinematic terms for each link have been determined, the joint kinetic terms (joint forces and torques) may be found. These are calculated during the backward propagation starting at the extreme link. For each link there can be three contributions to the kinetic terms. The first results from inertial loads due to acceleration of the link mass. The second results from coupling loads imposed by external links. The last is any external loads applied to the link. Any or all of these loads may be zero but the dynamic analysis must support all combinations.

The following derivation of the recursive Newton-Euler equations of motion is similar to the approach proposed by Luh et al.[86] and further modified by Fu et al.[47]. However the formulation was modified to integrate with the tools and data structures included in the DynaFrame system. Using the example configuration

shown in [Figure 4.2], the following variables can be defined:

ω_i = the angular velocity of link i with respect to the i^{th} coordinate system.

α_i = the angular acceleration of link i with respect to the i^{th} coordinate system.

${}^i\mathbf{v}_{i-1}$ = the linear velocity of coordinate system i with respect to the $i-1$ coordinate system.

${}^i\omega_{i-1}$ = the angular velocity of coordinate system i with respect to the $i-1$ coordinate system.

${}^i\mathbf{a}_{i-1}$ = the linear acceleration of coordinate system i with respect to the $i-1$ coordinate system.

${}^i\alpha_{i-1}$ = the angular acceleration of coordinate system i with respect to the $i-1$ coordinate system.

$\mathbf{f}_i$ = the force acting upon the origin of coordinate system i.

$\mathbf{m}_i$ = the moment acting upon the origin of coordinate system i.

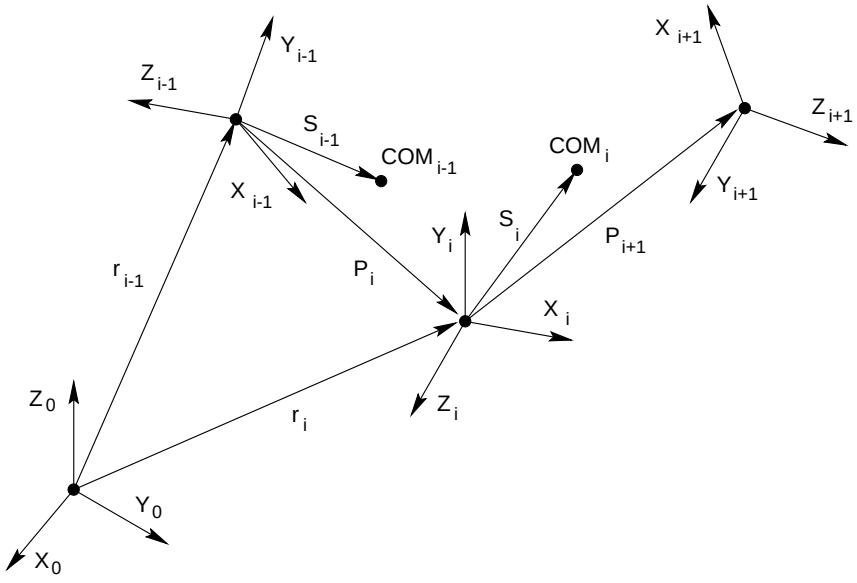

Figure 4.2. Definition of some of the kinematic variables used in dynamic analysis. In this case the links are P_i and P_{i+1}

m_i = the mass of link i.

$\mathbf{I}_i$ = the inertia tensor of link i calculated about the hinge axis.

${}^{i}\mathbf{T}_{i-1}$ = the mapping transformation between coordinate system i and the $i-1$ coordinate system.

${}^{i}\mathbf{T}_{i-1}^{-1}$ = the inverse mapping transformation between coordinate system i and the $i-1$ coordinate system.

Using the above variables, the forward propagation between coordinate systems proceeds as follows:

$$ {}^{i}\omega_0 = {}^{i}\mathbf{T}_{i-1}({}^{i-1}\omega_0 + \omega_{i-1}\mathbf{u}_z) \tag{4.10} $$

$$ {}^{i}\alpha_0 = {}^{i}\mathbf{T}_{i-1}(\alpha_{i-1} + {}^{i-1}\,\omega_0 \times \omega_{i-1}\mathbf{u}_z + \alpha_{i-1}\mathbf{u}_z) \tag{4.11} $$

$$ {}^{i}\mathbf{v}_0 = {}^{i}\mathbf{T}_{i-1}({}^{i-1}\mathbf{v}_0 + \omega_{i-1}\mathbf{u}_z \times \mathbf{p}_i) \tag{4.12} $$

$$ {}^{i}\mathbf{a}_0 = {}^{i}\mathbf{T}_{i-1}({}^{i-1}\mathbf{a}_0 + \alpha_{i-1}\mathbf{u}_z \times \mathbf{p}_i + {}^{i-1}\,\omega_0 \times ({}^{i-1}\omega_0 \times \mathbf{p}_i)) \tag{4.13} $$

It should be noted that using the above representation, link i sees the kinematic parameters of link $i-1$ plus any ω_i or α_i terms. Typically, the initial conditions are $\omega_0 = \alpha_0 = \mathbf{v}_0 = (0,0,0)$ and $\mathbf{a}_0 = \mathbf{g}$, the gravity vector in world coordinates. The linear velocity term ${}^{i}\mathbf{v}_0$ (Equation 4.12) is not needed for the dynamic load computation but is included for completeness. The backward propagation of kinetic loads starts at the extreme links and propagates back down the chain as follows:

$$ \mathbf{f}_i = m_i({}^{i}\mathbf{a}_0 + {}^{i}\,\alpha_0 \times \mathbf{s}_i + {}^{i}\,\omega_0 \times ({}^{i}\omega_0 \times \mathbf{s}_i)) + \sum {}^{i+1}\mathbf{T}_i^{-1}\mathbf{f}_{i+1} \tag{4.14} $$

$$ \begin{aligned} \mathbf{m}_i = {} & \mathbf{I}_i^i\alpha_0 + {}^{i}\,\omega_0 \times (\mathbf{I}_i({}^{i}\omega_0)) + m_i({}^{i}\mathbf{a}_0 \times \mathbf{s}_i) \\ & + \sum {}^{i+1}\mathbf{T}_i^{-1}\mathbf{f}_{i+1} \times \mathbf{p}_{i+1} + \sum {}^{i+1}\mathbf{T}_i^{-1}\mathbf{m}_{i+1} \end{aligned} \tag{4.15} $$

External loads applied to the link are treated as another external linkage and included in the summation terms in equations 4.14 and 4.15 while additional mass and/or inertia may be included in the link inertia tensor. In equations 4.14 and 4.15, the distinct contributions to the kinematic computations can easily be seen. The external contributions, either links and/or loads are included by the summation

terms, and the local effects due to the mass and acceleration of the local link are the remaining terms. The computation of these kinetic terms was greatly assisted by the use of the load objects implemented for this project, which incorporate all of the necessary algorithms to perform the load transformations included in the summations.

The performance of the recursive Newton-Euler formulation proved satisfactory for this project, although improvement in the computational speed is always desirable. However the desire to include closed kinematic chains in future implementations and the related increase in computational complexity will require at the least a significant modification to the algorithm. A potentially better approach might be to integrate one of the existing commercial dynamics packages into the system, as they have a much wider configuration range. With the ability to compute the dynamic loads, the process of determination of the worst case loadings can begin.

4.3 Calculation of Load Cases

The determination of the "worst" loading that may be imposed upon a structure is, at best, an iterative problem and may be indeterminate. Regardless of the assumptions used, it is often possible to find a kinematic configuration that will produce higher stresses in some structural component. Despite this difficulty, it is still necessary for designers and analysts to at least determine representative loads that can be expected to be reasonably close to the "worst" loading the structure will see in service. The approach generally used is to decide upon general assumptions that make the problem tractable. It still remains up to the designer to ensure that these assumptions are not violated in actual service. For the DynaFrame system, the following load assumptions were used:

1. The structure is not subject to impact loads. The calculation of impact loads requires knowledge about the actuation system not included in the model. If the structure and actuator are expected to experience impact loads, then additional analysis and adjustment of safety factors is required.

2. Peak working loads are experienced at maximum designed acceleration. It is assumed that accelerations are limited to the designed values, even though the system may be capable of higher performance.
3. The maximum force the actuator and structure experience is at the maximum operating pressure. In practice, meeting this assumption may require other protection systems.

Before the "worst case" loads the structure will experience can be determined, a definition of "worst case" is needed. There is general agreement that the worst case load is the load that produces the highest stresses or is closest to initiating buckling in the structure. Unfortunately, this definition is of little use as it implies an iterative solution coupled with the dynamic analysis, which is impractical with current technology. Using the largest magnitude of the joint forces or torques is insufficient, as depending upon the structure and joint coupling, other loads might create significantly higher loads in individual structural members. Furthermore, buckling analysis will require the highest compressive loads, that may be different than the highest magnitude loads. Because of these problems, it was decided to try and bound the worst case loads by defining 16 load cases as follows:

- The maximum force in the direction of each axis (3).
- The minimum force in the direction of each axis (3).
- The maximum moment about each axis (3).
- The minimum moment about each axis (3).
- The force with the maximum magnitude.
- The moment with the maximum magnitude.
- The force with the maximum magnitude in the XY plane.
- The moment with the maximum magnitude in the XY plane.

The maximum and minimum moments and forces acting upon the local Z axis are understandable, as these directly translate into actuator torque and axial hinge load. However, the choice of the maximum and minimum moments and forces in the remaining (XY) directions is less clear, and represents a compromise common

to design. It is possible to exclude a load case that results in higher structural stresses or is closer to the buckling load due to the arbitrary orientation of the XY axes. The only apparent alternative would be to try to bound these *off-axis* loads by including additional cases with orientations between the local axes, a solution that still does not guarantee the actual worst case load. The decision was made to use ten off-axis load cases and inform the designer of the limitation, thus allowing them the opportunity to orient the local coordinate system in a manner that they feel is most likely to reflect the worst case dynamic loads. To provide a means of keeping track of these loads and related kinematic parameters, the *link_loads* object was implemented. This object contains a *dynamic_load_obj* and a linkage configuration specification for each of the 16 worst case load cases.

Determination of the "worst case" loadings for a linkage requires performing a dynamic analysis in the configuration that produces the "worst" load with the velocity, acceleration, and external loadings (both from external links as well as other sources). However, as the linkage configuration, velocity, acceleration and external loadings that will produce these loads are not known, a search of the linkage workspace coupled with dynamic analysis is in order.

The approach used in the DynaFrame system is to subdivide the ROM of each link into a number of different discrete configurations and then perform a dynamic analysis on each of the configurations. At each configuration, four different sets of acceleration and velocity vectors are evaluated, corresponding to two different directions of motion, each with two possible directions of acceleration. Thus, the number of dynamic analyses performed can be determined as follows:

$$An = (Ev * N_e)^{N_l} \tag{4.16}$$

where:

An = total number of dynamic analyses

Ev = number of different evaluations at each linkage configuration, typically 4

N_e = number of different evaluations points in the ROM of each link

N_l = number of links in the linkage

Examination of the "worst case" loads from many different linkages suggests that the worst loads generally occur either when the linkage is at the limits or in the center of the link ROM. The algorithm used in the DynaFrame system to determine the linkage configurations for evaluation will select the center of the ROM if $Ne = 1$ and the endpoints of the ROM if $Ne = 2$. Therefore, a minimum of $Ne = 3$ is recommended, with $Ne \geq 5$ being desirable. A search algorithm to improve the accuracy and reduce computational time is both feasible and desirable, and implementation of such a search algorithm is planned.

Because of the time required to perform the large number of dynamic analyses, the ability to extract and save the kinematic configurations producing the worst loads was incorporated into the DynaFrame system. This allowed for recomputation of the dynamic loads through the structural optimization process without repeating the search process, providing a significant reduction in computational time. As a majority of the design iterations are due to changes in link structure or actuator sizing, this reduction in computational effort can be very helpful. The use of the search results assumes that the kinematic parameters remain unchanged and the change in mass distribution in each link is relatively small. These are reasonable assumptions that are valid for most circumstances. However, it is recommended that the search process be repeated once the structural optimization process has been completed, thus reducing the potential error.

4.4 Actuator Sizing

After the worst case dynamic loads have been determined, it is possible to check the actuator size against the torque requirements determined by the dynamic analysis. The required actuator force F_a is determined as follows:

$$F_a = \frac{M_z}{r_e} \tag{4.17}$$

where:

F_a = the necessary actuator force.

$$
\begin{aligned}
M_z &= \text{Z component of the maximum dynamic moment about the hinge.} \\
r_e &= r\cos(\nu) \qquad (4.18) \\
r &= \text{the maximum moment arm calculated from Equation: 3.10.} \\
\nu &= \text{the current angle of the link measured from the perpendicular} \\
&\quad \text{moment arm from Figure: 3.5.}
\end{aligned}
$$

This must be repeated using M_z from both the maximum and minimum moment load cases. Due to the difference in force between the extension and retraction of linear fluid actuators, either direction may be the active constraint. It is possible using the *CylOffsetAng* parameter to skew the ROM such that r_e is maximized $r_e = r$ at the location of the active constraint. However, when doing this, care must be taken not to introduce singularities in the joint actuator kinematics. Thus in all cases:

$$CylOffsetAng < \frac{\pi - ROM}{2} \qquad (4.19)$$

Methods have been provided for calculation of F_a and the required fluid cylinder diameter given the worst case dynamic loads. It is highly recommended that the actuator be properly sized before performing a structural analysis. Although it is possible to automate the actuator sizing process, experience has shown that actuators are often selected based upon other criteria than required force. In particular, linear actuators are often stability limited and a reasonable initial approach is to size the actuator stroke at the recommended stability limit. Once the smallest allowable actuator bore size has been determined during the structural optimization process, the actuator stroke can be reduced to provide the desired joint performance or meet other geometric constraints. Currently, the structural synthesizer and actuator catalog only support pin-pin mounting conditions, although the necessary extensions to support trunnion mounted end conditions could easily be added.

With a reasonable estimate of the worst case dynamic loading conditions computed and the appropriate actuator sizes required to produce the necessary joint torques included in the structural model, the structural analysis can proceed.

CHAPTER 5

STRUCTURAL ANALYSIS

Engineering has been described as the art of doing for a dime what anybody else could do for a dollar. The purpose of performing structural analysis is to provide information regarding the anticipated structural performance without having to actually construct and measure the structure. Once the structural elements have been synthesized, structural inertias are known, the dynamic analysis has been completed, and the worst case loads have been determined, the construction of abstract computational models for stress and buckling analysis can be performed. The approach selected to perform the structural analyses for this project is known as finite element analysis (FEA).

5.1 FEA Analysis

FEA is a methodology during which a mathematical model is used to estimate the behavior of the structure under defined boundary conditions. This model is constructed by defining a number of points known as *nodes* connected by *elements.* These elements may be 1-D (e.g., beam, truss), 2-D (e.g., shells), or 3-D (e.g., solid) and a variety of possible configurations are available for each type. The behavior of each element is predefined by a set of equations, thus the solution turns into a problem of solving a large set of simultaneous equations. The time and computational effort required to solve this set of equations is primarily dependent upon the number of degrees of freedom, which is roughly proportional to the number of nodes. For purposes of the DynaFrame system, beam elements were selected for several reasons. First, they provide a simple representation with a low number of nodes, thus reducing the time required for determining the solution. Second, they fit in well with the existing data types used to represent link structures. Last, the

areas in which the accuracy of the beam element falters are the joints, areas in which the CAD model already does not accurately represent the geometry required for fabrication. The FEA problem was further simplified by restricting the analysis type to linear static, which assumes that the structural stresses stay in the elastic region and that the structure is in static equilibrium. The elastic assumption is a common engineering approach and should be valid for any of the design stress levels. However, the static assumption is not valid, as the structures are moving and the worst case loads describe an instantaneous state of dynamic equilibrium. There are several potential approaches to working around this difficulty. The approaches used in this project are described in detail in Section 5.1.2. Within the limits of these assumptions, FEA is a powerful tool for determination of the stress levels and stability. Therefore, the ability to automatically generate FEA models from the link structures was integrated into the DynaFrame system. Construction of the FEA models requires the following steps for each individual link structure:

- Generation of an element mesh of the structure. This contains the nodes and connective elements to construct a mathematical model of the elastic behavior of the structure.
- Assignment of the correct material and other properties to the element mesh.
- Application of reasonable displacement boundary conditions.
- Decomposition of the worst case dynamic loads into a reasonable set of load cases.
- Formatting of the model for a particular FEA package.

Once these steps are completed, the analysis may be performed and the results input and used in the structural optimization process.

In the DynaFrame system, nodes are assigned and the element mesh generated directly from the beam objects used to construct the structure. Currently the system uses single dimensional or beam elements, although some shell elements are supported in the FEA module. The system creates a FEA model using internal representations for the necessary FEA components: nodes, elements, displacement constraints and load cases.

A *NodeObj* was defined as a new object type in the DynaFrame system. This object contains the spatial location of the node and a unique identification number. A formal definition of the node object is shown below:

$$\mathbf{Fn}_i = \begin{cases} n_i & \text{a unique integer for the node} \\ \mathbf{p}_i & \text{the point describing the node location} \end{cases}$$

A new *BeamElementObj* was also defined. Beam elements are commonly defined as a straight line between two nodes. This fails to describe the cross section orientation, so it is a common practice to include an additional node for this purpose. A *BeamElementObj* is formally defined as follows:

$$\mathbf{Fe}_i = \begin{cases} n_i & \text{a unique integer for the element} \\ \mathbf{p}_i & \text{a list of three nodes, the endpoints and an orientation} \\ Em_i & \text{an element material identifier} \\ Ec_i & \text{an element cross section identifier} \end{cases}$$

Also included in the Dynaframe system were definitions of additional types of FEA elements including truss, triangular and quadrilateral shells.

In addition, the FEA model generation requires representations of the boundary conditions. For purposes of providing load information, the existing *LoadObj* proved quite useful. However, a means of providing displacement constraints was also needed. For the purpose of constraining the displacement and rotation of a point, the *NodeDispObj* object was defined as follows:

$$\mathbf{Fd}_i = \begin{cases} n_i & \text{the node identifier} \\ d_x & \text{the displacement in the X direction} \\ d_y & \text{the displacement in the Y direction} \\ d_z & \text{the displacement in the Z direction} \\ r_x & \text{the rotation about the X axis} \\ r_y & \text{the rotation about the Y axis} \\ r_z & \text{the rotation about the Z axis} \end{cases}$$

An empty slot implies no constraint, and any numerical value was assumed to be an imposed displacement or rotation of the given magnitude, typically zero. This object definition will be modified to use vector descriptions for displacement and rotation in the next code revision.

Last, some structure for the FEA model was desirable. This was accomplished by defining an *FEStateObj* that provided convenient slots for storage of the various components of the FEA model. It was defined as follows:

$$\mathbf{Fs}_i = \begin{cases} n & \text{a list of nodes} \\ e & \text{a list of elements} \\ eg & \text{a list of element types} \\ m & \text{a list of materials} \\ cs & \text{a list of cross sections} \\ lc & \text{a list of load cases} \\ bc & \text{a list of node displacement objects} \end{cases}$$

These objects provide the foundation for the FEA module and with them, the process of automatic model generation is much simpler.

5.1.1 Automatic Mesh Generation

Using the above objects, the generation of a FEA mesh of the link structures is relatively simple task.[Figure 5.1, Figure 5.2]. First, the beam material and cross section are checked against the materials and cross sections already defined in the FEStateObj which serves as a wrapper for the FEA database. If either the material and/or cross section are not already in the FEStateObj then the relevant objects are added.

The nodes are generated from the beam axes, which are NURBS curves. The first step is to extract the control polygon, that is a list of points roughly defining the curve. The length of this list is compared against the desired number of nodes for each beam. If additional nodes are needed, the curve is first refined, a process that increases the number of control points in the areas of high curvature[Figure 5.3].

If there are no areas of high curvature, then additional points are added by subdivision. Once a sufficient number of points have been added to the control polygon, each point is checked against the existing node database. If the point is not within the desired tolerance of an existing node, a new node is created at that point. For the purposes of meshing, all clevises are treated as male and a center axis is defined from the pivot point to the center of the clevis base. Material and cross section properties are defined using the same methodology as beams.

Once the mesh is generated, displacement boundary conditions are applied to the base hinge point(s). Typically, one hinge point is completely constrained in both

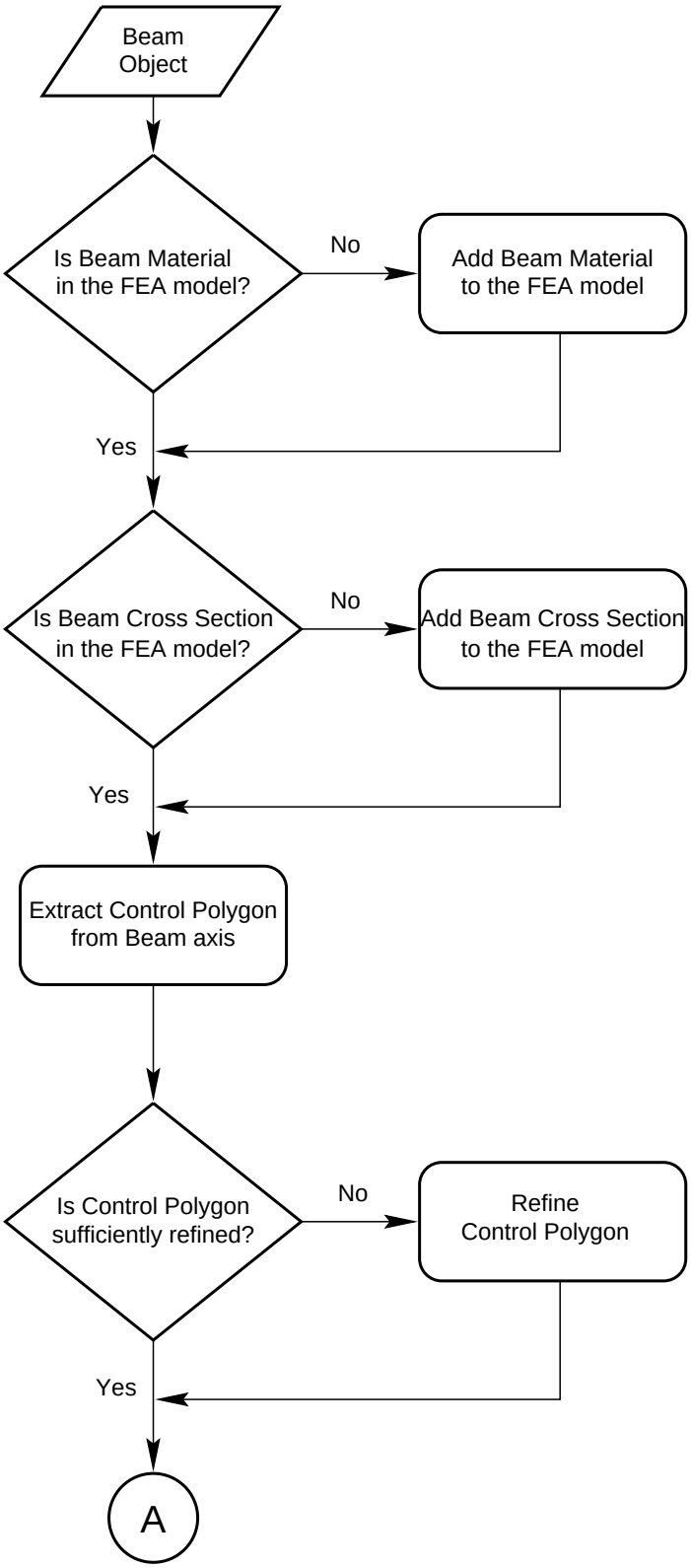

Figure 5.1. Flowchart describing the beam meshing process.

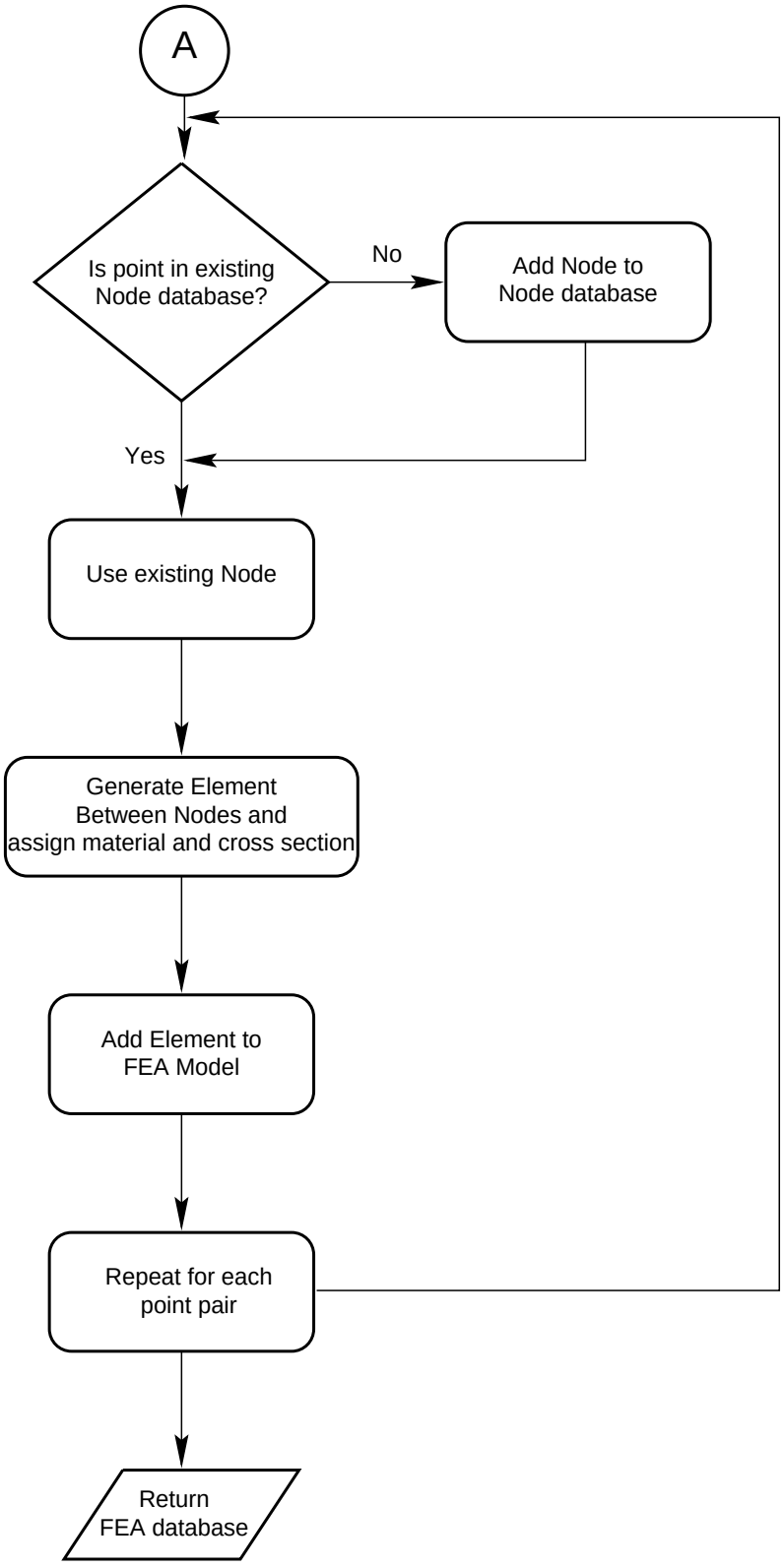

Figure 5.2. Figure 5.1 continued.

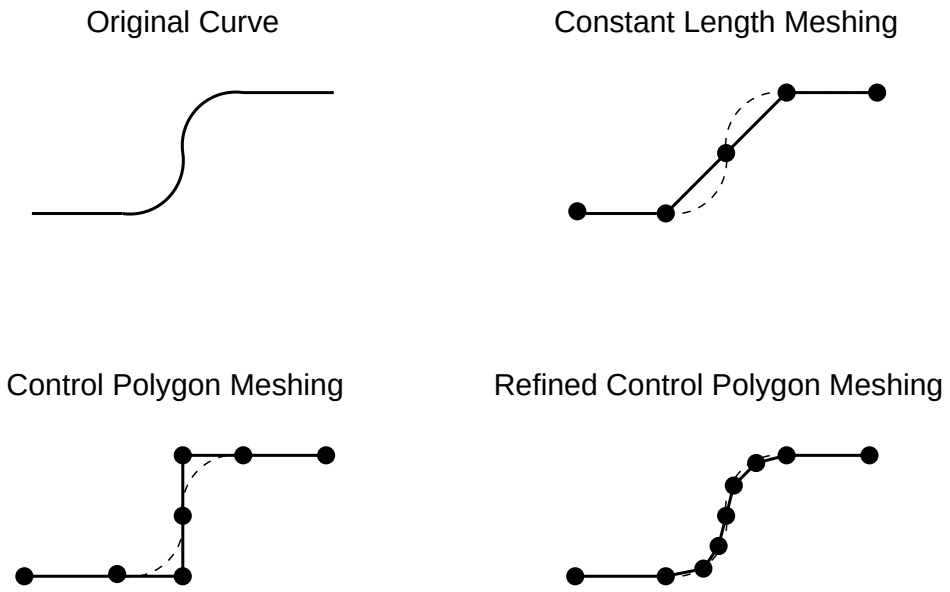

Figure 5.3. Different approaches to curve meshing.

translation and rotation, and the other is translationally constrained orthogonal to the hinge axis. If there is only one base hinge point, which is common in the single beam case, that point will carry the thrust loads and will be completely constrained. However, if there are two hinge points, the one completely constrained varies depending upon the bearing constraints. These thrust bearing constraints are controlled by the *ThrustBearing* attribute and there are four allowable possibilities[Figure 5.4]. If the origin of the link lies between the hinge points, which is usually the case, the *ThrustBearing* attribute controls the boundary conditions as follows:

- *Right* defines the hinge point in the positive Z direction as the thrust bearing and completely constrains its displacement.
- *Left* defines the hinge point in the negative Z direction as the thrust bearing and completely constrains its displacement.
- *Inside* defines the hinge point in the direction of the axial force vector as the thrust bearing and completely constrains its displacement.
- *Outside* defines the hinge point in the opposite direction of the axial force vector as the thrust bearing and completely constrains its displacement.

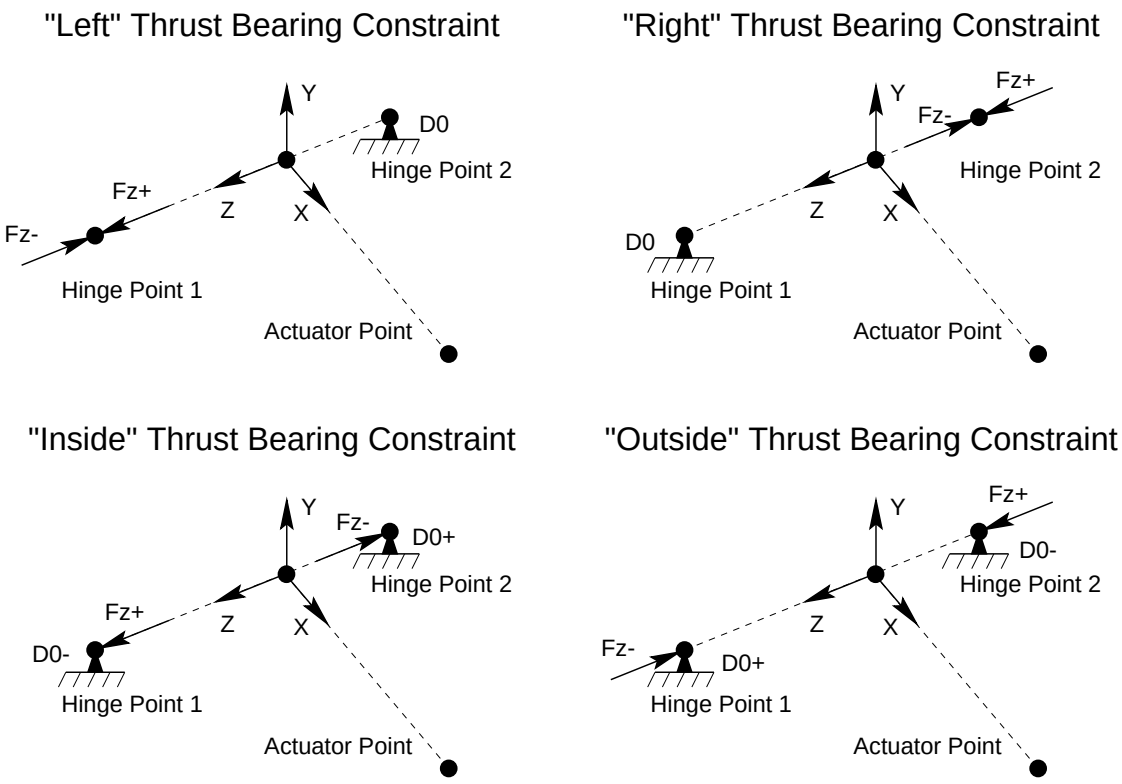

Figure 5.4. Displacement and axial force boundary conditions.

If the origin of the link coordinate does not lie between the hinge points, determination of the thrust point proceeds as follows:

- *Right* defines the hinge point which is furthest from the origin of the link coordinate in the positive Z direction, or closest in the negative Z direction, as the thrust bearing and completely constrains its displacement.
- *Left* defines the hinge point which is furthest from the origin of the link coordinate in the negative Z direction, or closest in the positive Z direction, as the thrust bearing and completely constrains its displacement.
- *Inside* defines the furthest hinge point in the direction of the axial force vector as the thrust bearing and completely constrains its displacement.
- *Outside* defines the closest hinge point in the direction of the axial force vector as the thrust bearing and completely constrains its displacement.

Some FEA packages include displacement boundary conditions as part of the FEA model and do not allow modification of these during the analysis process. In this

case, the relevant condition for the maximum axial force load case is used for all load cases. This will produce reasonable results for most cases and is not relevant to Ansys or Patran, which were the FEA packages used during this project. In the future, creation of a separate FEA model for those load cases with the incorrect boundary conditions could be performed for those FEA packages that require this approach.

5.1.2 Load Case Decomposition

The load cases determined by the dynamic analysis consist of sets of force and moment vectors describing an instantaneous dynamic load state. For purposes of the FEA, these vectors need to be transformed into a set of static forces that reasonably represent the dynamic loads. Examination of the physics of the system shows that the dynamic loads arise from two different sources: the acceleration applied to the individual link structure, and coupling loads passed down from external link structures. The size and point of application of the coupling loads are known. However, the same cannot be said of the structural inertial load. The center of mass is not an appropriate point to use, as there may not be any structure in that location. Even if structure exists at the center of mass, application of the entire inertial loads to a single point is not a reasonable approach. For that reason, the approach used was to transform the dynamic loads calculated for the base hinge and actuator to the outer joint (or the end of the structure if the structure is the last member of the chain). This slightly overstates the actual load, as the structure is required to carry the inertial loads, but provides a simple and conservative solution.

The first step in this process is to apply a force $\mathbf{f}_{ba}$ to the base actuator point in the direction of the base actuator line of action $\mathbf{u}_b a$. This force is designed to supply the required moment about the base hinge axis and is computed using the effective moment arm r_b as follows:

$$\mathbf{f}_{ba} = \frac{m_z}{r_b}\mathbf{u}_{ba} \tag{5.1}$$

Next, the base load $\mathbf{Ls}_b$ is transformed to the end of the structure. First, a

coordinate system $\mathbf{C}_s$ is defined for the external joint as shown in Figure 5.5. Once the local coordinate system is defined, the equivalent load at that location is computed using the load manipulation tools described in Section 2.2. If there are two hinge points, the load may be broken down into force vectors acting on the hinge and actuator points.

The load decomposition process continues by converting the moment vector $\mathbf{m}$ into sets of force couples. This is accomplished using the length variables d_1, d_2, r, and the actuator direction vector $\mathbf{u}_a$ shown in Figure 5.6. The first step is to calculate the force acting on the actuator mount point as follows. The negative sign is to provide the reaction moment:

$$\mathbf{f}_a = \frac{-m_z}{r(1 - \mathbf{u}_z \cdot \mathbf{u}_a)}\mathbf{u}_a \tag{5.2}$$

The corresponding reaction forces are applied to the hinge points $\mathbf{p}_{H1}$ and $\mathbf{p}_{H2}$ as follows:

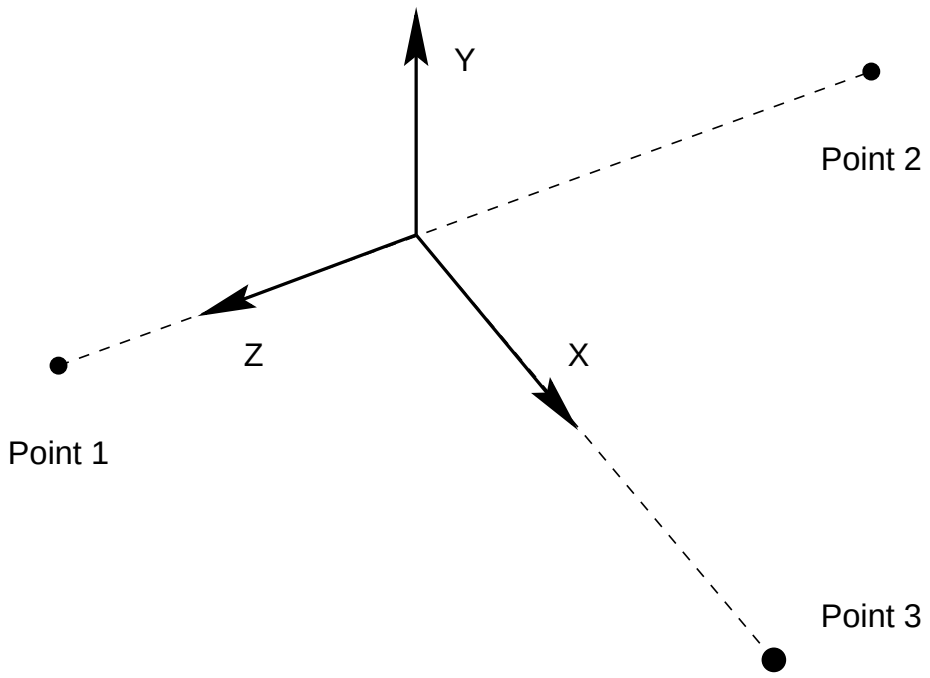

Figure 5.5. Definition of local coordinate system for load decomposition

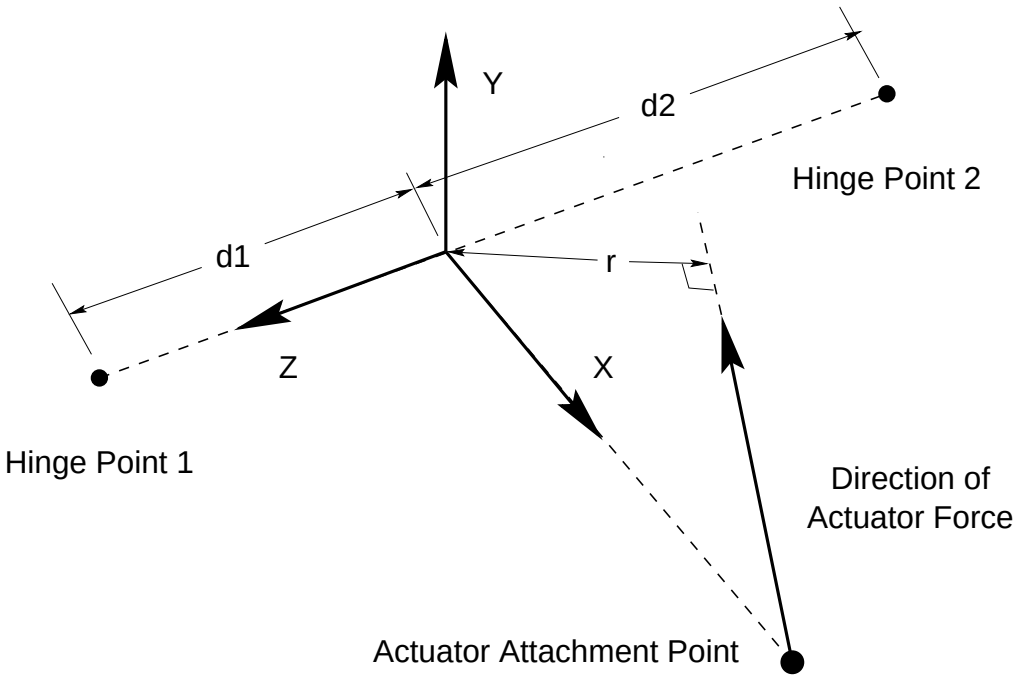

Figure 5.6. Description of variables used for load decomposition

$$\mathbf{f}_{H1_{mz}} = \frac{m_z d_2}{r(d_1+d_2)}\mathbf{u}_a \tag{5.3}$$

$$\mathbf{f}_{H2_{mz}} = \frac{m_z d_1}{r(d_1+d_2)}\mathbf{u}_a \tag{5.4}$$

$$\mathbf{f}_{z_{mz}} = \frac{m_z}{r\mathbf{u}_z \cdot \mathbf{u}_a}\mathbf{u}_z \tag{5.5}$$

Then the non-hinge moment vector components are also converted to force couples.

$$\mathbf{f}_{H1_{mx}} = \frac{-m_x}{d_1+d_2}\mathbf{u}_y \tag{5.6}$$

$$\mathbf{f}_{H2_{mx}} = \frac{m_x}{d_1+d_2}\mathbf{u}_y \tag{5.7}$$

$$\mathbf{f}_{H1_{my}} = \frac{-m_y}{d_1+d_2}\mathbf{u}_x \tag{5.8}$$

$$\mathbf{f}_{H2_{my}} = \frac{m_y}{d_1+d_2}\mathbf{u}_x \tag{5.9}$$

The non-hinge axis forces $\mathbf{f}$ are computed by requiring that the resulting moment contribution be zero.

$$\mathbf{f}_{H1_{fx}} = \frac{\mathbf{f}_x d2}{(d_1 + d_2)} \tag{5.10}$$

$$\mathbf{f}_{H2_{fx}} = \frac{\mathbf{f}_x d1}{(d_1 + d_2)} \tag{5.11}$$

$$\mathbf{f}_{H1_{fy}} = \frac{\mathbf{f}_y d2}{(d_1 + d_2)} \tag{5.12}$$

$$\mathbf{f}_{H2_{fy}} = \frac{\mathbf{f}_y d1}{(d_1 + d_2)} \tag{5.13}$$

These results are then combined to determine the final load state.

$$\mathbf{f}_{H1} = \mathbf{f}_{H1_{mz}} + \mathbf{f}_{H1_{mx}} + \mathbf{f}_{H1_{my}} + \mathbf{f}_{H1_{fx}} + \mathbf{f}_{H1_{fy}} \tag{5.14}$$

$$\mathbf{f}_{H2} = \mathbf{f}_{H2_{mz}} + \mathbf{f}_{H2_{mx}} + \mathbf{f}_{H2_{my}} + \mathbf{f}_{H2_{fx}} + \mathbf{f}_{H2_{fy}} \tag{5.15}$$

The hinge axis thrust component $\mathbf{f}_t$ is computed separately,

$$\mathbf{f}_t = \mathbf{f}_{z_{mz}} + \mathbf{f}_z \tag{5.16}$$

and applied to the point designated to carry the thrust load.

In the event there is only a single hinge point, which may be the case for some single beam structures, the hinge is required to support the nonhinge axis moment components. For this case, the decomposition of these components into forces is not performed and instead these components are applied as moments to the FEA model.

A slightly more complex solution is used for structures with multiple external joints. For these cases, one of the external links $\mathbf{Ln}_s$ is determined to be the most significant, usually the one at the greatest distance from the base. The worst case loads of the other external links $\mathbf{Ls}_i$ are applied to their respective attachment points. These loads are then transformed to $\mathbf{C}_s$ and subtracted from the transformed base load,

$$\mathbf{Ls}_s = \mathbf{T}_b^s \mathbf{Ls}_b - \sum_{i=1}^{n-1} \mathbf{T}_i^s \mathbf{Ls}_i \tag{5.17}$$

where $\mathbf{T}_b^s$ is the load transformation from $\mathbf{C}_b$ to $\mathbf{C}_s$ using the techniques described in Section 2.2.1.

This process is repeated for each of the 16 dynamic load cases, with the exception that each is checked before decomposition to ensure that the same load case is not applied more than once. Two additional load cases are also generated using the maximum $\mathbf{f}_{ae}$ and minimum $\mathbf{f}_{ar}$ actuator force,

$$\mathbf{f}_{ae} = \frac{\pi d^2 P}{4}\mathbf{u}_{ba} \tag{5.18}$$

$$\mathbf{f}_{ar} = \frac{-\pi(d^2 - d_r^2)P}{4}\mathbf{u}_{ba} \tag{5.19}$$

where d is the actuator bore diameter, d_r the actuator rod diameter, P is the maximum allowable fluid pressure and $\mathbf{u}_{ba}$ is a unit vector in the direction of the actuator line of action during extension. The resulting hinge moment produced by these forces is then calculated and the resulting load transformed and decomposed the same as the other load cases. These two load cases should be greater than the worst case dynamic loads and it is assumed that in normal operation actuator forces of this magnitude will not be required. However, these load cases are added to insure structural integrity in the event that some unexpected problem occurs, such as control system failure.

5.1.3 Exportation and Importation of the FEA Model

Once FEA models have been constructed for each link in the structure, the models can be exported to any of several commercial FEA packages. The use of external FEA packages for the required structural analysis has been proposed and implemented by others[147] and provides several advantages. These include:

- "Proven" analysis code and support.
- Existing preprocessing and post processing tools.
- Existing visualization capability.
- Reduced code construction and maintenance.
- Easy extensibility as additional analytical functionality is desired.

However, the use of external FEA packages is not without disadvantages, including:

- Difficulties with translation and exporting data from the internal CAD system representation to a representation acceptable to the FEA system.

- Difficulties with importing and interpretation of results from the FEA system back to the CAD system.
- Potential compatibility problems due to changes to the FEA system functionality and the details of its input and output data formats.

After careful consideration, none of these disadvantages appeared unsurmountable, and the decision was made to proceed with an external FEA package.

To support a number of different FEA packages, the internal FEA model is post processed to provide the necessary input files. This process typically involves writing out the components of the *FEAStateObj* in a format acceptable to the commercial FEA package. Wrapper methods have been created for exporting FEA models in several different formats, including Patran neutral format as well as the Cosmos/M GEOSTAR and Ansys file formats. Job control files for the Patran/FEA analysis modules are also generated during this process. A representation of an FEA model imported into the Cosmos/M GEOSTAR FEA package is shown in Figure 5.7.

Importation of the results of the FEA analysis into the CAD environment was desirable for several reasons:

- To provide two way communication between the DynaFrame system and the FEA package. Without this two way communication, manual interpretation and data entry would be required.

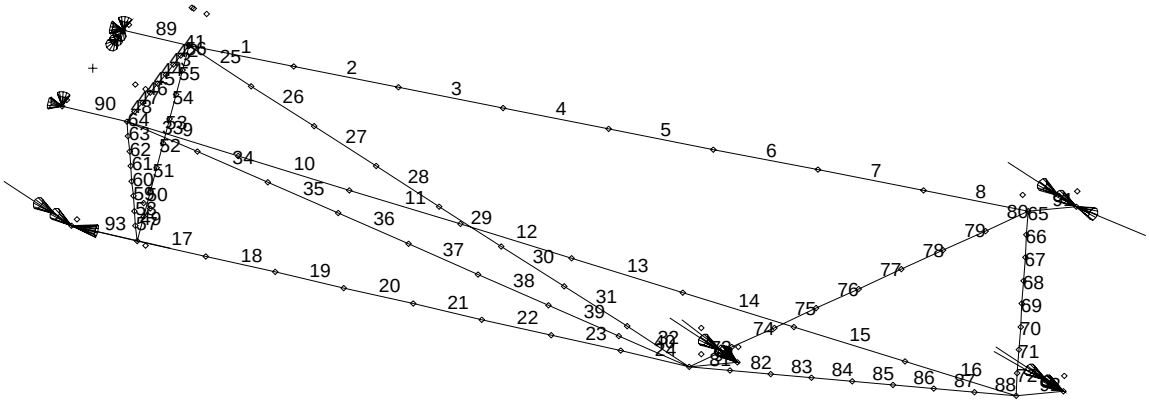

Figure 5.7. A graphic representation of the FEA model of a simple link structure

- To allow for discrete optimization. What little support is provided by current FEA packages is geared towards continuous optimization.
- To accurately reflect the changes in model geometry due to variation in beam sizes. Although not a particularly strong function, the FEA model geometry is not independent of the sizes of the structural members.

For beam elements, the results of the FEA are typically a set of stress tensors for each element in each load case. Often the force and moment loads on each element are also available. To provide a reasonable representation for the stress tensor quantities, a *StressObj* was defined as follows:

$$\mathbf{St}_i = \begin{cases} \mathbf{C}_i & \text{a local coordinate system} \\ \mathbf{S}_i & \text{the stress tensor, a } (3 \times 3) \text{ matrix} \\ \mathbf{P}_i & \text{the principal coordinate system} \\ \sigma_i & \text{a vector whose values represent the principal normal stresses} \\ \tau_i & \text{a vector whose values represent the principal shear stresses} \\ v_i & \text{a scalar value calculated using the Von Mises failure criteria} \\ t_i & \text{a scalar value calculated using the Tresca failure criteria} \end{cases}$$

To reduce the computational requirements, just the local coordinate system $\mathbf{C}_i$ and the stress tensor $\mathbf{S}_i$ are required, with the principal stress information and failure criterias being calculated on an as-needed basis. The *StressObj* provides a very convenient representation to store and manipulate information about the stress state at a point. However additional information was available from the FEA results that proved quite useful. To provide a means of storing and manipulating this information, the *FEAElementLoadObj* object was defined.

$$\mathbf{El}_i \begin{cases} e & \text{the related element number} \\ \mathbf{St}_i & \text{a stress object} \\ \mathbf{Ln}_i & \text{the corresponding element load} \\ p_i & \text{a numerical identifier locating the stress object} \\ l_i & \text{a numerical identifier of the relevant load case} \\ b_i & \text{a beam identification string of the relevant beam} \end{cases}$$

The FEA results are read by a pre processor similar to the post-processors used to output the FEA model. The pre processor reads in the result files created by the FEA package, creates an *FEAElementLoadObj* object for each element, and adds

it to a list containing all of the element loads. This list of element loads is then used by the structural optimization module to refine the link structures. Currently, preprocessor support is limited to beam element analysis that has been produced by the Ansys or Patran/FEA solver. This should not be considered a major constraint, as the extensibility and object oriented nature of this project makes interfacing with additional FEA packages into a simple task which is performed on an as needed basis.

5.2 Buckling Analysis

Stress limitations are often not enough to ensure proper structural performance, particularly when structural optimization is planned. As the section modulus of the members decrease, so does their stability, and constraints on buckling are necessary to avoid structural failure. A wide variety of different types of buckling analysis are available in commercial FEA packages. However, it can be difficult to algorithmically determine the correct failure mode and necessary correction from the results of these analyses. It also requires another iteration of the analysis process for every load case. Because of these difficulties, it was decided to use the closed form American Institute of Steel Construction (AISC) guidelines[3] for the the current implementation of this project.

For compression members, one of the principal indicators of stability is the slenderness ratio Kl/r where:

$$\begin{aligned} K &= \text{effective length factor} \\ &= 0.65 \text{ for frame structures} \\ &= 1.0 \text{ for truss structures} \\ l &= \text{length of the member} \\ r &= \text{appropriate radius of gyration} \end{aligned}$$

In general, the AISC guidelines prohibit compression members with a slenderness ratio greater than 200. Another nondimensional ratio that is used is known as

the column slenderness ration C_c which divides the elastic and inelastic buckling ranges. The column slenderness ration C_c is defined as:

$$C_c = \sqrt{\frac{2\pi^2 E}{F_y}} \tag{5.20}$$

$$\tag{5.21}$$

where:

$$\begin{aligned} E &= \text{modulus of elasticity} \\ F_y &= \text{yield strength of the material} \end{aligned}$$

For mild steel, this ratio $C_c \approx 128$ and is dependent only upon the structural material. Thus it need only be calculated once for each material type. Once the slenderness ratio and C_c have been computed, the allowable compressive stress in the member in the absence of bending F_a can be determined. Depending upon the compressive failure mode, a different formula is used. If $Kl/r \geq C_c$ then:

$$F_a = \frac{12\pi^2 E}{23(Kl/r)^2} \tag{5.22}$$

else, if $Kl/r < C_c$ then:

$$F_a = \frac{\left[1 - \frac{(Kl/r)^2}{2C_c^2}\right] F_y}{\frac{5}{3} + \frac{3(Kl/r)}{8C_c} - \frac{(Kl/r)^3}{8C_c^3}} \tag{5.23}$$

If $l/r > 120$, then the following correction of F_a can be used for bracing and other secondary members:

$$F_{as} = \frac{F_a}{1.6 - \frac{l}{200r}} \tag{5.24}$$

These functions have an included safety factor from the Euler buckling load that varies from 5/3 for $Kl/r = 0$ to 23/12 when $Kl/r = C_c$ and is constant at 23/12 for $Kl/r > C_c$. The safety factor specified by the *BucklingSF* attribute discussed in the next chapter is in addition to this AISC safety factor.

Once the allowable compressive stress has been determined, an adjustment must be made for bending loads. The AISC specification states that members subject to both compressive and bending loads must satisfy the following conditions:

$$\frac{f_a}{F_a} + \frac{C_{mx} f_{bx}}{\left(1 - \frac{f_a}{F'_{ex}}\right) F_{bx}} + \frac{C_{my} f_{by}}{\left(1 - \frac{f_a}{F'_{ey}}\right) F_{by}} \leq 1.0 \tag{5.25}$$

$$\frac{f_a}{0.60 F_y} + \frac{f_{bx}}{F_{bx}} + \frac{f_{by}}{F_{by}} \leq 1.0 \tag{5.26}$$

where:

$$\begin{aligned}
F_a &= \text{allowable axial stress from equation 5.22, 5.23 or 5.24} \\
F_b &= \text{allowable compressive bending stress} \\
F'_e &= \frac{12\pi^2 E}{23(K l_b / r_b)^2} && (5.27) \\
C_m &= \left(0.6 - 0.4\frac{M_1}{M_2}\right) \text{ but not less than } 0.4 && (5.28) \\
M_1 &= \text{smaller in-plane bending moment} \\
M_2 &= \text{larger in-plane bending moment} \\
l_b &= \text{unbraced in-plane length} \\
r_b &= \text{in-plane radius of gyration} \\
f_a &= \text{computed axial stress} \\
&= f/A && (5.29) \\
f_b &= \text{computed bending stress} \\
&= M_2/S && (5.30)
\end{aligned}$$

By ASIC specification, F_b is cross section dependent and varies from $0.6F_y$ for most applications to $0.75F_y$ for special cross sections under very restricted conditions. For the purposes of this project, $F_b = 0.6F_y$. There is also a sign dependency in C_m in which the worst case occurs when the moments M_1 and M_2 are of equal magnitude and in opposite directions, that results in the maximum eccentricity and $C_m = 1$. The stresses f_a and f_b along with the slenderness ratio (Kl/r) are different for every member in the structure and will need to be computed for each

member after the FEA has been performed and the related member loads have been determined. Algorithms to perform these computations have been implemented and are accessible to the designer in a number of ways. These routines may be called directly or may be called indirectly as part of a larger design verification or improvement process. The interaction between the stability and stress constraints, the beam member catalog, and the designer is considered structural optimization and is the subject of the next chapter.

CHAPTER 6

STRUCTURAL OPTIMIZATION

In general, the structural optimization process has three major components:

1. The generation of the structural topology.
2. Geometric optimization.
3. Member resizing.

The fundamental structural topology was heuristically determined during the structural synthesis process and is assumed to be close to the economic optimal. However, the initial member sizes were also heuristically determined and may be either too small to fulfill the design constraints or larger than necessary. For this reason, it is both desirable and necessary to perform some additional structural optimization to improve the design.

6.1 Background

As was discussed earlier, the history of structural optimization can be traced back over a hundred years to the initial work performed by Maxwell[90] and Mitchell[91]. However, the limitations of their work precluded application to general engineering problems. Over the intervening years, several improvements to their initial approaches have been proposed, usually in an effort to reduce the mathematical complexity or improve the range of application.

After the second world war, a great deal of effort went into optimization of planar trusses and frames. This resulted in what is known as "plastic collapse" design in which the expected service loads are scaled up and the individual members sized to "collapse" at that level. Using this approach, the problem could usually be stated as a linear programming problem and solved; however, they did not consider stress, deflection, or buckling constraints[123].

The field of modern structural optimization is generally considered to begin in 1960 with the publication by Schmit of the application of nonlinear mathematical programming methods and numerical techniques using a computer to the optimization of a simple three-bar truss[146]. One of the novel findings presented in this paper was that one member of the truss was not fully stressed by either load condition at the minimum weight design. Unfortunately, the computational requirements of the early nonlinear mathematical programming methods restricted their application to simple problems with a small number of design variables. This limitation resulted in a number of computational improvements to existing approaches as well as some new approaches.

A great deal of effort has gone into improving the computational efficiency of the various mathematical programming approaches[60, 123, 100, 117]. Most of these improvements reduced the number of finite element analyses required to determine the optimal solution. With this reduction in analysis, computation of the sensitivity or gradient rapidly became a significant factor and methods evolved to reduce the gradient computation. Many of these improvements involve either sensitivity approximation[9, 52, 51, 28, 101], variable transformation[125], or other approximation methods[118, 125, 52, 147] to reduce the computational requirements. Usually an initial FEA analysis is performed and the results are used to compute the necessary approximations. From these approximations, a new design is determined and checked against the approximations. This process is continued until the design sufficiently deviates from the analyzed design, in which case a FEA is performed on the new design. This process is repeated until some criteria is satisfied, usually minimal deviation from the previous design. Most of these approximation techniques are reasonably accurate (and may be exact) for truss structures but suffer when extended to frame structures[51, 87, 156, 42].

As with truss structures, a great deal of effort has gone into improving the performance of the approximations used in frame structures although there is still considerable room for improvement[147]. Over the years, several survey papers have been written[123, 7, 82, 146, 149] and serve as an excellent initial reference to

the subject. In addition, recently a number of books have published on the subject of structural optimization[75, 12, 6].

Some researchers have worked on optimization of the structural geometry, typically by including the coordinates of each joint as design variables[133, 74]. These approaches greatly increases the size of the design space and corresponding computational cost. Although of interest to truss designers, the functional requirements of the primary node locations of the link structures precluded application of this methodology to this project. The mathematical programming approaches have also been extended by several researchers to include frame structures[158]. As most frame structures are statically indeterminate, this greatly increases the complexity of the optimization problem and remains an active area of research and beyond the scope of this project.

Another area of research in structural optimization examines the problem of constraints on the natural frequency of the structure, an example of which was written by Lin et al.[84]. These types of constraints are of great interest to those involved in the design of earthquake resistant structures, light weight aerodynamic structures, and structures for low gravity environments, but are of limited use for the types of structures considered during this project. To date, no references have been located that include explicit dynamic loads into the structural optimization process, although they have been implicitly included in some space structure simulations[140].

Despite all of the effort, significant problems with the mathematical programming approach remain. Problems with discrete variables, local optima, high computational cost, difficulties in problem formulation, and stability remain. To avoid some of the difficulties presented by the various mathematical programming approaches, a wide variety of alternative optimization methods have been proposed. One set of these alternative methods can be described as stochastic approaches in which a certain amount of random variation is introduced, either in a attempt to avoid local optima[111, 110] or to represent some other phenomenon[108]. Others such as the those proposed by Jung et al.[67] integrate fuzzy constraints into the

structural optimization process.

A second set of alternative methods known as *optimal criteria methods* assumes a constraint or set of constraints that will be active at the optimal solution[46, 70, 72]. The classic optimal criterion takes the form of maximum allowable stress and the resulting design is known as a "fully stressed design"[122, 109, 29]. The underlying assumption is that a design that fulfills the optimal criterion is optimal, an assumption proven false by Schmit[123]. However, the assumption provides several mathematical simplifications to the optimization process, as direct computation of design sensitivities are not required and by some transformation of the design variables an optimal design may be produced[45]. The optimal criteria methods have been extended to support several different types of constraints in several differing applications, including displacement[50] stability[71] and other cost constraints. Several systems have been proposed and implemented to optimize large steel frameworks using this approach[135, 79, 50].

6.2 Discrete Optimization

It is often the case in engineering that the design variables are not continuously available but occur in finite discrete sizes or quantities[10]. In these cases, the sensitivity or other derivative information needed for the optimization algorithm is difficult or unavailable and, as such, the standard mathematical programming techniques cannot be directly applied. In certain cases, this information may be functionally approximated and the result rounded up to the next available size[124, 40, 50] but this approach has been demonstrated to potentially lead to incorrect results[64, 138, 55]. Therefore, it is desirable to examine alternative approaches to these types of problems.

There are several known exact discrete optimization methods which guarantee an optimal solution. The most primitive is complete enumeration, in which all possible combinations are evaluated. Although this is guaranteed to produce the optimal solution, the number of required evaluations rapidly becomes prohibitive[155]. The cutting plane and branch and bound methods involve generating and exhaustively

searching combinatorial graphs of the possible solution space, and the controlled enumeration method requires generating and ordering all possible solutions of the cost function[55]. This approach has been extended to include nonlinear problems by Sandgren[119, 120] and others[20]. Not surprisingly, the combinatorial nature of the design space prevents application of these approaches to many real world problems and a number of alternative solutions have been developed.

Another class of discrete optimization methods that have been developed are based upon stochastic search techniques[142, 55]. The most basic form of stochastic search is a random search in which random variables are selected and the resulting structures tested against previous designs and the design constraints. An improvement to the random search method is known as random walk. In this approach, the new design is determined by taking the previous design and randomly incrementing or decrementing the design variables, forming a "step" in the design space. If the new design satisfies the design constraints and is an improvement over the previous design, it becomes the "best" design, otherwise a new step is created and tested. Similar methods based upon directed search have been proposed by Cella et al.[23] and Liebman et al.[83] in which the search continues along the search direction until further improvement ceases. The simulated annealing and genetic algorithms have also been used successfully for discrete optimization, and there is a great deal of ongoing research in this area. All of the preceding optimization methods incur significant computational demands and none guarantee determination of the optimal solution.

A number of heuristic methods have also been devised in an effort to reduce the computational requirements[55]. These involve a methodology described as "segmental linear programming" in which the individual members of a truss structure are divided up into a number of "segments" each a different discrete size and unknown length. Standard linear programming techniques are then used to solve for the unknown segment lengths, after which some form of rounding optimization is performed to arrive at the final discrete design. This reduction of the discrete optimization problem to a linear programming problem by a careful choice of

assumptions has also been implemented by others[142, 95] and is quite attractive in that the behavior of these types of problems are well known and a wide variety of solution packages are available.

Numerous hybrid approaches have been implemented, usually combining some simple linear programming method to get close and then employ a discrete search method to fine-tune the optimal solution[81, 20]. This effort has been extended to nonlinear programming and Boolean search methods by Lu et al.[85]. Although not explicitly stated as discrete optimization, the work by Yoshida et al.[156] shares many similarities with this aspect of the current project, including the use of beam elements, inclusion of an element library with realistic cross-sectional geometry, the use of an equivalent stress constraint, and stress evaluation of multiple extrema points of the cross section.

In general, the field of discrete structural optimization is still quite active with much work still to be performed[138, 148]. Several survey papers[148, 139] and books[56, 55] have been published and should be referred to for detailed descriptions of the various methodologies.

6.2.1 Selected Approach

After review of the existing structural optimization methods and their potential application to this project, the following observations were made:

1. All of the existing structural optimization methods, both continuous as well as discrete, assume a set of unvarying loading conditions for which the structure is optimized.
2. Reducing the number of discrete sizes by grouping the members based upon functional requirements can significantly reduce the computational requirements of the optimization process, as well as reduce the overall fabrication cost.
3. For statically indeterminate structures, that include frame structures similar to those constructed during this project, the stress in any member is a function of all of the other members in the structure.

4. Most of the successful discrete optimization techniques have taken the hybrid approach of finding the continuous optimum first and then searching the immediate area of the design space for the optimal discrete solution.

For the structures of primary interest to this project, observation 1 is not true, as the inertial properties of the structures have a significant affect on the loading conditions. Thus, it appears that exerting significant computational effort to determine an optimal structure, only to modify the loading conditions, would not be a wise use of resources. The existing link structures already have the individual structural members subdivided by function, making inclusion of observation 2 a logical choice. Observation 3 would seem to suggest that a large number of FEA models will have to be solved to arrive at an optimal solution. However, experience suggests that for space frame structures similar to those produced by the structural synthesizer, the load distribution is not a strong function of the member sizes. Thus this observation can be ignored during most of the optimization process. Although observation 4 suggests that the hybrid approaches have a better chance of success, the requirement to formulate and implement the optimization problem in both continuous as well as discrete forms is a significant drawback. This, coupled with the corresponding computational costs and the need to repeat to optimization process several times to account for the changing loads, seems to preclude the use of hybrid approaches at this time.

Given these observations, a very simple discrete optimization scheme was selected based upon reducing the number of different sizes to the functional slots defined in the link structures. With this reduction in the search space, a simple directed search algorithm coupled with a fully stress design optimum criteria was implemented, the details of which will be discussed in the next section.

6.3 Implementation

As discussed previously, once the results of the FEA have been imported into the CAD environment, the structural optimization process can proceed. This is accom-

plished by comparing the element stresses against an allowable value or comparing the structural buckling load against a predefined minimum value. The predefined values for both the stress and buckling are known as the stress safety factor and the buckling safety factor and are contained in the attributes *StructuralSF* and *BucklingSF* respectively. From these values and the yield stress information from the beam material information, values for allowable stress are determined. These attributes may be set to any positive value, but the *StructuralSF* attribute defaults to a safety factor of 3, and the *BucklingSF* attribute defaults to a safety factor of 1.5. It should be remembered that the *BucklingSF* is imposed in addition to the AISC safety factor included in the allowable buckling load.

For ease in comparing the element stress tensors, the Von Mises failure criterion is often used to convert the stress tensor of each element stress into a scalar value[46]. This failure criterion was used as it shows good agreement with experimental data for ductile materials and is determined as follows[32]:

$$\sigma_{VM} = \frac{1}{\sqrt{2}}[(\sigma_x - \sigma_y)^2 + (\sigma_y - \sigma_z)^2 + (\sigma_z - \sigma_x)^2 + 6(\tau_{xy}^2 + \tau_{yz}^2 + \tau_{zx}^2)]^{1/2} \qquad (6.1)$$

where σ_{VM} is the Von Mises stress, σ_i is the normal stress in the i^{th} direction and τ_{ij} is the shear stress in the ij plane, quantities that are directly available from the stress tensor. Provisions have been made for calculation and the use of other failure criteria. However, they have not been included in the current structural optimization implementation.

The first step in any optimization process is to decide upon a metric by which to compare different solutions. This metric is best described as an economic model of the complete process and can be very difficult, if not impossible, to determine with any accuracy[103]. This economic model is commonly known by several different names and will be referred to as the *cost function.* Due to both the dynamic nature of these structures as well as the economic costs of the structural materials, the cost function chosen for the optimization process was based upon structural mass as shown below:

$$\mathcal{W}(\mathbf{x}) = \sum \rho_i V_i \qquad (6.2)$$

where:

$$\begin{aligned} \mathcal{W}(\mathbf{x}) &= \text{the total mass of structure } x \\ \rho_i &= \text{the density of the material } i \\ V_i &= \text{the volume of component } i \end{aligned}$$

For this project, the logical approach was to try to minimize the mass of each individual link structure. For a link structure, equation 6.2 can be written as:

$$\mathcal{W}(\mathbf{x}) = \mathrm{M_b} + \mathrm{M_c} + \mathrm{M_a} + \mathrm{M_e} \tag{6.3}$$

where M_b is defined as:

$$\begin{aligned} M_b &= \sum_{i=1}^{n} M_i \text{ the total mass of all } n \text{ beams} \\ M_i &= \rho_i A_i l_i \\ A_i &= \text{the cross sectional area of beam } i \\ l_i &= \text{the length of the centroidial axis of beam } i \end{aligned}$$

M_c is determined by:

$$\begin{aligned} M_c &= \sum_{i=1}^{m} M_i \text{ the total mass of all } m \text{ clevises} \\ M_i &= \rho_i A_i t_i \\ A_i &= (l_i + w_i/2) w_i \text{ the cross sectional area of clevis } i \\ l_i &= \text{the distance from the pivot to attachment points} \\ w_i &= \text{the width of clevis } i \\ t_i &= \text{the total thickness of the clevis } i \end{aligned}$$

and M_a, the mass of the actuator is defined as:

$$\begin{aligned} M_a &= \rho_i \pi \left(\frac{d_i^2}{4} \right) l_i \\ d_i &= \text{the maximum diameter of actuator } i \\ l_i &= \text{the overall retracted length of actuator } i \end{aligned}$$

and M_e is the total of any user defined external mass applied to the structure. Both M_c and M_a are conservative approximations and should slightly overstate

the mass of these components. It is possible to reduce the uncertainty in these approximations but it was felt that their mass is significantly less than other unknown mass components.

Examination of equation 6.3, the computation of the individual components and the structural synthesis process allows several observations regarding structural optimization.

- The user defined external mass M_e is constant, not dependent upon the structural geometry and can be dropped from equation 6.3 without affecting the optimization process.
- The actuator mass M_a is dependent upon the stroke and size of the actuator. The bore size has very little affect on the structural geometry but a large affect on the possible dynamic loads. The stroke length has more but still a relatively small affect on the structural geometry. Therefore, it seems reasonable to size the actuator primarily based upon the required dynamic loads and drop this term from the cost function.
- None of the structural components are completely independent of the structural geometry. The individual beam lengths and clevis sizes will vary with changes in beam member sizes. Therefore, resynthesis of the structure after modification of beam and/or clevis sizes should be performed.
- Unless specified by the user, the clevis sizes are dependent upon the mating beam sizes. They also provide a relatively minor component of the total structural mass and are at best a rough approximation of the required structural elements. Due to this dependency and approximation it was decided to eliminate M_c from the cost function and rely upon the dependency between clevis and beam sizes.

With these qualifications, the cost function described in equation 6.3 reduces to:

$$\mathcal{W}(\mathbf{x}) = \mathrm{M_b} = \sum_{\mathrm{i}=1}^{\mathrm{n}} \rho_{\mathrm{i}} \mathrm{A_i} \mathrm{l_i} \tag{6.4}$$

with the understanding that re-synthesizing the structure will affect M_c and the new dynamic analysis might require changes in M_a as well.

Examining equation 6.4, it can be observed that for practical reasons, it is unlikely that ρ_i will vary. The difficulties of welding and/or corrosion when using dissimilar materials, although not insurmountable, are sufficient to discourage this approach. In any event there are a limited number of available materials commonly used in the production of standard cross sections. For these reasons, the determination of ρ_i is left to the user and this parameter is not directly included in the optimization process. It should be recognized that the allowable stress values used in the optimization are dependent upon the *StructuralSF* attribute and the yield stress of the material. Thus, increasing the material strength while leaving the *StructuralSF* attribute the same will raise the allowable stress levels and indirectly allow a reduction in structural mass. However, as the choice of a material is generally determined by manufacturing or availability constraints, the material is not included as a variable in the optimization process.

6.4 Member Resizing

For the member resizing process and using the above assumptions regarding ρ_i and l_i in equation 6.4, it is apparent that A_i, the cross sectional area of each of the beams is the only parameter left to be optimized. Unfortunately, there are several problems with using area as an optimization parameter. These include:

- There is a poor correlation between area and the inertial properties of the cross section. The inertial properties have a direct correlation to stress levels and allowable buckling loads.
- Maximum stress levels and allowable buckling loads may be dependent upon the rotational orientation of the cross section about its centroid.
- Due to the discrete nature of the available beam cross sections, the use of traditional continuous optimization techniques is very difficult and of questionable value.
- For most structural cross sections, the area is a function of between two and four independent variables describing overall dimensions, thickness and other geometric parameters. Allowable values of these geometric parameters may be

further restricted by manufacturing or other economic constraints.

- There may be manufacturing or other constraints upon one or more of the geometric parameters defining the cross section.
- There are multiple types of cross sections available, and these may be further restricted by manufacturing or other constraints.
- The number of allowable cross sections from which a link structure was constructed is usually much less than the number of beams in the structure. This is due to both manufacturing as well as economic constraints.

To construct a useful optimization system, these difficulties should be addressed. In the current implementation of the optimizer, a number of assumptions were used to reduce the scope of the problem to a manageable size. These assumptions were as follows:

- Cross sectional type would not be used as a variable in the optimization process but as a user defined constant. Thus, the optimization process would try to find the best cross section of a given type.
- The orientation of the cross section is primarily based upon manufacturing constraints, and as such will be specified by the structural synthesizer and not included in the optimization process.
- For nonradial symmetric cross sections, which include all noncircular or nonsquare cross sections, the independent length and width variables are reduced to a single variable by minimizing the change in the height/width ratio. Given the discrete nature of the available beam sizes, it is not possible to maintain an exact height/width ratio. Therefore, for a given height, the width closest to the desired ratio is selected.
- There will be no more than four different cross sectional sizes used on any single link structure. Furthermore, these would be restricted to predefined functional groups and all beams in a group will be considered the same for the purpose of structural synthesis. This restriction is implemented to avoid greatly increasing the complexity of the structural synthesizer and can be

worked around if deemed necessary.

- Maximizing cross section inertial properties while minimizing area leads to large cross sections with extremely thin walls. Thus it seems reasonable to define a user specified minimum wall thickness. This is accomplished by using the one of the following attributes: *StructMinWall, BaseMinWall, EndMinWall, BraceMinWall.* If not explicitly defined by the user, these attributes default to the wall thickness of the initial cross sections.

With these assumptions, the optimization problem is reduced to a workable, although still complex, problem. The discrete nature of the available beam sizes prevents the use of traditional optimization techniques, so a different approach was implemented, a flowchart of which is shown in Figure 6.1.

The approach used was to use the resulting nodal forces from the FEA and beam theory to calculate approximate expected values of the element stresses for the new cross section. The use of nodal forces rather than stress approximations was also proposed by Vanderplaats et al.[147] and a similar approach was implemented for this project. This allows for the selection and testing of several different cross sectional sizes from the beam catalog without requiring performing an FEA on each of the potential solutions. This approach is based upon the assumption that the load paths are primarily dependent upon the structural geometry and reasonably insensitive to the beam sizes. Experience has shown that this is a reasonable assumption, as the changes in dynamic loads due to inertial modifications are usually significantly larger than the internal load redistribution.

The first step in the optimization process is to import the element stresses and loads from the FEA package, as discussed in Chapter 5. The Von Mises stress is then calculated for each element and compared against the other elements used to mesh each beam. The element with the highest Von Mises stress is defined as the current stress for that beam.

Once current stresses have been determined for each structural member, an estimated stress for each structural member is determined. This estimated stress is calculated using general beam theory[48] modified to include axial stresses and

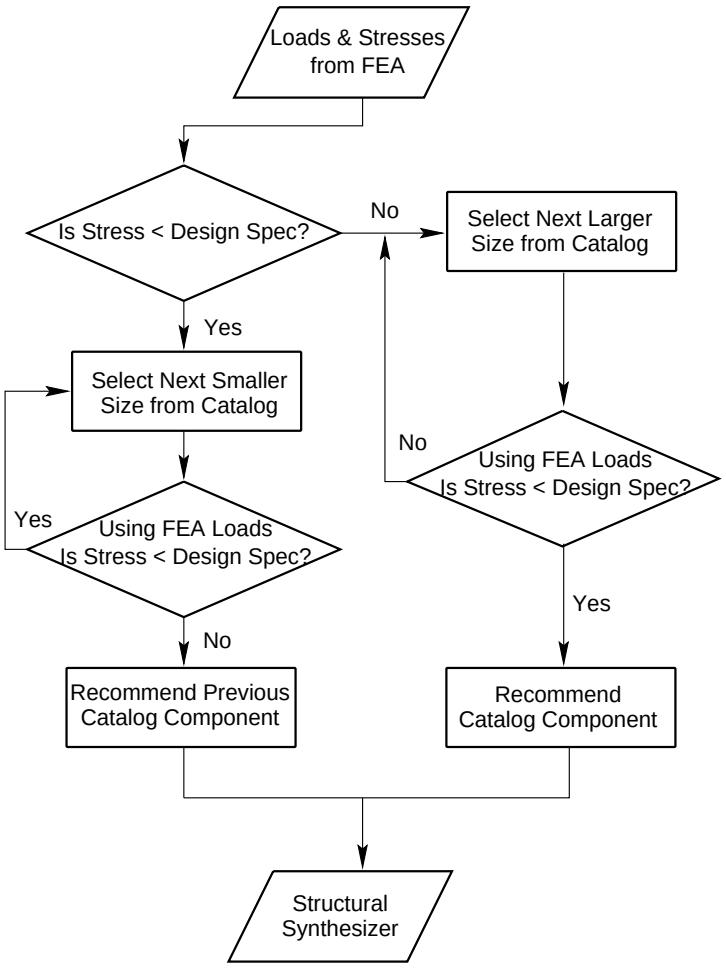

Figure 6.1. Optimization process flowchart

the maximum magnitude as shown below:

$$\sigma_x = abs\left|\frac{(M_y I_z + M_z I_{yz})z - (M_z I_y + M_y I_{yz})y}{I_y I_z - I_{yz}^2}\right| + abs\left|F_x\right| A \qquad (6.5)$$

where:

M_i = the moment about the i^{th} axis

F_i = the force in the i^{th} direction

I_i = the moment of inertia about the i^{th} axis

I_{ij} = the product of inertia relating the i^{th} and j^{th}axis

A = the area of the cross section

y, z = the distance from the centroid and the extreme cross section material measured in the y and z direction

This approximation assumes that the majority of the stress is due to bending or axial loads and does not include any stresses due to shear or torsional loads. For most space frame structures, this is a reasonable assumption, but there may be difficulties for single beam structures. For that reason, and to improve the accuracy of the stress estimates, a correction factor σ_{cf} was calculated as follows:

$$\sigma_{cf} = \frac{\sigma_x}{\sigma_{FEA}} \tag{6.6}$$

The new estimated stress σ_{ne} is then determined by:

$$\sigma_{ne} = \sigma_{cf}\sigma_{nx} \tag{6.7}$$

where σ_{nx} is the stress for the new cross section calculated from equation 6.5.

Using this stress estimate, the optimization proceeds by looking up a new cross section from the beam catalog using some search methods defined for this project. In addition to the minimum wall thickness attributes, there were several other attributes that proved useful in providing control over the catalog search process. One attribute type that proved necessary was a step increment *(SizeStepInc, WallStepInc)* on both the overall size as well as the wall thickness. This allows the user to eliminate some cross sections that may be uncommon or difficult to acquire. The second type is a tolerance range on the incremental step size *(SizeTol, WallTol)* that proved necessary as the available catalog sizes may not be available in the desired incremental size.

With these attributes, the catalog search process is reduced to looking for the next incremental size, returning the cross section closest to the desired increment if it is within the allowable tolerance, or returning an error. The direction of the search and exit conditions is dependent upon whether σ_{FEA} is greater or less than the allowable stress σ_a. If ($\sigma_{FEA} < \sigma_a$), then progressively larger beam cross sections are located in the catalog, the stress calculation performed and the result

compared to σ_a. This continues until ($\sigma_{ne} > \sigma_a$), in which case the previous cross section is recommended. In the event ($\sigma_{FEA} > \sigma_a$) then progressively smaller sections are located in the catalog, σ_{ne} calculated and compared to σ_a. The exit condition in this case is when ($\sigma_{ne} < \sigma_a$), after which the last cross section tested is recommended.

Once a set of new member sizes for each link structure have been recommended, the related attributes are entered by the user and the structures are resynthesized. The structural inertial properties are then determined and a new dynamic analysis is performed using the kinematic configurations determined to produce the worst loads. The resulting dynamic loads are then used in the creation of new FEA models, that are reanalyzed and the results imported back into the DynaFrame system. The process is repeated until the link structures are deemed acceptable and will result in near optimum structures.

A three-link example structure was created and the results of the structural optimization process are shown in Figure 6.2. Several different initial conditions for the structural optimization process were used, varying from an initial weight varying between 63 and 3946 pounds. The structural optimization algorithm proved

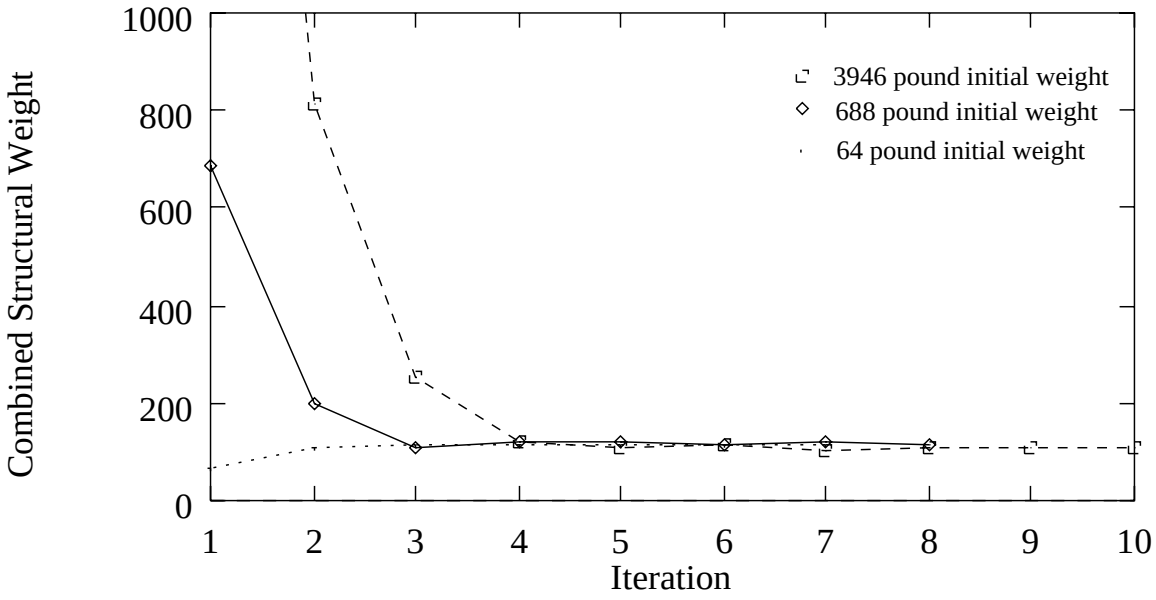

Figure 6.2. Plot of structural weight of a three-link structure using multiple initial conditions. Each iteration requires a single FEA.

sensitive to initial conditions, arriving at a slightly different optimal structure for each initial condition. This is not surprising as there are some continuous variables, such as actuator stroke, which are modified during the optimization process as needed. Nevertheless, the final structural weights differ by less than 9 percent and often can be improved with insight and minimal intervention from the design engineer.

Although it is possible to completely automate this optimization process, there were several reasons for not doing so. First, the human intervention required in each loop provides a measure of stability into the optimization process. Second, it is often desirable for manufacturing reasons to use a smaller number of beam sizes. Last, the human intervention provides the ability to eliminate individual structural members that may not contribute significantly to the structural strength.

Despite the adequate performance of the structural optimization process, improvement is still desirable. The most significant factor appears to be the interdependence of some of the nonoptimized variables to the structural member sizes. This is particularly evident in the dynamic loads, that can be significant functions of the structural inertia but are only indirectly included in the optimization process, requiring a loop through the dynamic and FEA analysis to update the element loads. Although a nontrivial problem, it appears possible to incorporate some estimation of the sensitivity of the dynamic loads to changes in structural inertia. Detailed examination of this problem is left for future research efforts.

6.5 Numerical Results

There are a number of classic optimization problems reported in the existing literature and a comparison of the structural optimization schema implemented in the DynaFrame with these examples is appropriate. The first two are truss structures, a 10-member planar truss and a 25-member space truss and the third is often referred to as the "portal frame" problem and is the most common frame example referenced in the literature.

6.5.1 10-Bar Planar Truss

The 10-bar planar truss is a very common example and has been used to illustrate many truss optimization approaches[46, 70, 72, 60, 100, 95, 147, 20]. The topology of the truss is shown in Figure 6.3. The truss is usually subject to the single load case shown and steel was selected as the material. Additional constraints include a stress limit in all members of less than 25,000 psi and the displacements of all nodes shall be less than 0.2 inches. To compare the implemented optimizer against those previously published, a *TrussObj* was implemented in the DynaFrame System and used for the purpose of modeling this and the following example. Details of the implementation of this object are described in the DynaFrame User's Manual. A graph of the structural optimization process for the 10-bar truss using two different initial conditions and only stress constraints is shown in Figure 6.4. Also included is data from a continuous optimization of the same structure performed by Vanderplaats et al.[147] who determined an optimal structural weight of 1497 pounds. The discrete optimization process produced a minimum weight structure that satisfied the design constraints and weighed 1616 pounds, that was within 8 percent of the continuous solution. The slight increase in structural weight was

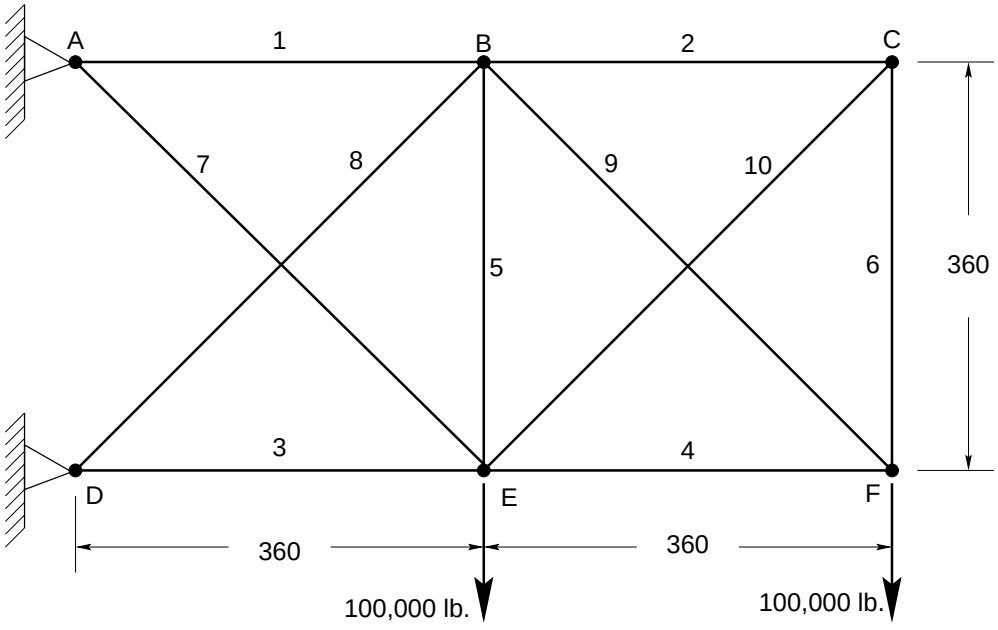

Figure 6.3. 10-bar planar truss

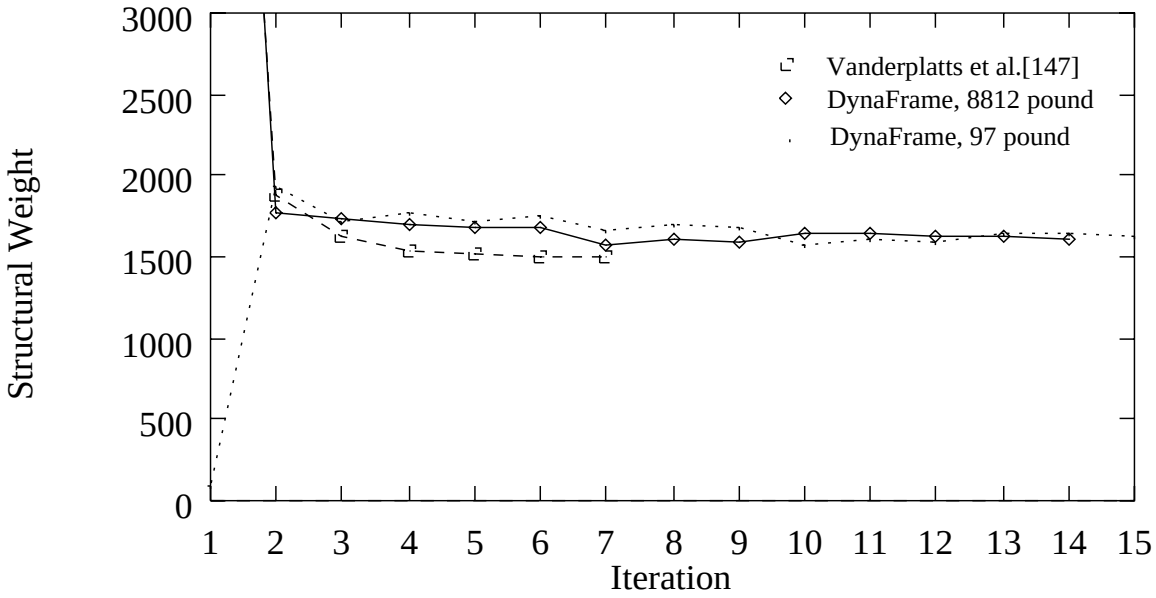

Figure 6.4. Plot of the structural optimization process for the 10-bar planar truss.

expected, as a well performed continuous optimization should always result in a solution at least as good as the best discrete optimization solution. For the discrete solution, the average safety factor of members constrained by stress was 1.137, or roughly 14 percent greater than optimal. Given the simplistic nature of the discrete optimization algorithm and the uncertainties inherent in structural loading, this performance was quite acceptable. Nevertheless, observations suggest some simple extensions to the discrete optimization algorithm.

The first observation is that during most of the discrete optimization process there was quite a bit of oscillation around the minimum weight as slight modifications in the load distribution resulted in discrete changes in the sizes of structural members. This is not surprising and can be corrected by providing some additional coupling between the members during the member selection process, usually by some form of sensitivity analysis. An alternative that may reduce this problem without requiring sensitivity analysis would be to alter the nature of the member selection criteria. Currently, members with a safety factor below 1.0 are not permitted and allowing slight deviations of the safety factor below this threshold early in the optimization process may provide improved performance. Implementation

and exploration of this is left for future work.

6.5.2 25-Bar Space Truss

The 25-bar space truss is also a very common 3-D truss and is often used in the literature[46, 70, 60, 117, 95]. An example of this structure with the members numbered and the nodes lettered is shown in Figure 6.5. All units are inches and the material is assumed to be aluminum with an elastic modulus of 10,000,000 psi. The truss structure is optimized to support the two load cases described in Table 6.1

In addition, the minimal size of any member is constrained to be greater than 0.1 in^2 and the maximum allowable stress is constrained to be less that 40,000

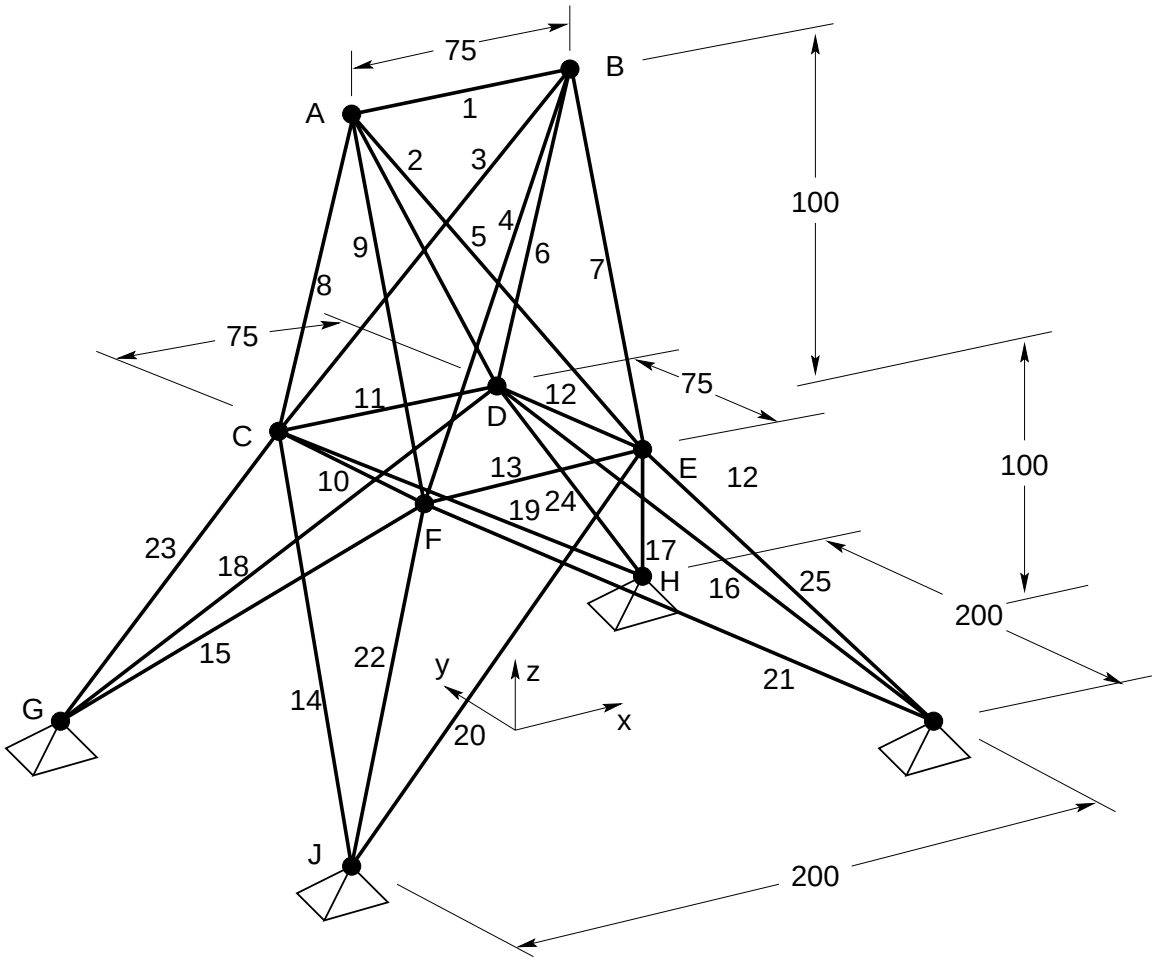

Figure 6.5. Twenty five bar space truss

Table 6.1. Load conditions for the twenty five bar space truss.

	Node	F_x	F_y	F_z
Load Case 1	A	1,000	10,000	-5,000
	B	0	10,000	-5,000
	C	500	0	0
	F	500	0	0
Load Case 2	A	0	-20,000	-5,000
	B	0	20,000	-5,000

psi. in either tension or compression. A plot of the structural weight during the optimization process is shown in Figure 6.6 which, in addition to data from two different initial conditions, also included data from Harless [60] for a continuous optimal solution for the same problem. The final structural weight was arrived at after three analysis iterations and was 98.9 pounds when using stress constraints. This was roughly 8 percent larger than the optimal solution of 91.1325 pounds determined by Harless for the same constraints. As with the 10-bar truss example, a slight increase in structural weight for discrete optimization should be expected. Closer examination of the optimal discrete solution again reveals an average safety

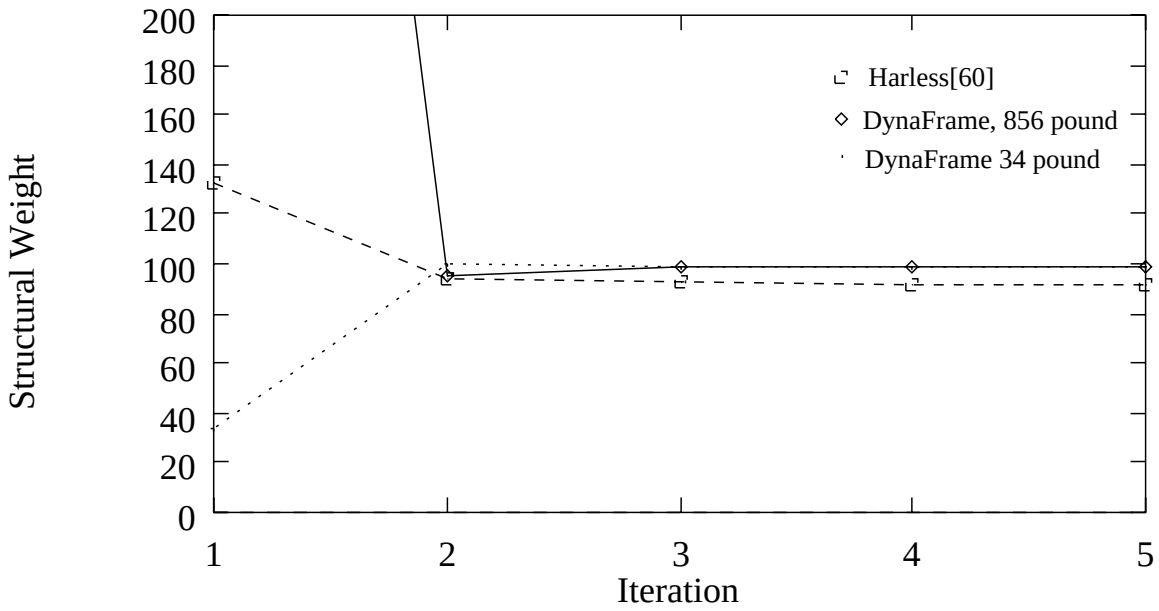

Figure 6.6. Plot of the structural optimization process for the 25-bar truss.

factor of 1.107 or roughly 10 percent greater than for a continuous solution for those members limited by the stress constraint. In this case, many of the structural members were limited by the minimum size constraint that also contributed to the increase in weight, as the best discrete member size was roughly 2 percent over the minimum. The rapid convergence of the member sizes to the optimal solution was also expected as the structure is close to determinate with minimal changes in load paths as the member sizes are modified.

6.5.3 Portal Frame

Although not as common as the truss examples, the portal frame problem described by Lust et al.[87] and found in other papers[147, 146, 149] is perhaps the most common of the frame optimization examples and is shown in Figure 6.7. Similar 2-D examples can be found in the literature[70, 118, 40, 67] but the example described by Lust et al. will be used here. The frame is subject to two load cases,

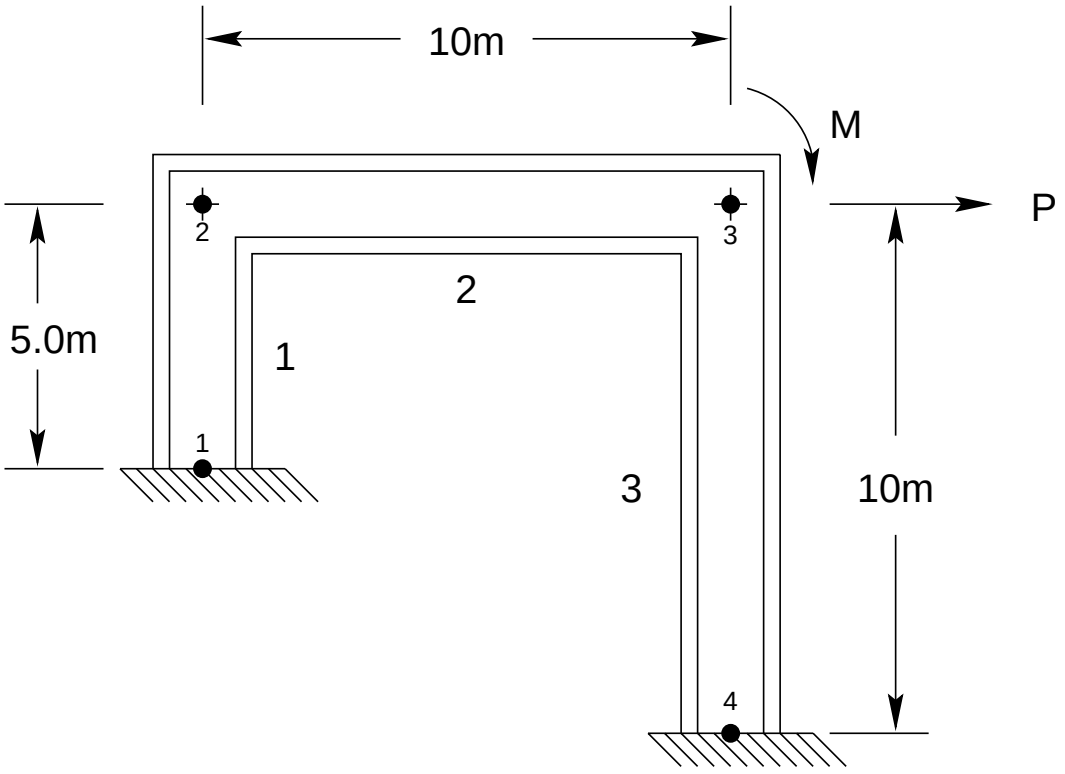

Figure 6.7. Portal frame structure

in the first $M = 0$, $P = 5.0 \times 10^4$ Newtons, in the second case the loads are $M = -2.0 \times 10^5$ Newton-meters. All three members are made from the same material with the properties $E = 7.0 \times 10^{10}$ Newtons/meter^2, $\nu = 0.3$ and all have a symmetric I cross section. Stress constraints are $\sigma_a = 2.0 \times 10^8$ Newtons/meter^2 and $\tau_a = 1.16 \times 10^8$ Newtons/meter^2, translational displacements are limited to ± 0.04 meters, and rotational displacements are limited to ± 0.015 radians.

A graph of the structural weight during the optimization process for three different initial conditions is shown in Figure 6.8. Attention should be paid to the rapid convergence of the structure to the area of the optimal weight. This reflects well on the assumption of reasonable insensitivity of the load distribution to cross sectional properties. For this example, the lightest structure that met the stress and displacement design constraints weighed slightly less than 1100 pounds. Buckling constraints were not included, as the loading conditions did not provide significant compressive loads in any structural members. Direct comparison with the literature results is difficult, as the published results assume a continuous optimization including modification of the cross section parameters and a more

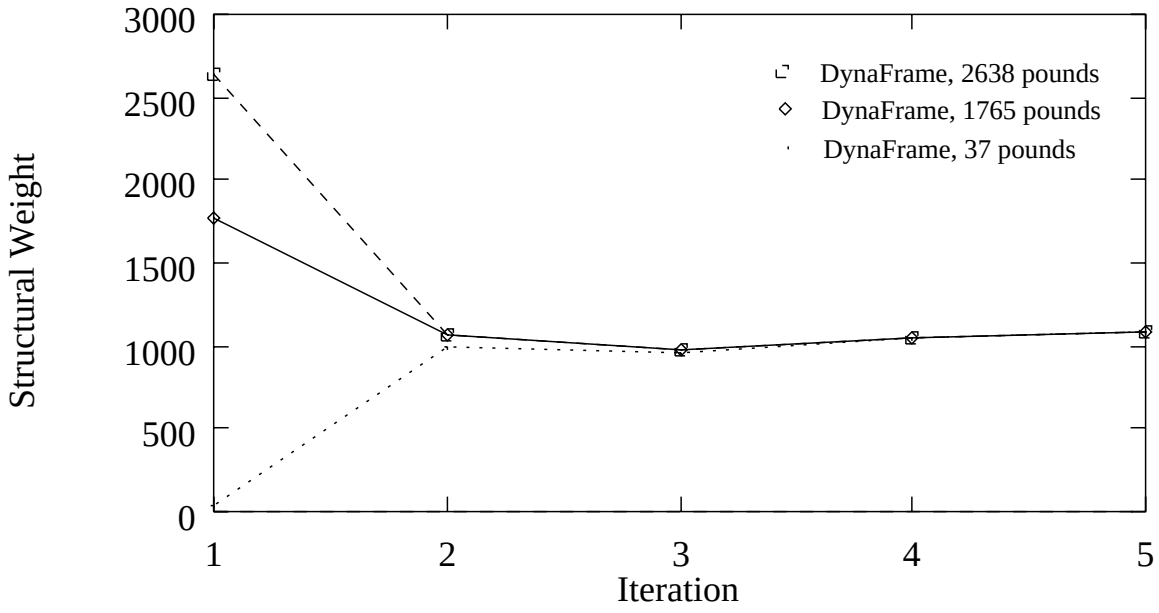

Figure 6.8. Plot of the optimization process for the portal frame from three different initial conditions.

efficient cross section shape. With this in mind, the results published by Lust et al. suggest an optimal structural weight between 581 and 519 pounds, depending upon the optimization technique employed. Better results could be expected from the discrete optimization algorithm as the beam cross section catalog expands in the future, allowing for optimization over a wider range of more efficient cross sections.

CHAPTER 7

RESULTS AND CONCLUSIONS

The principal goal of this project was to demonstrate the potential for extending the parametric modeling paradigm from the detailed design phase to the conceptual design phase over a complex but limited domain. In this we were successful, as the system is capable of synthesizing, analyzing and optimizing a very wide range of dynamic and static structures. Within the relevant domain, the system provides unparalleled assistance and performances for the design engineer. To date, no comparable system has been found which includes either the degree of integration between the geometric model and engineering analysis or approaches the model complexity, and given the difficulties in interfacing and extending most commercial CAD packages, it is unlikely that one exists. This extension could not have occurred without the ability to incorporate and manipulate a wide range of engineering information into the CAD environment. This close coupling of geometric and engineering information provides the foundation necessary to successfully integrate different types of engineering analysis and couple the results to the parametric model.

In looking back over the course of the project, it is estimated that roughly 50 percent of the development went into what can be classified as "support" features. These include both the extensive catalog support as well as the large number and variety of engineering tools. The inclusion of these types of functionality into existing CAD packages would greatly reduce the time and effort needed to develop future systems of comparable performance and complexity. It may be hoped that examples like this project will have a positive effect on future CAD development.

During this project several areas of difficulty were uncovered and investigated

that had previously been unknown or avoided. Although no attempt is made to claim ideal solutions for these problems, first order approximations have been developed and implemented that greatly reduce the level of uncertainty. It is expected that future efforts will address the accuracy and limitations of these approximations and pursue improvements whenever possible. Specific results and contributions are detailed in the following sections.

7.0.1 Integrated CAD systems

The system designed and implemented during this project demonstrates the advantages of tight integration between the CAD model and related engineering analysis. The advantages of an extensible, object oriented environment are readily apparent, both in the substantial reduction in design cycle time as well as the ease of maintaining model integrity. Specific novel features that have been developed and implemented include:

- Construction of a parametric modeling system for complex structures that is valid over a wide range of the "design space."
- The tight integration of the design specifications, including numeric, geometric, and engineering, into a parametric solid model.
- The development of a simple abstract representation useful for both conceptual linkage design, structural synthesis, and structural analysis.
- Inclusion of engineering information into abstract levels, allowing for both preliminary and detailed structural analysis.
- Development and implementation of a second order method for integration of planar NURBS.
- Definition of object representations and algorithms for the description, manipulation, and analysis of a wide variety of catalog components including materials, beam cross sections and linear fluid actuators.
- Definition of object representations and algorithms for description and manipulation of abstract linkages.

Several conclusions can be drawn from the integration effort. First, the inclusion of engineering information and discrete catalog information allows for nonlinear scaling that is usually not possible with conventional parametric modelers. The inclusion of heuristics to extend the valid parametric range allows for a significant increase in model complexity without the limitations normally encountered with parametric modelers. Second, examination of the heuristics reveals a very close interaction between abstract analysis, specifications, and the model geometry. The amount and effectiveness of this coupling would seem to challenge the potential effectiveness of many of the current, top down AI approaches to the design problem, most of which have little to no interaction with model geometry. A third conclusion that can be drawn from observation of this coupling is the necessity of creating an extensible CAD system with wide access to both the geometric data and internal functionality. The required low level of geometric manipulation becomes increasingly difficult as the system developer is isolated from the underlying data and functionality. Again, it may be hoped that projects of this nature will encourage and increase accessibility and extensibility of future CAD packages.

7.0.2 Structural Synthesis

The development and implementation of the structural synthesis represents the highest level functional module created during this project. The functionality and performance of this module is highly dependent upon the proper performance of all underlying modules and would have been impossible to create without this foundation. The general purpose of the structural synthesizer is to rapidly produce reasonable structural models for further analysis and optimization from a minimal set of design specifications. The existing structural synthesizer developed during this project performs this task quite well, although a great deal of improvement would be required before this could be considered sufficiently developed for a commercial product. The following list summarizes the specific advances due to this project:

- Development and implementation of a heuristic system to perform structural synthesis of a wide range of dynamic structures with linear actuators.
- Definition of a set of high level variables sufficient to define and control the structural synthesis.
- Definition of object representations and algorithms for description, manipulation and analysis of link structures.
- Development and implementation of engineering analysis tools for computation of a wide range of abstract information necessary for proper design and functionality of these types of dynamic systems.

The advantages of incorporation of the design specifications into the parametric model are readily apparent. This greatly extends the range of application, in addition to reducing the required input and modeling effort. The relatively small number of high level variables suggests a high level of coupling between various underlying geometric elements. The advantage of providing access and support for this types of functionality into existing CAD packages is apparent.

7.0.3 Structural Analysis

In many ways, the structural analysis portion of this project generated the most surprises. This is not a new field and it was unexpected to discover the quite limited support provided by most available software packages. In particular, the limited input restrictions for dynamic analysis and the complete lack of tools for conversion of dynamic results into a form suitable for static analysis were surprising. Although the solutions arrived at during this project are not ideal, they suggest an alternative approach to the fundamental nature of the design of these types of systems. Some of these novel contributions are listed below:

- Definition of object representations and algorithms for description and manipulation of point loads.
- Tight integration between the widely varying abstract representations required by different functional analysis.

- The development and implementation of velocity trapezoids to bound the kinematic parameters.
- Definition of "worst case" loads sufficient to allow for algorithmic search and automatic determination of load cases.
- Development and implementation of algorithms for decomposition of dynamic loads into a form suitable for static structural analysis.
- Completely algorithmic FEA model generation from the solid model and dynamic analysis results.

A fundamental conclusion that can be drawn from the structural analysis work is the need for much better understanding of the actual structural loading conditions. The velocity trapezoid approach can be successful when corresponding limits are implemented in a servo controller. For systems that are manually operated, other limits need to be provided or the system dynamics well understood to allow reasonable correlation between the analysis and actual system performance. In all cases, the use of velocity trapezoids and corresponding search of the system workspace has a much higher probability of determination of the "worst case" loading conditions than the traditional trajectory approach.

7.0.4 Structural Optimization

The structural optimization algorithm implemented during this project demonstrates the significant performance advantages that can be realized by discrete optimization. The heuristic optimization algorithm realized comparable performance to the mathematical programming approaches without requiring continuous variables and sensitivity analysis. The minimal degree of manual intervention provides both stability as well as insight during the process without seriously affecting optimization performance. Some other features of the system include:

- Discrete selection of beam cross section from a large catalog of common cross sections.
- Selection constraints on both minimal and incremental member size.

- The use of general beam theory for stress approximation during the optimization process.

A conclusion that can be drawn from this work is that for many situations a guaranteed optimal solution is not necessary. For this system, in which both the structural loads and actuator sizes are indirectly coupled with the structural mass, the ability to get close to an optimal solution in a short period of time is much more important than finding the guaranteed optimal solution. Given the small number of weight critical applications, it would be surprising if this were not also true for a majority of engineering applications. A second indirect conclusion that could be drawn is that high optimization performance is less important than other aspects of the optimization problem. It is understood but seldom stated that the difficult part of any optimization process is the determination of an optimal set of metrics. The heuristic approach to this problem may well provide more assistance to design engineers than improved performance of the optimization algorithms.

7.1 Possible Future Work

During the course of this project several potential avenues for future extension have been identified. As with most projects of this nature, the work is rarely completed, as broader scope and higher levels of functionality are always desired. As with the results, these extensions are listed in their relevant sections. Many of these are relatively minor extensions or modifications, but others could provide years of work. When this project was initially undertaken, the principal supporting language of Alpha_1 was a version of Lisp, and as such this project was implemented in Lisp. Due to performance and cross platform support issues, Lisp is no longer supported by the Alpha_1 project and conversion of the system to the current C++ system is the first and major piece of future work. However, much has been learned during the implementation process and the translation process provides an opportunity to implement numerous improvements in object structure and algorithms. Many of these are listed below and others are included in the DynaFrame Users Manual. Still others are part of the evolutionary progress of the Alpha_1 project.

7.1.1 Integrated CAD systems

In many ways, the integration aspects of this project are both the most useful as well as the hardest for which to predict future needs. The CAD and CAE industry are known for a high rate of innovation and improvements, any of which may significantly alter the optimal interface points. The following items assume that the current state of incremental improvements in existing software packages continues in the foreseeable future, not a particularly good assumption. However, as the crystal ball is currently quite foggy, this is the best available approach. Many of the suggestions could potentially fall in multiple categories and as such their location is somewhat arbitrary.

Extend the geometric data exchange options. Currently, exchanging geometric data with other CAD packages for detailed modeling or other purposes is limited to IGES. Unfortunately, much of the higher level information is lost during this process. Extension of the "post processor" analogy to convert existing system models to common CAD formats may provide some relief to this problem and improve the ability to perform detailing in existing commercial CAD packages.

Inclusion of additional engineering tools into the CAD environment. Although this project demonstrates the advantages of close coupling between the CAD and analytical packages, it is widely understood that most designs do not require such a high level of analysis. One of the strengths of this approach was the flexibility provided by incorporation of engineering analysis tools for manipulation of point loads into the CAD environment. This approach could be extended to other tools, allowing simple analysis within the CAD environment and inclusion of the results into the parametric model.

Generalization of the linkage data structure to include additional degrees of freedom. Initially, this could include extensions to support two and three rotational degrees of freedom, with future extensions to translational degrees of freedom and/or closed kinematic chains. This would require considerable ad-

ditional functionality in both the structural synthesizer and structural analysis modules.

7.1.2 Structural Synthesis

The structural synthesizer is the most complex part of this project and, as such, is the least refined. Although it is always possible to achieve reasonable designs by adjusting the various control parameters, it would be helpful if more of the adjustment process could be automated. It should be possible to spend some time and further improve the general synthesizer performance after the system has been translated into C++. Other potential improvements or future lines of inquiry follow:

Extension of the structural synthesis algorithms to include plate structures. Although the space frame and single beam structures currently synthesized are quite useful over a wide range of sizes, there are occasions where an alternative type of structure may be an improvement. One of these types of structures that could be incorporated into the existing system would be those composed of 2-D plates welded together. This could also be extended to include simple bent plates to reduce the amount of welding and related fabrication costs.

Algorithmic selection of bearing and shaft components. Currently, the actual structural joints are roughly modeled as simple male and female pairs with minimal regard to inclusion of the actual bearing and shaft sizes into the CAD model. The addition of these types of components to the solid model would improve the analysis accuracy, as well as eliminating the need for manual sizing and specification of these components.

Modification or integration of improved database functionality. The implemented database functionality for materials, beam cross sections and linear fluid actuators has proved quite useful but the limitations of flat database structures are well known. Adding relational database functionality and providing associated tools for searching and custom database extensions would prove useful for a

wide range of applications.

Extension of the existing actuator placement heuristics to account for dynamic loads. The existing actuator placement tools support placement of the actuator to maximize joint torque anywhere in the joint range of motion. However, there is currently no support for algorithmic modification of actuator placement based upon the dynamic analysis results. It should be possible to keep track of the required actuator torque throughout the joint range of motion and derive and implement heuristics to reposition the actuator for maximum performance.

Extend the structural synthesis heuristics to include higher secondary loads for the multiple branching case. Currently, the heuristics used to synthesize multiple branching structures assume that the secondary branch loads are significantly below those generated from the primary branch. There are cases where this is not a good assumption and additional heuristics could be developed to improve the structures for these cases.

Implementation of algorithms for joint detailing. One of the difficulties with fabrication of the synthesized structures is that the joints can be very complex. One possible solution for this problem is to construct a plate "box" around each joint in which each face is perpendicular to the beam axis. Thus each beam can simply be cut to length and welded in place. This approach has been used successfully in the past, and the algorithms for generating the joint boxes could be updated to work with the existing data structures and analysis systems.

Modification of the objects to better reflect the optimization process. There is currently a great deal of duplication of data between the various objects. This is particularly evident in the link structure and frame objects, in which the material and cross section information is duplicated for each beam member. An improved approach may be to extend the beam object to support multiple centroidal axes and orientation vectors. This would simplify both the aggregate

objects and the optimization code.

Continual extension of the heuristics to improve performance. Despite generally good performance, there are still significant improvements that can be made to the synthesis heuristics. These include better understanding of joint access and a variety of other minor annoyances.

7.1.3 Structural Analysis

The current implementation of the structural analysis tools have proven very successful for the range of problems relevant to this project. Extension of these tools to encompass a wider problem domain could prove quite useful. Despite the claims of commercial software vendors, integration between CAD and the variety of analysis packages is still poor at best. Continuing and extending the approach of building-in the ability to perform simple engineering computation in the CAD package may well provide the fundamental support to overcome many of the existing difficulties. Other more specific suggestions for future work are listed below:

Integration of a commercial dynamic analysis package into the system. For a variety of performance reasons the existing dynamic analysis was implemented in the Alpha_1 system. However, with the improvements in available dynamic analysis packages it may be a better idea to integrate a commercial package using an approach similar to that used with the FEA packages. This would provide analytical support for more complex dynamic analysis such as closed kinematic chains.

Extension of the CAD representations and tools to include closed kinematic systems. The integration of a commercial dynamic analysis package into the system should provide the necessary tools for performing dynamic analysis on closed kinematic chains. However, a great deal of additional work would be required to provide reasonable CAD tools for the construction and manipulation of these types of kinematic systems.

Reexamine the intermediate FEA model specification. The current intermediate FEA model specification was developed in conjunction with a requirement to interface with the Cosmos/M FEA package and, as such, has some residual structure from that particular FEA model description. Extension of the interface to include both the Patran and Ansys analysis packages demonstrates the viability of this approach but a judicious redesign to increase the object nature of the intermediate model specification should prove worthwhile.

Extension of the load object approach to include surface loads and other variable loading descriptions. The Alpha_1 system contains excellent tools for construction and manipulation of NURBS curves and surfaces. This should allow for additional complex load representations that could be algorithmically decomposed to interface with the various FEA packages.

Extension of the algorithmic meshing approach to include surface and solid objects. The reliable algorithmic meshing of complex plate structures remains a difficult problem. The development and implementation of a reasonable set of user-controllable heuristics to this problem may provide many of the same benefits observed during this project.

7.1.4 Structural Optimization

The performance of the structural optimizer demonstrates the sizeable potential benefits from the ability to perform reasonable simple discrete catalog optimization. Extension of this approach to a wider range of catalog items and metrics could provide much the same benefits to a far wider range of similar discrete optimization problems. In particular, the development of heuristics for algorithmic generation of reasonable metrics would be a natural extension of the current work. A list of other, more specific suggestions for possible future work in this area is included below:

Development and implementation of heuristics to account for dynamic sensitivity. Although the performance of the structural optimizer is quite acceptable, it

might be possible to develop a heuristic to approximate the sensitivity of the structural loads to changes in structural inertia. An initial approach would be to perform a static analysis at each critical configuration before optimization, from which an approximate percentage of the loads could be assumed to be from dynamic effects. The existing worse case loads could then be modified by this percentage and multiplied by the percentage change in structural inertia, to arrive at an approximate structural load without requiring additional analysis.

Improved integration of nonstress constraints. Currently the structure optimizer only supports constraints on allowable stress, minimum size, and minimum wall thickness. Buckling analysis is supported but is not integrated into the optimizer. Extending the optimizer to include both buckling and displacement constraints appears to be a natural extension.

Development and implementation of heuristics to reduce the oscillation around the optimal solution. The sharp safety factor cutoff at 1.0 appears to promote some instability early in the optimization process. It may be possible to reduce this by providing a "dead band" around 1.0 in which members would not be resized. This dead band would then shrink to near zero as the optimization process continued.

There are undoubtedly other areas that would benefit from additional effort. The rapid increase in performance of computational hardware is providing the opportunity to develop computational engineering tools that would have been unthinkable just a few years ago. There is no reason not to expect this trend to continue and the field of computer-aided design and engineering to grow and evolve. I believe that this project has assisted that growth process.

REFERENCES

[1] H. Adeli and K. V. Balasubramanyam, *A knowledge-based system for design of bridge trusses*, Journal of Computing in Civil Engineering, 2 (1988), pp. 1–20.

[2] V. Akman, P. J. W. ten Hagen, and T. Tomiyama, *A fundamental and theoretical framework for an intelligent CAD system*, Computer-Aided Design, 22 (1990), pp. 352–367.

[3] American Institute of Steel Construction, *Manual of Steel Construction*, American Institute of Steel Construction, 7 ed., 1970.

[4] G. Anagnostou, E. M. Rønquest, and A. T. Patera, *A computational procedure for part design*, Computer Methods in Applied Mechanics and Engineering, 97 (1992), pp. 33–48.

[5] Anonymous, *Do benefits outweight costs?*, Machine Design, 61 (1989), p. 4.

[6] J. S. Arora, ed., *Guide to Structural Optimization*, ASCE, New York, NY, 1997.

[7] J. S. Arora and A. D. Belegundu, *Structural optimization by mathematical programming methods*, AIAA Journal, 22 (1984), pp. 854–856.

[8] e. B. Cobb, *Alpha_1 User's Manual*, Department of Computer Science, University of Utah, Salt Lake City, UT, March 1992.

[9] J. M. Barthelemy and J. Sobieszczanski-Ssobieski, *Extrapolation of optimum design based upon sensitivity derivatives*, AIAA Journal, 21 (1983), pp. 797–799.

[10] J. Bauer and W. Gutkowski, *Discrete structural optimization: A review*, in First World Congress of Structural and Multidisciplinary Optimization, N. Olhoff and G. I. N. Rozvany, eds., Goslar, Germany, May 28 - June 2 1995, Pergamon, pp. 901–908.

[11] C. Bédard and M. Ravi, *Knowledge-based approach to overall configuration of multi-story office buildings*, Journal of Computing in Civil Engineering, 5 (1991), pp. 336–353.

[12] M. P. Bendsøe, *Optimization of Structural Topology, Shape, and Material*, Springer-Verlag, 1995.

[13] M. P. Bendsøe and N. Kikuchi, *Generating optimum topologies in structural design using a homogenization method*, Computer Methods in Applied Mechanics and Engineering,, 71 (1988), pp. 197–224.

[14] W. A. Bennage and A. K. Dhingra, *Optimization of truss topology using tabu search*, International Journal for Numerical Methods in Engineering, 38 (1995), pp. 4035–4052.

[15] J. A. Bennett, *Topological structural synthesis*, Computers and Structures, 12 (1980), pp. 273–280.

[16] J. D. Biedermann, *Addressing current issues in structural design software*, Journal of Computing in Civil Engineering, 10 (1996), pp. 286–294.

[17] R. J. Boltz and T. J. Avery, *Mechanical design productivity using CAD graphics - a users point of view*, IEEE Computer Graphics Applications, 5 (1985), pp. 40–44.

[18] A. H. Bond and B. Soetarman, *Integrating Prolog and CADAM to produce an intelligent CAD system*, Software–Practice and Experience, 20 (1990), pp. 1049–1076.

[19] G. Booch, *Object-Oriented Analysis and Design with Applications*, Benjamin-Cummings Publishing Company, Inc., Redwood City, CA, 1994.

[20] M. Bremicker, P. Y. Papalambros, and H. T. Loh, *Solution of mixed-discrete structural optimization problems with a new sequential linearization algorithm*, Computers and Structures, 37 (1990), pp. 451–461.

[21] D. C. Brown, *Understanding the nature of design*, IEEE Expert, 12 (1997), pp. 14–16.

[22] K. Brown, *Grammatical design*, IEEE Expert, 12 (1997), pp. 27–33.

[23] A. Cella and R. D. Logcher, *Automated optimal design form discrete components*, Journal of the Structural Division, Proceedings of the American Society of Civil Engineers, 97 (1971), pp. 175–189.

[24] C. Chapman, K. Saitou, and M. Jakiela, *Genetic algorithms as an approach to configuration and topology design*, Journal of Mechanical Design, 116 (1994), pp. 1005–1012. ISSN: 0738-0666.

[25] S. C. Chapra and R. P. Canale, *Numerical Methods for Engineers*, McGraw-Hill, 2 ed., 1988.

[26] A. Chattopadhyay and C. E. Seeley, *A simulated annealing technique for multi-objective optimization of intelligent structures*, Smart Materials and Structures, 3 (1994), pp. 98–106.

[27] J. L. Chen and P. Hajela, *A rule based approach to optimum design*

modeling, Computers and Structures, 33 (1989), pp. 117–128.

[28] J. L. Chen and J. S. Ho, *Direct variational method for sizing design sensitivity analysis of beam and frame structures*, Computers and Structures, 42 (1992), pp. 503–509.

[29] J. M. Chern and W. Prager, *Minimum-weight design of statically determinate trusses subject to multiple constraints*, International Journal of Solids and Structures, 7 (1971), pp. 931–940.

[30] M. Chirehdast, H.-C. Gea, N. Kikuchi, and P. Papalambros, *Structural configuration examples of an integrated optimal design process*, Journal of Mechanical Design, 116 (1994), pp. 997–1004. ISSN: 0738-0666.

[31] K. C. Choi and C. W. Ibbs, *CAD/CAE in construction: Trends, problems and needs*, Journal of Management in Engineering, 6 (1990), pp. 394–415.

[32] R. D. Cook and W. C. Young, *Advanced Mechanics of Materials*, Prentice Hall, 1985.

[33] J. J. Craig, *Introduction to Robotics*, Addison-Wesley, 1989.

[34] M. R. Cutkosky, R. S. Engelmore, R. E. Fikes, M. R. Genesereth, T. R. Gruber, W. S. Mark, J. M. Tenenbaum, and J. C. Weber, *PACT: an experiment in integrating concurrent engineering systems*, Computer, 26 (1993), pp. 28–37.

[35] T. P. Darr and W. P. Birmingham, *Automated design for concurrent engineering*, IEEE Expert, 9 (1994), pp. 35–42.

[36] A. R. Díaz and B. Belding, *On optimum truss layout by a homogenization method*, Journal of Mechanical Design, 115 (1993), pp. 367–373.

[37] Y. Ding, *Shape optimization of structures: A literature survey*, Computers and Structures, 24 (1986), pp. 985–1004.

[38] J. R. Dixon, *Knowledge-based systems for design*, Journal of Mechanical Design, 117 (1995), pp. 11–16.

[39] A. H. B. Duffy, *The "what" and "how" of learning in design*, IEEE Expert, 12 (1997), pp. 71–76.

[40] F. Erbatur and M. M. Al-Hussainy, *Optimum design of frames*, Computers and Structures, 45 (1992), pp. 887–891.

[41] A. G. Erdman, *Computer-aided mechanism design: Now and the future*, Journal of Mechanical Design, 117 (1995), pp. 93–100.

[42] P. A. Fenyes and R. V. Lust, *Displacement approximations for optimization of beams defined in nonprincipal coordinate systems*, AIAA Journal, 30

(1992), pp. 2359–2362.

[43] R. Fish, S. Drake, E. Cohen, and R. F. Riesenfeld, *Feature-based process planning for CNC machining*, Tech. Rep. UUCS-98-001, Department of Computer Science, University of Utah, Salt Lake City, UT, January 1998.

[44] U. Flemming and R. Woodbury, *Software environment to support early phases in building design (SEED): Overview*, Journal of Architectural Engineering, 1 (1995), pp. 147–152.

[45] C. Fleury, *An efficient optimality criteria approach to the minimum weight design of elastic structures*, Computers and Structures, 11 (1980), pp. 163–173.

[46] C. Fleury and M. Geradin, *Optimality criteria and mathematical programming in structural weight optimization*, Computers and Structures, 8 (1978), pp. 7–17.

[47] K. S. Fu, R. C. Gonzalez, and C. S. G. Lee, *Robotics, Control, Sensing, Vision and Intelligence*, McGraw-Hill, 1987.

[48] J. M. Gere and S. P. Timoshenko, *Mechanics of Materials*, PWS Engineering, 2 ed., 1984.

[49] A. K. Goel, *Design, analogy, and creativity*, IEEE Expert, 12 (1997), pp. 62–70.

[50] D. E. Grierson, *An optimality criteria design method for tall steel buildings*, Advances in Engineering Software, 16 (1993), pp. 119–123.

[51] D. E. Grierson and T. C. W. Chiu, *Optimal synthesis of frameworks under multilevel performance constraints*, Computers and Structures, 18 (1984), pp. 889–898.

[52] D. E. Grierson and L. A. Schmit, *Synthesis under service and ultimate performance constraints*, Computers and Structures, 15 (1982), pp. 405–417.

[53] D. E. Grierson, *Conceptual design optimization of structural system*, in Proceedings of the 1996 12th Conference on Analysis and Computation, Chicago, IL, April 1996, ASCE, ASCE New York NY USA, pp. 99–110.

[54] R. Guditz, *An alternative to IGES and PDES: direct data base conversion*, Journal of Engineering Computing and Applications, 4 (1990), pp. 11–16.

[55] W. Gutkowski, *Discrete Structural Optimization*, SpringerWienNewYork, 1997.

[56] W. Gutkowski and J. Bauer, eds., *Discrete Structural Optimization*, Springer-Verlag, 1994.

[57] R. T. Haftka and R. V. Gandhi, *Structural shape optimization–a survey*, Computer Methods in Applied Mechanics and Engineering, 57 (1986), pp. 91–106.

[58] P. Hajela and E. Lee, *Genetic algorithms in truss topology optimization*, International Journal of Solids and Structures, 32 (1995), pp. 3341–3357.

[59] S. R. Hansen and G. N. Vanderplaats, *Approximation method for configuration optimization of trusses*, AIAA Journal, 28 (1990), pp. 161–168.

[60] R. I. Harless, *A method for synthesis of optimal weight structures*, Computers and Structures, 12 (1980), pp. 791–804.

[61] M. Hauser and R. J. Scherer, *Application of intelligent CAD paradigms to preliminary structural design*, Artificial Intelligence in Engineering, 11 (1997), pp. 217–229.

[62] G. A. Hegemier and W. Prager, *On Mitchell trusses*, International Journal of Mechanical Science, 11 (1969), pp. 209–215.

[63] D. A. Hoeltzel and W. H. Chieng, *Knowledge-based approaches for the creative synthesis of mechanisms*, Computer-Aided Design, 22 (1990), pp. 57–67.

[64] H. M. Hua, *Optimization for structures of discrete-size elements*, Computers and Structures, 17 (1983), pp. 327–333.

[65] M. S. Hundal, *A systematic method for developing functions, structures, solutions and concept variants*, Mechanisms and Machine Theory, 25 (1990), pp. 243–256.

[66] W. M. Jenkins, *Towards structural optimization via the genetic algorithm*, Computers and Structures, 40 (1991), pp. 1321–1327.

[67] C. Y. Jung and V. A. Pulmano, *Improved fuzzy linear programming model for structural designs*, Computers and Structures, 58 (1996), pp. 471–477.

[68] T. R. Kane and D. A. Levinson, *Dynamics: Theory and Applications*, McGraw-Hill, 1985.

[69] T. R. Kane, P. W. Likins, and D. A. Levinison, *Spacecraft Dynamics*, McGraw-Hill, 1983.

[70] M. R. Khan, K. D. Willmert, and W. A. Thornton, *An optimality criterion method for large-scale structures*, AIAA Journal, 17 (1979), pp. 753–761.

[71] N. S. Khot, *Nonlinear analysis of optimized structure with constraints on system stability*, AIAA Journal, 21 (1983), pp. 1181–1186.

[72] N. S. Khot, L. Berke, and V. B. Venkayya, *Comparison of optimality criteria algorithms for minimum weight design of structures*, AIAA Journal, 17 (1979), pp. 182–190.

[73] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, *Optimization by simulated annealing*, Science, 220 (1983), pp. 671–680.

[74] U. Kirsch, *Synthesis of structural geometry using approximation concepts*, Computers and Structures, 15 (1982), pp. 305–314.

[75] U. Kirsch, *Structural Optimization: Fundamentals and Applications*, Springer-Verlag, 1993.

[76] U. Kirsch, *Optimal topologies of truss structures*, Computer Methods in Applied Mechanics and Engineering, 72 (1989), pp. 15–28.

[77] A. V. Kumar and D. C. Gossard, *Synthesis of optimal shape and topology of structures*, Journal of Mechanical Design, 118 (1996), pp. 68–74.

[78] S. E. Lander, *Issues in multi-agent design systems*, IEEE Expert, 12 (1997), pp. 18–26.

[79] T. Lassen, *Optimum design of three-dimensional framework structures*, Journal of Structural Engineering, 119 (1993), pp. 713–727.

[80] J. Lee, *Design rationale systems: Understanding the issues*, IEEE Expert, 12 (1997), pp. 78–85.

[81] G. E. Levey and K. C. Fu, *A method in discrete frame optimization and its outlook*, Computers and Structures, 10 (1979), pp. 363–368.

[82] R. Levy and O. E. Lev, *Recent developments in structural optimization*, Journal of Structural Engineering, 113 (1987), pp. 1939–1961.

[83] J. S. Liebman, N. Khachaturian, and V. Chanaratna, *Discrete structural optimization*, Journal of the Structural Division, Proceedings of the American Society of Civil Engineers, 107 (1981), pp. 2177–2197.

[84] J. H. Lin, W. Y. Che, and Y. S. Yu, *Structural optimization on geometrical configuration and element sizing with statical and dynamical constraints*, Computers and Structures, 15 (1982), pp. 507–515.

[85] P. Lu and J. N. Siddall, *A boolean local improvement method for general discrete optimization problems*, Journal of Mechanical Design, 115 (1993), pp. 776–783.

[86] J. Y. S. Luh, M. W. Walker, and R. P. C. Paul, *On-line computational scheme for mechanical manipulators*, Journal of Dynamic Systems, Measurement and Control, 102 (1980), pp. 69–76.

[87] R. V. Lust and L. A. Schmit, *Alternative approximation concepts for space frame synthesis*, AIAA Journal, 24 (1986), pp. 1676–1684.

[88] M. L. Maher, *Expert systems for structural design*, Journal of Computing in Civil Engineering, 1 (1987), pp. 270–283.

[89] M. L. Maher and A. G. de Silva Garza, *Case-based reasoning in design*, IEEE Expert, 12 (1997), pp. 34–41.

[90] J. C. Maxwell, *On reciprocal figures, frames and diagrams of forces*, Transactions of the Royal Society, Edinburgh, 26 (1872), pp. 1–40. also: Scientific Papers, Vol. 2, pp 161-207. University Press, Cambridge, 1890.

[91] A. G. M. Mitchell, *Limits of economy of materials in frame structures*, Philosophical Magazine, 8 (1904), pp. 589–597.

[92] K. S. Mortensen and R. E. Nehring, *The effectiveness of CAD/CAM/CAE systems*, International Journal of Computer Applications in Technology, 2 (1989), pp. 15–19.

[93] S. Naoum and D. Fong, *Proposed computer system for optimizing building design*, in Proceedings of the 2^{nd} Congress on Computing in Civil Engineering, no. 2 in 2, Atlanta, GA, USA, June 1995, ASCE, ASCE New York NY USA, pp. 1561–1568.

[94] N. Olhoff, M. P. Bendsøe, and J. Rasmussen, *On CAD-integrated structural topology and design optimization*, Computer Methods in Applied Mechanics and Engineering, 89 (1991), pp. 259–279.

[95] G. R. Olsen and G. N. Vanderplaats, *Method for nonlinear optimization with discrete design variables*, AIAA Journal, 27 (1987), pp. 1584–1589.

[96] J. V. Owen, *CAD/CAM comes of age*, Manufacturing Engineering, 119 (1997), pp. 48–51, 54, 57–59.

[97] G. Pahl and W. Beitz, *Engineering Design, a systematic approach*, Springer-Verlag, 1993.

[98] M. Papadrakakis, Y. Tsompanakis, E. Hinton, and J. Sienz, *Advanced solution methods in topology optimization and shape sensitivity analysis*, Engineering Computations, 13 (1996), pp. 57–90.

[99] P. Y. Papalambros and M. Chirehdast, *An integrated environment for structural configuration design*, Journal of Engineering Design, 1 (1990), pp. 73–96.

[100] M. Pappas, *Improved methods for large scale structural synthesis*, AIAA Journal, 19 (1981), pp. 1227–1233.

[101] S. N. Patnaik, D. A. Hopkins, and R. Coroneos, *Structural optimiza-*

tion with approximate sensitivities, Computers and Structures, 58 (1996), pp. 407–418.

[102] R. E. Perri, *CAD/CAM productivity*, Design News, (1985), pp. 107–109.

[103] R. W. Pike, *Optimization for Engineering Systems*, Van Nostrand Reinhold, 1986.

[104] W. D. Pilkey, *Formulas for Stress, Strain, and Structural Matrices*, John Wiley & Sons, 1994.

[105] W. Prager and R. T. Shield, *A general theory of optimal plastic design*, Journal of Applied Mechanics, 35 (1967), pp. 184–186.

[106] B. Prasad, *An integrated system for optimal structural synthesis and remodeling*, Computers and Structures, 20 (1985), pp. 827–839.

[107] R. Ramaswamy, K. Ulrich, N. Kishi, and M. Tomikashi, *Solving parametric design problems requiring configuration choices*, Journal of Mechanical Design, 115 (1993), pp. 20–28.

[108] S. S. Rao, *Structural optimization by chance constrained programming techniques*, Computers and Structures, 12 (1980), pp. 777–781.

[109] R. Razain, *Behavior of fully stressed design of structures and its relationship to minimum-weight design*, AIAA Journal, 3 (1965), pp. 2262–2268.

[110] G. M. Reddy and J. Cagan, *An improved shape annealing algorithm for truss topology generation*, Journal of Mechanical Design, 117 (1995), pp. 315–321.

[111] G. M. Reddy and J. Cagan, *Optimally directed truss topology generation using shape annealing*, Journal of Mechanical Design, 117 (1995), pp. 206–209.

[112] K. F. Reinschmidt and G. A. Stone, *Smarter computer-aided design*, IEEE Expert, 9 (1994), pp. 50–55.

[113] G. I. N. Rozvany and C. M. Wang, *On plane Prager-structures-I*, International Journal of Mechanical Science, 25 (1983), pp. 519–527.

[114] G. I. N. Rozvany and C. M. Wang, *On plane Prager-structures-II*, International Journal of Mechanical Science, 25 (1983), pp. 529–541.

[115] G. I. N. Rozvany, C. M. Wang, and M. Dow, *Prager-structures: Archgrids and cable networks of optimal layout*, Computer Methods in Applied Mechanics and Engineering, 31 (1982), pp. 91–113.

[116] W. K. Rule, *Automated truss design by optimized growth*, Journal of Structural Engineering, 120 (1994), pp. 3063–3070.

[117] E. A. Sadek, *Application of methods of feasible directions to structural optimization problems*, Computers and Structures, 17 (1983), pp. 183–191.

[118] M. P. Saka, *Optimal design of rigidly jointed frames*, Computers and Structures, 11 (1980), pp. 411–419.

[119] E. Sandgren, *Nonlinear integer and discrete programming for topological decision making in engineering design*, Journal of Mechanical Design, 112 (1990), pp. 118–122.

[120] E. Sandgren, *Nonlinear integer and discrete programming in mechanical design optimization*, Journal of Mechanical Design, 112 (1990), pp. 223–229.

[121] T. J. Scallan, *CAD framework initiative - a users perspective*, in Proceedings of the 29th ACM/IEEE Design Automation Conference, Anaheim, CA, June 1992, IEEE Piscataway, pp. 672–675.

[122] L. C. Schmidt, *Fully-stressed design of elastic redundant trusses under alternative load systems*, Australian Journal of Applied Science, 9 (1958), pp. 337–348.

[123] L. A. Schmit, *Structural synthesis - its genesis and development*, AIAA Journal, 19 (1981), pp. 1249–1263.

[124] L. A. Schmit and C. Fleury, *Discrete-continuous variable structural synthesis using dual methods*, AIAA Journal, 18 (1980), pp. 1515–1524.

[125] L. A. Schmit and C. Fleury, *Structural synthesis by combining approximation concepts and dual methods*, AIAA Journal, 18 (1980), pp. 1252–1260.

[126] L. K. Seah and K. T. Ooi, *Computer-aided design of an artillery system*, Computers and Structures, 53 (1994), pp. 1023–1031.

[127] D. B. Sieger and R. E. Salmi, *Knowledge representation tool for conceptual development of product designs*, in Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, no. 2 in 5, IEEE, October 1997, pp. 1936–1941.

[128] C. E. Smith, *Applied Mechanics: Dynamics*, John Wiley & Sons, 2 ed., 1982.

[129] C. K. Soh and J. Yang, *Fuzzy controlled genetic algorithm search for shape optimization*, Journal of Computing in Civil Engineering, 10 (1996), pp. 143–150. ISSN 0887-3801.

[130] A. H. Soni, M. H. F. Dado, and Y. Weng, *An automated procedure for intelligent mechanism selection and dimensional synthesis*, Journal of Mechanisms, Transmissions and Automation in Design, 110 (1998), pp. 130–137.

[131] D. Sriram, M. L. Maher, and S. J. Fenves, *Knowledge-based expert systems in structural design*, Computers and Structures, 20 (1985), pp. 1–9.

[132] K. Suzuki and N. Kikuchi, *A homogenization method for shape and topology optimization*, Computer Methods in Applied Mechanics and Engineering, 93 (1991), pp. 291–318.

[133] K. Svanberg, *Optimization of geometry in truss design*, Computer Methods in Applied Mechanics and Engineering, 28 (1981), pp. 63–80.

[134] K. Svanberg, *The method of moving asymptotes-a new method for structural optimization*, International Journal for Numerical Methods in Engineering, 27 (1987), pp. 359–373.

[135] E. I. Tabak and P. M. Wright, *Optimality criteria method for building frames*, Journal of the Structural Division, Proceedings of the American Society of Civil Engineers, 107 (1981), pp. 1327–1342.

[136] J. A. Tárrago, J. Canales, and A. Arias, *CODISYS: an integrated system for optimal structural design*, Computers and Structures, 52 (1994), pp. 1221–1241.

[137] T. Tascione, *Concurrent engineering and electronic data interchange*, in Proceedings of the 1996 12th Conference on Analysis and Computation, Chicago, IL, April 1996, ASCE, ASCE New York NY USA, pp. 397–406.

[138] A. B. Templeman, *Discrete optimal structural design*, Computers and Structures, 30 (1988), pp. 511–518.

[139] P. B. Thanedar and G. N. Vanderplaats, *Survey of discrete variable optimization for structural design*, Journal of Structural Engineering, 121 (1995), pp. 301–306.

[140] H. L. Thomas, A. E. Sepulveda, and L. A. Schmit, *Improved approximations for control augmented structural synthesis*, AIAA Journal, 30 (1992), pp. 171–179.

[141] D. L. Thurston and R. Sun, *Machine learning preferences for structural design*, Microcomputers in Civil Engineering, 9 (1994), pp. 185–197.

[142] A. R. Toakley, *Optimum design using available sections*, Journal of the Structural Division, Proceedings of the American Society of Civil Engineers, 94 (1968), pp. 1219–1241.

[143] C. Tong and A. Gomory, *A knowledge-base computer environment for the conceptual design of small electro-mechanical appliances*, Computer, 26 (1993), pp. 69–71.

[144] B. H. V. Topping, *Shape optimization of skeletal structures: A review*, Journal of Structural Engineering, 109 (1983), pp. 1933–1951.

[145] Y. Umeda and T. Tomiyama, *Functional reasoning in design*, IEEE Expert, 12 (1997), pp. 42–48.

[146] G. N. Vanderplaats, *Thirty years of modern structural optimization*, Advances in Engineering Software, 16 (1993), pp. 81–88.

[147] G. N. Vanderplaats and E. Salajegheh, *New approximation method for stress constraints in structural synthesis*, AIAA Journal, 27 (1989), pp. 352–358.

[148] G. N. Vanderplaats and P. B. Thanedar, *A survey of discrete variable optimization for structural design*, in Proceedings of the 10^{th} Conference on Electronic Computation, April 29-May 1 1991, pp. 173–180.

[149] G. N. Vanderplaats, *Structural design optimization, status and direction*, in AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference, Kissimmee, Fl, April 7-10 1997, pp. 1178–1192.

[150] A. C. Ward and W. P. Seering, *Quantitative inference in a mechanical design 'compiler'*, Journal of Mechanical Design, 115 (1993), pp. 29–35.

[151] E. A. Warman, *Integration revisited–an appraisal of the state of integration of CAD*, Computers in Industry, 14 (1990), pp. 59–65.

[152] A. S. Watson and C. J. Anumba, *The need for an integrated 2D/3D CAD system for structural engineering*, Computers and Structures, 41 (1991), pp. 1175–1182.

[153] M. D. Wehrman, *Productivity improvements through the use of CAD/CAM*, Journal Aircraft, 22 (1985), pp. 1013–1017.

[154] B. Wielinga and G. Schreiber, *Configuration-design problem solving*, IEEE Expert, 12 (1997), pp. 49–56.

[155] D. F. Yates, A. B. Templeman, and T. B. Boffey, *The complexity of procedures for determining minimum weight trusses with discrete member sizes*, Computers and Structures, 18 (1982), pp. 487–495.

[156] N. Yoshida and G. N. Vanderplaats, *Structural optimization using beam elements*, AIAA Journal, 26 (1988), pp. 454–462.

[157] W. J. Zhang and C. A. Luttervelt, *On the support of design process management in integrated design environment*, CIRP Annals - Manufacturing Technology, 44 (1995), pp. 105–108.

[158] W. Zhao and S. Azarm, *A cross-section shape multiplier method for two-level optimum design of frames*, Journal of Mechanical Design, 15 (1993), pp. 132–142.