MORPHING AIRCRAFT STRUCTURES

USING

TENDON ACTUATED COMPLIANT CELLULAR TRUSS

A Thesis in

Aerospace Engineering

by

Deepak Shyamlal Ramrakhyani

© 2005 Deepak Shyamlal Ramrakhyani

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

August 2005
The thesis of Deepak Shyamlal Ramrakhyani was reviewed and approved* by the following:

George A. Lesieutre  
Professor of Aerospace Engineering  
Head of the Department  
Thesis Co-Advisor  
Chair of Committee

Mary I. Frecker  
Associate Professor of Mechanical Engineering  
Thesis Co-Advisor

Farhan Gandhi  
Associate Professor of Aerospace Engineering

James H. Marden  
Professor of Biology

George Lesieutre  
Professor  
Head of the Department of Aerospace Engineering

*Signatures are on file in the Graduate School
The research described in this thesis aims to develop a structural concept capable of achieving continuous stable deformations over a large range of aircraft shapes. The basic concept underlying the approach is a compliant cellular truss, with tendons used as active elements. The members of the truss are connected through compliant joints such that only modest bending moments may be transmitted from one member to another. Actuation is achieved by pulling on one set of cables while controlling the release of another, so that the stability of the structure is maintained in any intermediate position. The tendon-actuated compliant truss can be made to behave locally, and temporarily, as a compliant mechanism, by releasing appropriate cables. As a result, in the absence of aerodynamic forces, the structure can be morphed using relatively low forces.

A parallel genetic algorithm is developed for the design of unit cells using topology optimization. A “fitness” value is assigned to each candidate structural layout and it is a measure of how well the structure meets the design requirements. The fitness function was formulated to include the morphing capabilities on actuation of the cables, as well as the stability of the structure under external loads. The forces in the cables are optimized to obtain the best match between the morphed structure and the required configuration.

A six-noded octahedral unit cell with diagonal tendon actuation was obtained for a bending type deformation of the NASA HECS wing. Initial cell geometry and orientation is determined by “strain matching” to the local morphing deformation required by the application. The cell size is dictated by the available space, the morphing strain, and discretization errors in approximating a smooth desired shape. A finite element analysis is performed on a wing made of these unit cells and sized for a representative vehicle weighing 3000 lbs (1360 kg). The weight of the truss wing (without the skin and actuators) was comparable to a conventional stiffened skin
construction although its deflections are larger. Aeroelastic concerns of flutter and divergence can perhaps be addressed through the use of active control.

Additional cell topologies using fewer cables are also obtained. A procedure is presented to get an estimate of the size of the compliant joint made of pseudoelastic shape memory alloy given the loads. A compliant joint was designed for a smaller vehicle weighing about a kilogram. A skin system is required that can accommodate large shape changes while carrying and transferring aerodynamic loads. The skin could be designed either using high strain-capable materials or sliding skins could be used.

An analysis on the scaling of actuator requirements with aircraft size showed that, if actuation of the aircraft structure involves the morphing of the wing against the lift forces or if the actuators are in the load path, then the weight of the aircraft can be limited to something in the range of a few kilograms to a few thousands of kilograms.

These analyses show that a 3D cellular truss structure can be fairly complex with a large number (18) of cables per unit cell and the area change possible using these structures is limited. Even in-plane actuation of the truss structures involves high actuation forces in the cables resulting in a high actuator weight. This can be addressed pin/compliant joint connected beams to carry the lift loads and the actuating cables then act only against the in-plane loads which are much lower. The genetic algorithm can easily be modified to design such structures. Preliminary results are shown using the algorithm and suggestions are made to improve the results.

This investigation into smoothly morphing aircraft structures has yielded insight into the problem, design procedures and a powerful tool for the design of morphing aircraft structures using cables, truss elements and beams. There are still many challenges to be addressed before they become practical.
TABLE OF CONTENTS

LIST OF FIGURES .................................................................................................................vii

LIST OF TABLES .........................................................................................................................xi

ACKNOWLEDGEMENTS ...........................................................................................................xii

Chapter 1  INTRODUCTION.................................................................................................1

  1.1 Morphing Benefits .....................................................................................................1
  1.2 Morphing Categories ...............................................................................................6
  1.3 Background ..............................................................................................................7
  1.4 Conclusion .................................................................................................................14

Chapter 2  TENDON ACTUATED COMPLIANT CELLULAR TRUSS ....................................15

  2.1 Scaling Considerations ...........................................................................................18
  2.2 Shape of the Unit Cell ............................................................................................23

Chapter 3  THE OPTIMIZATION APPROACH ....................................................................26

  3.1 Topology Optimization ...........................................................................................26
  3.2 The Genetic Algorithm ..........................................................................................28
    3.2.1 Selection ...........................................................................................................28
    3.2.2 Crossover .........................................................................................................31
    3.2.3 Elitism .............................................................................................................32
    3.2.4 Parallel GA ......................................................................................................34
  3.3 Fitness Calculation ....................................................................................................37
    3.3.1 Fitness value involving the actuation of cables ...............................................37
    3.3.2 Fitness value involving displacement due to external loads ...........................39
    3.3.3 Weight consideration.......................................................................................41
  3.4 Results and Discussion .............................................................................................43
    3.4.1 Forcing symmetry in the ground structure ......................................................44
    3.4.2 3D unit cells ....................................................................................................47
    3.4.3 Shape conformation.........................................................................................49

Chapter 4  THE OCTAHEDRAL UNIT CELL .......................................................................53

  4.1 Orientation of the Unit Cell ....................................................................................53
  4.2 Size of Unit Cells .....................................................................................................55
  4.3 Strength and Weight Considerations .........................................................................59
    4.3.1 Representative vehicle and wing ....................................................................59
    4.3.2 Wing loading and member sizing ....................................................................61
    4.3.3 Flexibility and weight .....................................................................................64
    4.3.4 Nonlinear finite element analysis ....................................................................67
  4.4 Conclusions ..............................................................................................................69

Chapter 5  DESIGN USING THE GENETIC ALGORITHM ..................................................70
LIST OF FIGURES

Figure 1-1: Drag and Power versus Velocity, with Morphing.................................................2

Figure 1-2: Flight mission for a military-type aircraft [2] (a) Mission to intercept (b) Mission to penetrate .........................................................................................................................3

Figure 1-3: Variation of Falcon Wing Planform with Speed. [3] ............................................4

Figure 1-5: Airfoils for different speeds a) Subsonic airfoil. b) Transonic airfoil c) Supersonic airfoil ...............................................................................................................................4

Figure 1-6: Typical limitations in a rotorcraft in forward flight .............................................5

Figure 1-7: Bell X-5: The first plane with variable sweep wings .............................................8

Figure 1-8: The F-111(left) in the forward swept mode and the B-1B Lancer (right) in the aft swept mode .........................................................................................................................8

Figure 1-9: The North American XB-70 “Valkyre” ................................................................9

Figure 1-10: Front view of the XB-70 with all three wingtip angles .......................................9

Figure 1-11: The AFTI-F111 Mission Adaptive Wing (MAW) with smooth variable camber wing .....................................................................................................................................10

Figure 1-12: Active Aeroelastic Wing concept.......................................................................12

Figure 1-13: Folding wing concept of Lockheed Martin .......................................................13

Figure 1-14: “Bat wing” concept of NextGen Aeronautics ......................................................13

Figure 2-1: (a) Passive cellular truss structure [26] (b) Proposed unit cell with tendons ......15

Figure 2-2: Frame and Compliant Truss ...............................................................................16

Figure 2-3: Face of a unit cell scaled down 5 times ...............................................................18

Figure 2-4: Cell structure loaded in tension ........................................................................20

Figure 2-5: Morphed and unmorphed configurations of HECS wing ...................................24

Figure 2-6: Cellular truss structure for HECS wing .............................................................25

Figure 3-1: (a) Fully connected Ground structure (b) Reduced ground structure ...........27

Figure 3-2: Probabilities of selection of the sorted chromosomes ........................................30

Figure 3-3: Crossover operation .......................................................................................31
Figure 3-4: Flow chart of the Genetic Algorithm .................................................................33
Figure 3-5: Flowchart showing the algorithm of assigning tasks by the master node .........36
Figure 3-6: A 2D structural layout. .....................................................................................38
Figure 3-7: External forces on the 2D structure.................................................................40
Figure 3-8: Solution of the 2D problem.............................................................................43
Figure 3-9: Convergence plot for the 2D problem.............................................................44
Figure 3-10: A 4X4 cell 2D structure formed by reflection..............................................45
Figure 3-11: Solution to the 2D 4x4 cell structure............................................................45
Figure 3-12: Results of 2D 4x4 cell structure for modified fitness functions.....................46
Figure 3-13: A 3D 2x2 cell ground structure formed by reflection....................................47
Figure 3-14: The 3D structural problem setup..................................................................48
Figure 3-15: The optimized 3D structural solution to the problem in Figure 3-14 ............49
Figure 3-16: The optimized 3D structural solution where the cables are penalized .........50
Figure 3-17: Result for the 3D structure with uniform displacement of the unit cell .........51
Figure 4-1: The central octahedron....................................................................................53
Figure 4-2: Unit cell for top-surface of the HECS wing....................................................54
Figure 4-3: Bench-top demonstration model .................................................................55
Figure 4-4: Unit cell for the bottom surface of HECS wing..............................................55
Figure 4-5: Deformation of the parallelogram of truss members of an octahedron ..........56
Figure 4-6: Large movement in the x and z directions ...................................................56
Figure 4-7: Initial angle required for a given strain.........................................................58
Figure 4-8: Tsch Wing [19] .........................................................................................58
Figure 4-9: A wing made of the octahedral unit cells.....................................................60
Figure 4-10: Cross section of the wing structure .............................................................60
Figure 4-11: Figures illustrating the Schrenk’s approximation. Source: [34]...............62
Figure 4-12: Weight comparison of a truss-wing and a conventional wing at the ultimate load...........................................................................................................................65

Figure 4-13: Deflection of truss wing using nonlinear finite element analysis ..................68

Figure 5-1: Illustration of a “pseudo-mechanism”..............................................................75

Figure 5-2: Deformed geometry of the pseudo-mechanism shown in Figure 5-1 ..........76

Figure 5-3: An optimal unit cell for the modified genetic algorithm...............................77

Figure 5-4: An optimal unit cell for the modified genetic algorithm with a higher weight penalty for the cables ............................................................................................................78

Figure 5-5: 2D wing made of beams..................................................................................80

Figure 5-6: Degrees of freedom for a three-dimensional frame at node j..........................80

Figure 5-7: The ground structure for the 2D beam design of the wing.............................82

Figure 5-8: Required deformed shape of the wing. ..............................................................83

Figure 5-9: Loading on the wing made of beams ..............................................................85

Figure 5-10: Optimal beam solution for the morphing wing .............................................86

Figure 5-11: Actuated configurations of the optimal solution: a) using the first set of cables b) using the second set of cables ..........................................................................................87

Figure 5-12: Beams with end springs .................................................................................88

Figure 5-13: Local degrees of freedom of a beam element .................................................89

Figure 6-1: Deflection and loading on the compliant joint ..................................................95

Figure 6-2: Deflection of the SMA joint............................................................................96

Figure 6-3: Deformation of the beam ...............................................................................98

Figure 6-4: Plot of axial load vs. tip deflection (constant tip angle). ..................................98

Figure 6-5: Variation of moment required to maintain 15° tip angle with increasing axial load and tip deflection.................................................................................................................99

Figure 6-6: Variation of the maximum stress in the beam with changing length and thickness for an axial load of 110 N..........................................................................................100

Figure 6-7: Stress-Strain diagram for ideal pseudoelastic material ..................................101

Figure 6-8: Stress distribution across the thickness of the SMA beam. [53]....................102
Figure 6-9: Variation of bending moment with increasing curvature.............................................103

Figure 6-10: Effective linear modulus of the SMA beam for different thickness values. ......104

Figure 6-11: Variation of the maximum stress in the beam with changing thickness and Young’s Modulus for an axial load of 110 N .................................................................105

Figure 6-12: Feasible thickness values of the SMA beam with maximum stress less than 1150 MPa .........................................................................................................................105

Figure 3-1: Local degrees of freedom of a beam element .........................................................166
LIST OF TABLES

Table 5-1: Fitness values of different optimal solutions.......................................................... 79

Table 7-1: Weight of the morphing plane for reasonable actuator weight............................... 113
ACKNOWLEDGEMENTS

“A hundred times a day, I remind myself, that my inner and outer life are based on the labours of others” -Albert Einstein.

So many people helped things go so smoothly I am sure I won’t be able to mention all of them. I’d like to mention a few my lab mates, who enlivened my work place Mike Grissom, Jie Jang, Eric Salesky, Andrew Kanky, Germain, Mike Thiel, Mike Remally, Brian Zellers, Benoit, Eichi Nishida, Barbara, and especially, Brad Bheil and Lee Gorney who helped me with my computer and the required software. Jamie Brown made the movie and the solid works models of the 3D truss structures. Dr. Gary H Koopman, the director of our lab, always had a loving presence. Though he was not directly involved in my research he would keep enquiring about my progress and always would introduce the guests to the lab and show them my work. I am grateful to my committee members, Dr. Farhan Gandhi and Dr. Jim Marden for their ideas and suggestions, which have helped significantly in improving the quality of my work.

I would like to thank Smita Bharti, my project partner for helping me with genetic algorithms and finite element analysis.

I would like to thank Dr. Paul Plassman for giving me access to the computer cluster. I wouldn’t have been able to complete my thesis without access to those machines. The computation of the 200 cases each using an average of 10 hours of computation time with an average of 20 processors (sometimes up to 140 processors) would not have been possible elsewhere. This thesis would not be the same without access to those machines.

Dr. Murali ‘Lin’nayanur ‘Torvalds’, the linux and MPI expert. The man who knew it all, at least, as far as MPI was concerned. He introduced me to the cool tools in Linux like valgrind, cvs and gdb, and gave me accounts on his machines, so that I could use them to debug my programs. He hasn’t been successful in converting me to a Linux guy. I still use Windows.
Though, most of my useful work is done on Linux, I would still call myself a Windows person. He however, has had one success, he sold me on vim, saying that emacs would give me ‘carpal tunnel’. My advisor however, is in a different class, he is an Apple person

Dr Mary I Frecker, my co-advisor always had insightful and encouraging suggestions in our meetings. She was up to date with the other research in the same field and always had references that helped me a lot in my research. She was also the one who kept track of who went first in our meetings.


My advisor: Dr. George Andrew Lesieutre, My brother: Prakash Shyamlal Ramrakhyani, My parents: Lata Shyamlal Ramrakhyani and Shyamlal Gobindram Ramrakhyani, My Guru: H. H. Sri Sri Ravishankar. “Namo Namah”
Chapter 1

INTRODUCTION

Morphing of an aircraft structure refers to its ability to change its shape in flight in a smooth, stable and controlled manner. The mission profile of a typical aircraft consists, at minimum, of the following phases: take off, climb, cruise, descent and landing. These different phases present different loading and speed conditions and call for different configurations of the wing to maintain optimum performance. The configuration of the aircraft also needs to be changeable for control while maneuvering. In present day aircraft, the configuration change is achieved through additional fixed surfaces (flaps, ailerons, etc) sliding out or rotating about a joint. Morphing presents a smoother and more efficient approach to adapt to the changing conditions. The benefits of morphing are described in greater detail in the following section.

1.1 Morphing Benefits

The area of a wing required for efficient flight varies with speed. This can be shown as follows. The lift of a wing of planform area $S$ and lift coefficient $C_L$ is given by:

$$L = \frac{1}{2} \rho_\infty V_\infty^2 S C_L$$ \hspace{1cm} 1.1

where $\rho_\infty$ and $V_\infty$ are the density and relative velocity of the air to the wing, far away from the wing. The total drag on an aircraft is a combination of profile drag and induced drag. Profile drag is primarily due to skin friction and the induced drag is due to the lift produced by the wing.
In this equation, $e$ is the Oswald’s efficiency factor and $AR$ is the aspect ratio (area/span$^2$) of the wing [1]. In steady, level flight, the lift produced by the wing is equal to the weight of the plane. Induced drag (second term in Eq. 1.2) dominates the total drag on a plane at lower velocities and the profile drag (first term, $C_D$, in Eq. 1.2) dominates at higher velocities. Using these equations the drag ($D$) and power ($D*V_\infty$) curves for three aircraft with different geometric configurations are shown in Figure 1-1.

![Figure 1-1](image.png)

Figure 1-1: Drag and Power versus Velocity, with Morphing

The curves show that efficient flight at lower velocities calls for wing with high aspect ratio and, at higher velocities, for a wing with lower wetted area.

As mentioned before an aircraft mission profile consists of different phases marked by different requirements for the wing. During the takeoff phase, an aircraft requires high lift for low take off speeds and low drag for increased acceleration, so that the take off distance is low. The drag needs to be less to maximize the climb rate, so that the excess power is used for climb. In the cruise phase the drag needs to be minimum for maximum range. During the loiter phase the power required needs to be minimum for maximum endurance. During the landing phase the
wing needs to have a high lift coefficient to be able to land at a lower velocity and a short runway distance. In addition to these, combat aircrafts have other requirements. A fighter aircraft, for instance, requires exceptional maneuver capability along with good subsonic and supersonic cruise performance, Figure 1-2(a), while a bomber, must combine long-range subsonic cruise with low-level high speed dash to penetrate, Figure 1-2(b) [2].

A fixed wing design represents a compromise between these conflicting requirements. Morphing the wing planform would enable efficient operation at any operating speed, from dash to loiter, effectively shifting vehicle power or drag curves with velocity. Birds do this naturally, as illustrated in Figure 1-3. Morphing thus enables the design of true multi-mission aircraft, perhaps having endurance for surveillance and high maneuver and speed capability for attack and dash.
Apart from the wing area, the wing profile or efficient airfoil shape is also different at different speeds. At lower speeds, the airfoil shapes are thicker and more cambered to increase the lift coefficient ($C_l$) (Figure 1-4). At higher speeds, thinner airfoils are preferable, especially at transonic speeds, the airfoil shape is critical as the air reaches supersonic speeds, at some points on the airfoil forming shock waves. Transonic airfoils are designed such that the pressure over most of the top surface is almost constant. At super-sonic speeds ($1<M<5$) the airfoil shapes are slender with sharp edges where as at hypersonic speeds ($M>5$) a blunt profile is preferred due to aerodynamic heating [1].

Using discrete control surfaces, as in typical aircraft, also increases the radar signature of an aircraft. A continuously morphing wing would also improve stealth performance apart from reducing drag.
In addition to fixed-wing aircraft, rotary-wing aircraft can also benefit from the morphing of the rotor blades. The maximum speed in forward flight in a helicopter is limited due to rotor blade stall on the retreating blade and supersonic drag (wave drag) on the advancing blade [4] as shown in Figure 1-6. Morphing would enable changing the airfoil during the advancing and retreating phases, thus increasing efficiency. The most efficient diameter of the rotor, like the span of a wing, also varies with varying forward flight speeds. Large diameters are better for hovering but smaller diameters are better for higher speeds in forward flight [5].

![Figure 1-6: Typical limitations in a rotorcraft in forward flight.](image)

While the morphing of an aircraft shows significant benefits, the scope and limitations of morphing technology are yet unknown. But increasing advances in material sciences, aerodynamics and computations make significant morphing capability seem more achievable.
1.2 Morphing Categories

Having seen the benefits of morphing, we shall now have a look at the different aspects of morphing. Morphing of an aircraft structure can be either active or passive. When the shape of the aircraft structure is changed by internal actuators, as is done by muscles in birds, the morphing is said to be active. And when the shape of the aircraft structure changes due to the aerodynamic forces, like the morphing of insect wings, the morphing is called passive or aeroelastic morphing.

Another aspect of morphing in nature suggests a hierarchical approach to morphing. That is, the means for achieving a spectrum of shape control range from coarsely-controlled gross limb positioning (wing area, span, and sweep), to moderately-controlled digit positioning (wing chord, thickness, twist, and camber), to finely-controlled accurate local shape maintenance and flow control. Each level in the hierarchy involves different geometric scales (in terms of the size of the structure, the magnitude of its motion, and the accuracy of positioning), different bandwidths (large, slow motion to fast, small motion), and different uses of global and local information.

The highest level of morphing involves large-scale shape changes, as would be required for control of wing span, area and sweep. Variable wing span is a very challenging problem. However, the DG505 Orion sailplane can vary its wingspan by roughly a factor of two, and there has been research into variable-span helicopter rotor blades [5,6]. A key aspect of this problem for large geometry changes is the use of a telescoping joint, the linear analog of the rotary joint used for variable wing sweep. A continuously-morphing solution to this problem, however, has not been developed to date.

Variable wing sweep is already satisfactorily accomplished by several modern aircraft, using rotary joints and hydraulic actuation. This approach is congruent with the articulated
skeletons of birds. A continuously-morphing solution to this problem has not been developed to date.

The next level of morphing involves chord- or thickness-scale shape changes, as would be required to implement variable wing chord, thickness, twist, and camber. In general, modern air vehicles use discrete control surfaces, such as ailerons, elevators and rudders, for vehicle attitude control, including force trim. The ability to morph at this scale is essential to provide such control capability plus additional attitude control functions, such as the ability to fly without a tail, over a large range of speeds. Also importantly, it would provide the additional ability to implement “power trim” or “drag trim” functions for a vehicle.

Wing chord control could be used to modify the lift produced by a wing, especially at low speed. This is commonly accomplished today through the use of discrete control surfaces like Fowler flaps. Chord control could also be used to move the vehicle center of pressure as needed for tailless flight; this can also be accomplished with sweep. Wing chord control by continuous deformation of a structure represents a novel capability.

Wing thickness control could also be an important factor in a power trim system, in combination with other morphing changes, especially to reduce transonic drag. It may be complicated somewhat by the typical desire to use at least some of the volume in the wing for fuel or payload.

1.3 Background

Most of the research in aircraft structural morphing has been in the mid-scale region. In fact the first morphing technique was used as early as the first powered flight. The Wright brothers achieved controlled flight by pulling on cables to twist the wing. However, as the speed of aircraft increased, the wings became stiffer and discrete control surfaces were used instead for
control and to adapt to varying flight conditions. Variable-sweep using hinged wings was used to reduce drag at higher Mach numbers. The Mach number that the airfoil sees decreases as the wings are swept back and hence the drag due to the compressibility effects (shocks) decreases.

An experimental aircraft, the Bell X-5, shown in Figure 1-7, flight tested from 1952-1955, was the first aircraft capable of sweeping its wings in flight. Since then other military aircraft such as the F-111, F-14 Tomcat and the B1-B Lancer (Figure 1-8) have been built with the ability to sweep back their wings for efficiency during high-speed flight. Now, variable sweep is also being considered for a supersonic business jet [7].

---

**Figure 1-7**: Bell X-5: The first plane with variable sweep wings

---

**Figure 1-8**: The F-111(left) in the forward swept mode and the B1-B Lancer (right) in the aft swept mode
The North American XB-70 “Valkyre” (Figure 1-9), with cruise speed of 2000 mph (Mach 3.0 at 72,000 ft), was a research aircraft for the advanced study of aerodynamics, propulsion, and other subjects related to large supersonic aircraft. It was designed to make use of a phenomenon called "compression lift," achieved when the shock wave generated by the airplane flying at supersonic speeds supports part of the airplane's weight. In flight, the XB-70 could lower the outer wing sections either 25 degrees for flying from 300 knots to Mach 1.4, or a severe 65 degrees for speeds from Mach 1.4 to Mach 3+ (Figure 1-10). Measuring just a bit over 20 feet at the trailing edge, these wingtips are the largest movable aerodynamic device ever used. Lowering the wingtips had three distinct effects on the XB-70: 1) the total vertical area was increased, allowing shorter vertical stabilizers than would otherwise be needed; 2) the reduction in rearward wing area countered the delta wing's inherent rearward shift of the center of lift as speed increased, keeping drag-inducing trim corrections to a minimum and 3) compression lift was 30 percent more effective because the pressure under the wing was better managed.

Figure 1-9: The North American XB-70 “Valkyre”

Figure 1-10: Front view of the XB-70 with all three wingtip angles
Several researchers considered smoothly varying the camber instead of using discrete control surfaces. Having a continuous surface lowers the drag and increases the stall angle of attack of the airfoil. The Advanced Fighter Technology Integration (AFTI) F-111 (Figure 1-11) was a joint Air Force/NASA/Boeing venture to develop smooth, variable-camber wing technology [8]. Six independent trailing-edge flap segments (three per wing) and two leading-edge flap segments provided smooth continuously-variable camber using flexible fiberglass skins on the upper surface and sliding panels on the lower surface. An extensive wind-tunnel and flight-test program was performed. Significant drag reductions were observed at high lift coefficients due to the variable-camber concept. Spanwise lift distribution could also be changed to move the center of lift inboard to reduce the wing root bending moments.

Figure 1-11: The AFTI-F111 Mission Adaptive Wing (MAW) with smooth variable camber wing.

During the mid-1990s and later, researchers in Germany looked at improving the transonic wings of civilian aircraft, to actively adapt to varying parameters such as altitude, Mach number and aircraft weight. Their morphing goals were to achieve spanwise differential camber, and shock reduction using an adaptive contour (“bump”) in the shock area [9,10]. The following advantages were expected: higher aerodynamic efficiency due to optimized L/D, higher
maximum L/D, reduced wing root loads, and improved performance at extended transonic conditions.

At about the same time, a team led by the Northrop-Grumman Corporation in the US was addressing smoothly varying camber under the DARPA/ARFL/NASA Smart Wing program [11]. A flexcore elastomeric skin trailing edge structure, actuated using high power piezoelectric ultrasonic motors, was developed as a hingeless trailing edge control surface. Significant benefits were observed in increased maneuverability and reduced drag at off-design cruise speed in transonic flight, accomplished by using morphing leading and trailing edge surfaces that could take on smooth shapes that are not readily obtained with conventional control surfaces.

Another concept that has been studied is varying wing twist to increase roll performance. A semi-passive morphing wing, the Active Aeroelastic Wing (AAW) was pioneered and advanced by Rockwell (now Boeing), and later supported by the USAF Wright Laboratory and the NASA Langley Research Center. This concept used a wing, flexible in torsion, and leading and trailing edge control surfaces to induce wing twist (Figure 1-12). High performance roll rates were achieved [12]. Another approach that was used to control the wing twist was using a variable stiffness spar (VSS), which controlled the wing torsional stiffness [13]. The VSS also significantly enhanced roll performance, but without multiple control surfaces. The team at Northrop Grumman obtained a spanwise wing twist of 5° using an SMA torque tube [14]. This was done at a much slower rate to increase lift during takeoff and landings. Siclari, et. al., at the Grumman Corporate Research Center studied an active rib structure to reduce fuel expenditure during transonic cruise [15]. Active truss elements in the rib structure were used to change the shape of the airfoil. Various optimized airfoil shapes were obtained to reduce drag under transonic cruise conditions. Significant drag reductions of up to 36% were observed by reducing the thickness at higher Mach numbers.
So far, most research has addressed the mid-scale region of morphing, viz., variable camber, twist, and in the fine-scale region of local shape control, like adaptive contour changes and oscillating surfaces to delay flow separation. Approaches to achieve smooth large-scale shape changes have only been investigated recently, especially at NASA [16]. Aircraft structures that are more flexible have also been considered for micro air vehicles. Studies developed wing shapes that were inspired from bird wings and sharks fins. With minor in-flight wing tip adjustments, yaw, pitch and roll control are expected to be accomplished [17].

Telescoping wings have also been used to change the area and aspect ratio of wings. For example, there has been research in variable-span rotor blades [20]. In addition, Blondeau, et al. designed and developed a telescopic wing with nylon fabric skins and pneumatic actuation of the spar. The aspect ratio of the telescopic wing could increase by 130% while supporting aerodynamic loads. The telescoping wing had a lower lift/drag ratio when compared to a rigid fixed-wing specimen, due to an increase in parasitic drag due to discontinuous seams [21]. Another telescopic wing design can be found in the patent filed by Gevers Aircraft, Inc (Gevers, 1998). The wingspan of this design can change by 100%. The final aircraft is supposed to cruise over 280 mph when the wing is retracted and stall at a speed as low as 63 mph when the wing is

Figure 1-12: Active Aeroelastic Wing concept
extended. The telescopic wing used in this proposed aircraft is composed of a central wing section with completely retractable lift sections which are guided on rollers in a spanwise direction.

More recently in the late 1990s and the early 2000s a lot of interest was generated in morphing aircraft structures, fueled by a new research program, the Morphing Aircraft Structures program. This program was funded by DARPA/NASA, for the development of novel aircraft and structures capable of achieving 50 percent change in wing area and 200 percent change in aspect ratio without much increase in weight of the aircraft [22]. There were three main contractors and several university-based groups in phase one of the program. Lockheed Martin is developing a small Unmanned Air Vehicle (UAV) with folding wings for an air force mission (Figure 1-13); Raytheon Missile Systems is improving on the Navy Tomahawk cruise missile by adding telescoping wings; NextGen Aeronautics in California, has the most challenging and radical design goal of developing a UAV which can change the aspect ratio of a wing, its area and sweep by continuous morphing of the wing (Figure 1-14).

Figure 1-13: Folding wing concept of Lockheed Martin

Figure 1-14: “Bat wing” concept of NextGen Aeronautics.
The design of multi-mission vehicles using morphing aircraft structures is another area of research currently being pursued. When compared to fixed wing aircraft both the requirements and the design variables have increased, and hence traditional design methods that rely solely on historical data and legacy codes are not viable options. Ordaz et al., at Georgia Tech and Crossley et al., at Purdue university are developing design methodologies using optimization tools [23, 24]. Ordaz is using a physics based response surface methodology to find the optimum parameters of a particular configuration of aircraft. The approach used by Crossley is more general, where he uses morphing as an independent variable, assigning a certain weight penalty for certain kinds of morphing like area morphing and then optimizing to obtain the different parameters.

1.4 Conclusion

Morphing of aircraft structures is potentially the next big step in aviation technology. Morphing enables the contemplation of true multi-mission vehicles. Important capabilities made possible by morphing include efficient flight over a broad range of flight speeds and effective high-rate maneuvering. For efficient flight, the main needs are large changes in wing area (span and chord), along with sweep and thickness. For flight control, the required capability involves rapid modifications of local camber and twist. The development of a structural concept capable of changing shape while carrying loads is a challenge. Most of the designs so far have used traditional methods like rotating and sliding joints. This thesis investigates a new concept for morphing aircraft structures.
Chapter 2

TENDON ACTUATED COMPLIANT CELLULAR TRUSS

Morphing of aircraft structures offers many benefits but poses many challenges. A morphing aircraft structure needs to be able to change its shape, carry the required loads and be light. So far designers have been able to accomplish only two of these three goals simultaneously [25]. Modern aircraft wings are highly weight-efficient but are stiff and cannot be morphed. The required change in shape is achieved using articulated or sliding joints and additional surfaces. With respect to morphing this type of construction is not as efficient as it might be.

The basic concept underlying the present approach is a compliant cellular truss with tendons used as active elements. Depending on the application, the cell structures used could be two-dimensional or three-dimensional. A passive cellular truss and a candidate unit cell for a tendon-actuated cellular truss are shown in Figure 2-1. The cables might be activated using electric motors or, perhaps, piezo inchworm motors, which are currently under development.

![Figure 2-1: (a) Passive cellular truss structure [26] (b) Proposed unit cell with tendons](image)

The truss members are connected through compliant joints. A compliant truss differs from an ideal truss in that modest bending moments may be transmitted from one member to another through the joints. However, such a truss differs from a welded frame in that effective
member bending stiffnesses are substantially reduced in the vicinity of a joint, as illustrated in Figure 2-2. In a sense, such joints are analogous to flexures in 2D structures, and act as “rotational springs” at the end of each element.

One possible realization of a compliant joint involves a shape memory alloy (SMA) cylinder or wire. Used passively in a pseudoelastic or “superelastic” mode, SMA can be regarded as a high-strain-capable structural material. To operate in this mode, the temperature must be above the transition temperature of the material. Depending on the temperature, strains approaching 8% may be experienced reversibly.

Cables or tendons cannot carry compressive loads. A truss structure involving cables might be stiff in tension under one loading condition, but might collapse under an opposite load condition. The truss and unit cell should be designed so that they are stable in any load condition. And if they are designed to be stable even without pretension in the cables, different cables are active in different load conditions. A key feature of this tendon-actuated compliant truss is that, by releasing appropriate cables, the structure can be made to behave locally, and only temporarily, as a compliant mechanism, i.e., the structure can change shape without storing much strain energy. As a result, in the absence of aerodynamic forces, the structure can be morphed using relatively low forces. This is not possible with conventional structures, as the (highly redundant) stiff structure absorbs significant strain energy when forced into any shape other than the nominal one [27]. Furthermore, the use of compliant joints rather than true rotating joints eliminates binding as a significant concern. Highly-distributed actuation also enables the
achievement of global shape changes as the accumulation of local ones. The cables are reeled in or released in a controlled manner while the structure is loaded; hence, the stability of the structure is maintained in any intermediate position.

Passive trusses are known to be efficient structures from a weight standpoint. They have been considered in the design of, for example, the High Altitude Long Endurance (HALE) aircraft [28]. Wicks and Hutchinson have found that the strength of optimized truss plates in bending, compression and shear compares favorably to honeycomb core sandwich plates and stiffened panels [29]. And tendons can be made of very light and strong materials. One possible realization of cables involves high-specific-strength polymeric materials such as Spectra® (polyethylene) or Vectran® (polyester-polyarylate). Vectran®, for instance, exhibits specific strength more than ten times that of steel, while its specific modulus is twice that of steel.

Truss structures involving cables have so far been used in relatively low-loading conditions like deployable space structures [30-32]. Use of tendon-actuated compliant cellular trusses in aircraft structures potentially presents a novel method for achieving a wide range of morphing shapes in aircraft wings without accruing an excessive weight penalty. Some of the promising key features of a cable-actuated truss structure are: 1) truss structures are light; 2) smooth changes in the shape can be achieved since large changes in the structure take place as the accumulation of small local changes; 3) truss structures can provide a dense support for a skin; and 4) cables have a large actuation capability, since, in principle the cables can be reeled in by any amount.

The design of truss structures for morphing applications is pursued by addressing the following issues.

Shape of the unit cell: If the structure is made of repeated cells. What should the shape of the unit cell be? Which members should be cables? What degree/level of morphing is achievable? Is a cellular approach the best?
Strength and weight: Can the truss structure be made strong enough to carry the loads? Is it efficient from a weight standpoint and comparable to existing passive structures?

Scaling: How does the stiffness and strength of the truss structure vary with the size of the unit cell?

The effect of scaling on the strength and stiffness of the truss structure addressed in greater detail in this chapter. The shape of the unit cell is found using topology optimization method and this is discussed further in chapter 3. A wing structure built of the resulting unit cells is analyzed for strength and stiffness considerations. These results are discussed in chapter 4. Preliminary studies will also be made on the design of the compliant joint as well as the issues regarding the skin which are discussed in chapter 6.

2.1 Scaling Considerations

The effect of decreasing the size of the unit cell, as shown in Figure 2-3, on the stiffness and strength of a truss structure is studied in this section. The shape of a unit cell and the connectivity truss members in the unit cell is assumed to remain the same after scaling as shown in the figure below.

---

![Figure 2-3: Face of a unit cell scaled down 5 times](image)
The scale factor, $n$, is defined as the ratio of the length of the original cell to the scaled cell. If a unit cell is scaled down $n$ times, the total number of cells that can fit in the same volume increases by $n^3$ times. The unit cell shown in Figure 2-3 has been scaled down 5 times (the two diagrams shown in figure have the same length). A tensile or compressive load acts on a unit cell as shown in Figure 2-4. The load need not be aligned with a direction of symmetry or a principal direction of the cell. The load carried by each member of the unit cell depends only on the geometry (shape and connectivity) of the unit cell in a statically determinate. And, in a statically indeterminate unit cell the load carried by each member depends on the geometry and the relative areas of the members. By assumption, the geometry (shape and connectivity) of the cell doesn’t change with scaling and the relative cross-sectional areas also remain the same. Hence, the ratio of load carried by any member of the unit cell to that carried by any other member of the same unit cell doesn’t change with scale for a simple loading in tension or compression.

The elongation $U$ of the unit cell under the load $F$ and, in the direction of $F$, can be written as

$$U = \alpha \frac{L}{EA} F$$

where $L$ is the length of the unit cell, in the load direction $A$ is the characteristic cross-sectional area of a member, $E$ is the Young’s modulus of the material of the truss member, $\alpha$ is a parameter that depends on the geometry (shape and connectivity) and the relative cross-sectional areas of the members of the unit cell and is independent of the size of the unit cell.

Consider a unit cell shown in Figure 2-4 and another cell structure of the same size made up of scaled unit cells scaled down $n$ times. There would be $n^3$ scaled unit cells in the cell structure. The same load is applied on both the structures and the deflections are compared as shown below.
The volume fraction and density, $m$, of the cell is related to the characteristic cross-sectional area of a member and length as follows:

$$m \propto \frac{AL}{L^3} = \frac{A}{L^2} \quad \text{(2.2)}$$

The length of each scaled unit cell is given by

$$l = \frac{L}{n} \quad \text{(2.3)}$$

If the volume fraction remains the same, the members of the scaled unit cell will have a cross-sectional area given by

$$a = \frac{A}{n^2} \quad \text{(2.4)}$$

The surface area of the unit cell decreases by $n^2$. On the face 1234, in Figure 2-4, there would be $n^2$ scaled unit cells occupying the same area as one of the original unit cells. Hence the force acting on each unit cell is given by

$$f = \frac{F}{n^2} \quad \text{(2.5)}$$
The length of the scaled unit cell as mentioned before decreases by $1/n$. Hence there would be $n$ scaled unit cells along the side 1-8 in the figure. If the deflection of each of the scaled unit cells was $u$, the deflection, $U'$ of the cell structure made up of the unit cells is given by

$$U' = n \cdot u$$  \hspace{1cm} (2.6)

Since the cell shape geometry and relative cross-sectional areas do not change after scaling the deflection $u$, of the scaled unit cell can be written in terms of the length, $l$, the characteristic cross-sectional area of its members, $a$, and Young’s modulus, $E$, as shown in Eq. 2.1. Equations 2.3, 2.4 and 2.5 are used to relate the scaled variables to the original unit cell as follows

$$U' = n \cdot \frac{l}{AE} f = n \cdot \frac{L}{AE} \cdot \frac{n^2}{n^2} \cdot \frac{F}{n^2}$$  \hspace{1cm} (2.7)

$$U' = \frac{L}{AE} F = U$$

The deflection of the cell structure made of the scaled unit cells is same as of the original unit cell. Hence the stiffness of a cell structure doesn’t change with scaling if the geometry and density are kept constant.

The stress in the members of the original unit cell is proportional to the applied force divided by the characteristic cross-sectional area.

$$\sigma = \alpha \cdot \frac{F}{A} = \frac{n^2 f}{n^2 a} = \frac{f}{a}$$  \hspace{1cm} (2.8)

where $\alpha'$, like $\alpha$, is a function of geometry (shape and connectivity) and relative cross-sectional areas of the members of the unit cell and independent of size (scaling). Hence the stress in a scaled unit cell can be written similarly as:

$$\sigma_{sc} = \alpha' \cdot \frac{f}{a}$$  \hspace{1cm} (2.9)
Again using equations 2.4 and 2.5 to relate the scaled quantities to the original quantities we get:

$$\sigma_{sc} = \alpha' \frac{F}{n^2} \frac{n^2}{A} = \alpha' \frac{F}{A} = \sigma$$ \hspace{1cm} 2.10

The stress in the members is not affected by scaling. Hence the strength of the structure, when considering failure of a member reaching the ultimate yield strength, does not vary with scaling.

The strength of a structure, if it fails by local member buckling under compressive load, does not depend on scaling either. This can be shown as follows. The buckling load of a member of the scaled unit cell is given by

$$P_{cr} = \frac{\pi^2 EI_{sc}}{l^2}$$ \hspace{1cm} 2.11

where $I_{sc}$ is the area-moment of inertia of the scaled member. Since the geometry and relative area of the members of the unit cell does not change, the maximum compressive load that can be carried by the unit cell is be written as,

$$f_{cr} = \beta P_{cr} = \frac{\pi^2 EI_{sc}}{l^2}$$ \hspace{1cm} 2.12

where $\beta$ is some constant of proportionality, which depends on the geometry and relative cross-sectional areas of the members of the unit cell and is independent of the scale. The total load carried by the scaled cell structure is $n^2$ times the load carried by a single unit cell (Eq. 2.5),

$$n^2 f_{cr} = \beta n^2 \frac{\pi^2 EI_{sc}}{l^2} = \beta n^2 \frac{\pi^2 E I}{n^4 L^2} = \beta \frac{\pi^2 EL}{L^2} = \beta P_{cr} = F_{cr}$$ \hspace{1cm} 2.13

$P_{cr}$ is the buckling load of a member of the original unit cell and $F_{cr}$ is the maximum compressive load carried by the original unit cell.
So far we have seen that scaling doesn’t affect the stiffness of a truss structure in tension or compression nor does it affect the strength due to yielding or buckling of its members. The structure can also fail due to global buckling. The calculation of global buckling load is more complicated and the effect of scaling on global buckling load has not been calculated.

2.2 Shape of the Unit Cell

The Hyper Elliptic Cambered Span (HECS) wing shown in Fig. 3, and developed at the NASA Langley Research Center, provided a baseline geometry to guide the initial development of this concept. The HECS wing originated as a biologically-inspired configuration that reduces induced drag [17]. Yaw, pitch, and roll control are accomplished through continuous wing morphing, especially at locations near the tip. The development of the truss structure design is approached in two ways.

In the HECS wing, the primary mode of deformation is spanwise bending, with some twist near the tip. Figure 2-5 shows the HECS wing in both the unmorphed and morphed configurations. The morphed and unmorphed configurations, as defined by sets of 3D points on the surface, were provided by NASA LaRC [18]. The data for a finer mesh of points was obtained by interpolation. The approximate strain, defined as the change in length due to morphing over the original length, along the chordwise, spanwise and thickness directions in the wing, were calculated from the interpolated data as follows:

The coordinates, in the original and deformed configurations, of the point at which the strains were to be calculated and three neighboring points along the span, chord and thickness, were obtained from the interpolated data. The lengths of the three lines connecting the point of interest and the three neighboring points were calculated in the original and deformed configurations. The strains in these three directions were calculated as the change in length of the
three lines divided by the un-deformed length. The corresponding shear strains were calculated in a similar manner by calculating the change in angles between the three lines from the original to deformed configurations.

The ratio of the chordwise to the spanwise strains varies over the surface. Over most of the top surface, this ratio is positive, suggesting the use of a “material” with an effective negative in-plane Poisson’s ratio. Over most of the bottom surface the ratio is negative. The normal and shear strains through the thickness were much smaller than the surface strains.

An effective way of accomplishing the required deformation is to use a different layer of unit cells for each of the top and bottom surfaces as shown in Figure 2-6. In the figure, the projections of the corresponding unit cells on the top and bottom surfaces are shown. The topology orientation and configuration of each unit cell is determined by the local morphing strain requirements.
A topology optimization approach is used to design the shape of the unit cells. The development of the approach is described in the next chapter.

Figure 2-6: Cellular truss structure for HECS wing
Chapter 3

THE OPTIMIZATION APPROACH

The unit cells are designed for the HECS wing using a topology optimization approach. A parallel Genetic Algorithm (GA) is used to solve the problem. The following sections will describe this approach in details.

3.1 Topology Optimization

Topology optimization is an automated approach to find the optimal layout of a structure that fits the design requirements. In this approach the structure is assigned a “fitness” value based on how well it fits the design requirements. Then the layout is varied using an optimizing technique to find the best solution. There are three aspects of topology optimization: 1) the way in which the structural layout is formulated and varied; 2) the way in which the “fitness” value is assigned to a candidate structural layout; and 3) the optimization technique used.

In the current approach a “ground structure” method is used to formulate the structural layout. An initially chosen layout of nodal points in the truss structure is taken. Each of these nodes can be connected: to every other node in the structure forming a “fully-connected” ground structure (Figure 3-1 (a)), or just the neighboring nodes, forming a “reduced” ground structure (Figure 3-1 (b)). The members can be one of three kinds of elements: a cable, a truss element or a void. The total number of variables in the problem is equal to the total number of members. As these members take different values we get different structural layouts.
The next step is to find out how good each structural layout is, considering the design requirements. In the current case we want a structure that deforms easily into a particular shape when the cables are actuated but is stiff when the cables are locked. Depending on how well the structure performs, it is given a “fitness” value. There can also be some constraints imposed on the structural layout and, if the structure violates these constraints, it is labeled infeasible. The calculation of fitness and feasibility of the current structure is discussed in section 3.3.

An optimization algorithm can now be used to find a set of parameters that will give a good fitness value. In this research, a genetic algorithm is chosen as an optimization technique. The reasons for this choice are as follows:

1) The problem is formulated in such a way that the variables have discrete integer values and the genetic algorithm is well-suited for handling discrete variables.

2) The fitness function does not vary in a continuous way with the variables. It has an arbitrary dependence, and the genetic algorithm is fairly robust when non-differentiable, multimodal, noisy, and otherwise unconventional functions are involved.

The following section describes the details of the genetic algorithm implemented. The algorithm described in this section was adapted from a serial Genetic algorithm written by Bharti [46].
3.2 The Genetic Algorithm

The genetic algorithm adopts the evolutionary process in nature to optimize the process at hand. It involves a mathematical formulation of the genetic reproduction process and is based on the survival of the fittest strategy.

The problem is formulated as follows. Each of the members of the ground structure is represented by a variable which takes one of the values 0, 1, 2, etc., if it is a void, truss member, cable, etc. These variables are concatenated to form a “chromosome.” Each chromosome is a sequence of numbers with \( n \) digits, where \( n \) is the total number of variables. The variable can also be represented in binary format, in this case the length of the chromosome is \( n \times m \) bits where \( m \) is the length of each variable in binary format. Hence each chromosome represents a unique structural layout.

In the first step of the genetic algorithm, an initial population of chromosomes is generated randomly. The fitness and feasibility of each of these designs represented by chromosomes is calculated. The chromosomes are added to the initial population only if they are feasible. The size of the population is one of the parameters that determine the convergence characteristics of the genetic algorithm. If a large population size is used, there is a greater probability of finding the global optimum, however the time taken to converge increases as the population size increases. This initial population forms a “parent” mating pool from which the “child” chromosomes are generated using processes of selection, crossover and mutation.

3.2.1 Selection

Two chromosomes are selected from the mating pool using a rank based selection. The chromosomes in the mating pool are ranked according to the fitness with the chromosome with
the best fitness value getting the highest rank, equal to the population size and the chromosome with the lowest fitness value is given rank 1. Then a roulette-wheel type selection procedure is used, where the probability of a parent being selected is proportional to its rank. The probability of a parent being selected is calculated as shown in Eq. 3.1

\[
\text{Probability of a chromosome}(i) \text{ being selected } \propto \text{rank}(i) \\
\Rightarrow P(C_i) = \text{rank}(i) \times \text{constant} \\
\sum P(C_i) = 1 \Rightarrow \left( \sum \text{rank}(i) \right) \times \text{constant} = 1 \\
\text{constant} = \frac{1}{\sum \text{rank}(i)} \\
P(C_i) = \frac{\text{rank}(i)}{\sum \text{rank}(i)}
\]

A roulette-wheel is implemented as follows. A random number, r, between 0 and 1, is generated. This number represents the position of the needle between 0° and 360°. The sum of all the probabilities of a chromosome being selected is 1. The chromosomes are ordered according to rank. The position of a chromosome with rank \( i \), is after \( (i-1) \) chromosomes. And this chromosome is selected if the random number \( r \) is greater than the sum of the probabilities of selection of \( (i-1) \) chromosomes and is less than the sum of the probabilities of selection of \( (i-1) \) chromosomes as shown in Eq. 3.2. A parent chromosome with a higher rank has a larger piece of the pie as shown in Figure 3-2.

\[
\sum_{i=1}^{\text{populationSize}} j < r \leq \sum_{i=1}^{\text{populationSize}} j
\]
Another selection procedure that is sometimes used is the proportional selection method, where the probability of a chromosome being selected is proportional to its fitness. In the proportional selection method, if there is a chromosome with a relatively large fitness compared to the rest of the chromosomes, then its probability of being selected will be high and hence it will dominate the population. The algorithm might converge prematurely even though the current solution is not the optimum. There are two factors that control the speed at which a GA converges. One is the selective pressure which pushes the solution towards the best chromosome, and the other factor increases diversity in the group, so that a broader search space is covered. Processes like crossover and mutation increase the diversity in the group. Another aspect that increases diversity is increasing the population size. Increasing diversity helps in exploring a broader search space; however it takes more time to converge to a solution. On the other hand, increasing selective pressure helps in honing down to the best solution. These two factors have to be balanced to get the best solution in the available computation time. Some of the parameters affecting these two factors will be discussed here. These are discussed in great length, in the Handbook of Evolutionary Computation by Back, et al. [47].
### 3.2.2 Crossover

Having selected two parents, two children are produced from these two parents using a two point crossover. Two numbers are randomly generated in the range of 1 to (the population size-1), which determine the crossover points. The digits between these crossover points are interchanged as illustrated in Figure 3-3 to produce two child chromosomes. Another way of performing the crossover operation is to chose a single point for crossover, and interchange the genes before and after this point. The two point crossover produces more diversity in the population.

![Crossover Operation Diagram](image3-3.png)

**Figure 3-3: Crossover operation**

Another way of increasing diversity in the population is the mutation process. In this process each digit of the child is randomly changed to any of the other possibilities with a very small probability, mu (~ 0.01-0.1). As mu is increased the diversity increases in the algorithm. The effect of changing mu was studied and will be discussed in the results section.

Once a child is generated, its fitness is calculated and it is added to the child pool if it is feasible. If the total number of chromosomes in the child pool has reached the population size, then the child chromosomes of the current generation are made the parents for the next generation. The process is repeated for a predetermined number of generations.
3.2.3 Elitism

In the above algorithm if the entire parent population is replaced by the child population, then each generation has a new set of chromosomes. One problem with this procedure is that, if in one generation a chromosome was found which had the optimum layout, then it is lost in the next generation. Though the entire population improves on average, the optimum is not reached. Hence a concept of elitism is used wherein a fixed number of best chromosomes in the parent population are retained for the next generation, by directly adding them to the child population. As the number of chromosomes to be directly added to the child population is increased, the selection pressure increases and can lead to premature convergence. The flowchart for the GA is shown in Figure 3-4.
Figure 3-4: Flow chart of the Genetic Algorithm
3.2.4 Parallel GA

The serial GA is parallelized to increase the computation speed. The main time lag in the entire algorithm is in the calculation of the fitness function for each chromosome. Hence, the calculation of the fitness function is split up among different processors. The calculation of the fitness of a chromosome is independent of the calculation of fitness of other chromosomes. So this algorithm can be easily parallelized. The parallelization is done in a master-slave process. A master node does the GA calculations, while the slave nodes calculate the fitness and feasibility of the chromosomes. The master node generates chromosomes either randomly or using selection, crossover, and mutation. It then sends these chromosomes to the slave nodes for calculation. The calculation time is not the same for all the processors since some of the processors return early if the chromosome is not feasible.

The code for the GA was written in C++ and the parallelization was done using the Message Passing Interface (MPI) library. In order to make the best use of all the available processors, a processor which has returned a chromosome is immediately given a new chromosome. This is done using an asynchronous non-blocking command in MPI, `MPI_Irecv`. The algorithm is as follows: In each generation the chromosomes are first sent to all the available slave nodes, immediately implementing a non-blocking receive for each node. Once chromosomes have been sent to all the processors, the master node waits to receive data from any slave processor. As it receives the fitness and feasibility from a particular slave processor, it adds the corresponding chromosome to the child pool or initial pool, as the case might be, if the chromosome is feasible. If the number of chromosomes in the pool is less than the population limit, then it sends another chromosome to the same slave node and implements a non-blocking receive. Once the mating pool is full, there might still be some workers returning the fitness and feasibility values for the chromosomes they have been working on. So the master node receives
all the fitness and feasibility data from these nodes without adding them to the pool. Hence every
generation additional $np-2$ chromosomes are evaluated, where $np$ is the number of processors.
The algorithm of the master node dealing with the assigning of tasks is shown in Figure 3-5.
Figure 3-5: Flowchart showing the algorithm of assigning tasks by the master node.


3.3 Fitness Calculation

The fitness of a structural layout represents how well a particular design meets the design requirements. In general a unit cell should deform easily on the application of actuation forces, i.e., when the cables are actuated and should be stiff when the external forces are applied (when the cables are locked). The fitness function can also include other aspects, such as the desire that a unit cell should deform into a required shape and that it should have fewer cables. These will be discussed later in this section. First deflection due to the actuation of the cables is discussed.

3.3.1 Fitness value involving the actuation of cables

Linear finite element analysis is used to find the deflection due to the actuation of cables. A stiffness matrix \( K \) is formed relating the forces \( F \) at the nodes to the displacement \( U \) of the nodes Eq. 3.3. The structure is supported so as to prevent any rigid-body motion. Hence the displacement vector can be solved for, given the forcing on the structure.

\[
F = K \cdot U
\]

Since truss elements are used in the structure each node has two or three degrees of freedom in a 2-dimensional (2D) or 3-dimensional (3D) structure respectively. A 2D structural layout shown in Figure 3-6 is used to illustrate the formulation. The size of the displacement vector is 2*number of nodes – the number of constrained degrees of freedom due to the supports (2*9 – 4 = 14). There are four types of members in any structural layout: 1) truss members, shown in black in the figure, which carry axial tension and compression; 2) cables, shown in red, which morph the structure in a particular direction; 3) antagonistic cables, shown in blue, which morph the structure in the opposite direction; and 4) voids, shown by dashed lines, which essentially indicate that no real member is present. The void elements are assigned a small stiffness value compared
to the other members. It is not set to zero since it could lead to the stiffness matrix, $K$, being singular (indefinite displacement) in some cases. As mentioned before, one set of the cables morphs the structure to a required shape when actuated and another set of cables “un-morphs” the structure or brings the structure back to its original shape when actuated. A cable is assumed to be actuated by an actuator placed at the node. When the first set of cables is actuated, the second set of cables is released. So, essentially, it is a structure of truss members that is being morphed. And ideally, the structure of truss members would form a mechanism so that it can be deformed without storing much strain energy in the structure, except in the compliant joints. In order to capture this effect the stiffnesses of both cables (blue and red) are set to small values during the actuation stage. An actuated cable is represented by applying small forces at the ends of the cable as shown in Figure 3-6. The magnitudes of the forces are such that they produce deflections of the order of the length of the member when applied on a void element. The global force vector ($F$) is calculated taking into account all the cable forces. Eq. 3.3 is solved to find the displacement vector ($U$) of the nodes. The displacement of the nodes in the direction of interest, shown by the black arrow in the current example, is added to the fitness function. The first set of cables is actuated and the displacements are added to the fitness function. When the second set of cables is actuated the displacements are subtracted from the fitness function.

---

Figure 3-6: A 2D structural layout.
So the fitness function now addresses the maximization of displacement of a single point in a single direction, using equal forces in all the actuated cables. The fitness due to actuation is formulated follows.

\[ U_1 = + U^d(output\ node) \text{ when the first set of cables is actuated} \]
\[ U_2 = - U^d(output\ node) \text{ when the second set of cables is actuated} \]

\[ (fitness)_a = | U_1 + U_2 | \tag{3.4} \]

The cables need to be antagonistic, \textit{i.e.}, when the second set of cables is actuated then the displacement of the output point should be in the opposite direction compared to that obtained when the first set is actuated. If they are in the same direction then the structural layout is infeasible, and the feasibility variable of the structural layout is set to ‘0’. Ideally, the magnitude of displacements in both directions should also be the same. However, all the cables in the above case are actuated using the same load, so there is room for improvement if the displacements in the two directions are nearly equal. So even if the displacements in the two directions differ by a factor of 3, the structure is said to be feasible.

### 3.3.2 Fitness value involving displacement due to external loads

The structure needs to be stiff when the external aerodynamic loads are applied. One way to measure this is by using \textit{compliance}, which is the work done by external forces on a structure as it deforms, since, this is equal to the energy stored in the structure [48]. It is given by the product of force \((F_{ex})\) and the displacement of the point of application in the direction of the force \((U^f)\) (Eq. 3.5).

\[ \text{compliance} = \frac{1}{2} F_{ex} \cdot U^f \tag{3.5} \]
External forces can be applied to a unit cell only at the points where the unit cell is connected to other unit cells as shown in Figure 3-7. These points are chosen by the designer. The unit cell needs to be stable under any combination of the external loads. Hence mutually independent unit loads are applied at each point and the compliance is calculated individually. If cables are involved the stability of the structure needs to be checked in both directions. A cable might be in tension and provide stiffness between the connecting points if the load is applied in one direction and might become slack for a loading in the opposite direction. Hence, the compliance is calculated in each case. The problem is solved 12 times for the structure shown in Figure 3-7, with 3 points of contact and 2 mutually independent directions at each point and 2 cases (+ve and –ve) for each direction. A unit magnitude of force is applied in each case. Since linear analysis is used, the deflection under any combination of the loads would be a linear combination of the corresponding displacements. The net compliance is written as the summation of all the individual compliances (Eq. 3.6).

\[
\text{compliance} = \frac{1}{2} \sum_{i=1}^{2n\text{App}} F_{\text{ex}} \cdot U_i^F = \frac{1}{2} F_{\text{ex}} \sum_{i=1}^{2n\text{App}} U_i^F \tag{3.6}
\]

Figure 3-7: External forces on the 2D structure.
The finite element analysis of the structure during the application of the external force is done assuming the cables are locked, *i.e.*, they have a fixed length and provide stiffness to the structure when extended. However cables go slack under compressive load. During the finite element analysis, the stiffness of any cable having a compressive load is set to a small value to effectively eliminate such cables, and the analysis is repeated. This process is continued until the same cables are set to void, as set in the previous iteration [36].

The cell structure should be able to deform easily, *i.e.*, the fitness value due to the actuation should be high, and should be stiff when the external loads are applied so the compliance should be low. One way to use a single fitness function to accommodate these contradictory requirements is by taking a weighted sum as shown in Eq. 3.7

$$fitness = \alpha \cdot (fitness) - (1 - \alpha) \cdot compliance$$

3.7

In some cases one of these values could be really large, overshadowing the other and could dominate the result. In order to avoid this, a ratio of the two values is taken as shown in Eq. 3.8.

$$fitness = \frac{(fitness)}{compliance}$$

3.8

### 3.3.3 Weight consideration

Weight considerations are also included in the fitness evaluation. A unit cell having fewer cables and truss members is less complex and is therefore preferable to one having more cables and truss members. This is addressed in two ways, both as a constraint and in the fitness function. A constraint is set on the volume fraction of trusses \(V_{f_{truss}}\) given by the following equation. Eq. 3.9
In this equation, $volume_{Truss}$ is the volume of truss members in the current structural layout, $totalVolume$ is the volume of all the members in the initial ground structure assuming they are truss elements. Incidentally, $totalVolume$ volume does not change even if all the members are cables, since the areas used for the truss and the cable elements is the same. The stiffness, however, is different and is set assuming the truss members are made of aluminum and the cables are made of Spectra® [38]. A constraint is also set on the volume fraction of cables ($Vf_{cable}$), which is calculated in a similar way.

Another way of handling the weight considerations is to decrease the fitness function as more material is used. The fitness function was divided by the sum of cables and trusses, with more weight given to the cable elements.

\[
Vf_{truss} = \frac{volume_{Truss}}{totalVolume}
\]

In the genetic algorithm, any chromosome violating the constraint limits is termed infeasible and is discarded. This way of handling constraints, where anything outside the boundary is eliminated, is rigid. If the feasible regions are small then it is difficult to find an initial set of feasible solutions and, if there are islands of feasible regions in the search space then it is difficult to traverse or jump in between these different islands as the algorithm progresses. However, if constraints are not used then the size of the search space is very large. And the solution could take a large time to converge. Hence, constraints with a reasonably large margin are chosen and then the weight requirements are included in the fitness function so that the weight is reduced, as

\[
fitness = \frac{fitness}{(Vf_{cable} + 0.5Vf_{truss})^{1.5}}
\]
shown in Eq. 3.10. Cables are penalized more in the fitness function since the objective was to obtain a truss with fewer cables.

### 3.4 Results and Discussion

The 2D problem shown in Figures 3-6 and 3-7 was analyzed using the parallel GA algorithm. Another constraint was added to limit the displacement of the output point, orthogonal to the required direction. If the orthogonal displacement was greater than ten percent of the displacement in the required direction then the structure was labeled infeasible. The optimal solution obtained is shown in Figure 3-8. The problem was also solved by including the orthogonal displacement in the fitness function. The fitness function was divided by the orthogonal displacement.

![Figure 3-8: Solution of the 2D problem](image)

The program was re-run a few times with different start seeds for the random number generator, which essentially gives different chromosomes for the initial population, the same solution was obtained each time. There are 20 elements in the initial ground structure. The number of possible structural layouts, if there are 4 kinds of elements, is $4^{20}$ ($1.0995\times10^{12}$). The algorithm used a population of 250 chromosomes and calculated for 1500 generations and
analyzed 1.98e6 structural layouts. Even with such a small sample size the algorithm found the optimum solution. This shows that the GA is robust in finding the optimum solution. The plot of the fitness function with the number of generations for one run is shown in Figure 3-9.

![Convergence plot for the 2D problem](image)

Figure 3-9: Convergence plot for the 2D problem

### 3.4.1 Forcing symmetry in the ground structure

The algorithm converges for this simple problem but as the number of members in the ground structure increases, the number of possibilities increases exponentially. It is possible to reduce the number of possibilities, if the structure is forced to be symmetric. If symmetry is enforced in the 2D case about both the x and y axes then the members in only one-fourth of the structure need to be changed. The other members of the structure are found by reflection. A 4x4 cell 2D structure found by reflecting the members in the 2x2 cell is shown in Figure 3-10. There are 72 members in the structure but only 20 of them are independent. Hence, the total number of possibilities remains the same as in the previous case. However if reflection was used for the
previous case then there would be just six independent variables and the total number of possibilities reduced to a mere 4096. The previous problem of 2x2 grid structure was solved using reflection and yielded the same result.

Once the complete structure is obtained from reflection, the structure is solved in the same way as before. The 4X4 cell problem was solved for the same loading case and output point. The result from the genetic algorithm is shown in Figure 3-11. This solution is similar to that of the previous problem, but the diamonds are smaller. Increasing the fineness of the ground structure did not really change the solution.
The formulation of the fitness function determines the result obtained. For example, if the fitness function was modified to penalize the weight of the trusses and cables more shown in Eq. 3.11 and 3.12, the optimal solutions have fewer members, but might not be as stiff. The optimal solutions to the problems using these fitness functions are shown in Figure 3-12(a) and (b). However, in the 2D case both these results are similar to the previous solution; in fact, the second solution is almost the same as the one shown in Figure 3-8.

\[
\text{fitness} = \frac{V_f}{(V_{f_{\text{cable}}} + 0.5V_{f_{\text{truss}}})^2} \quad \text{(3.11)}
\]

\[
\text{fitness} = \frac{V_f}{(V_{f_{\text{cable}}} + 0.5V_{f_{\text{truss}}})^3} \quad \text{(3.12)}
\]

These results are all optimal solutions associated with different fitness functions. The fitness functions are different only in the nature of the weights given to the different requirements. For example, in the results in Figure 3-12, more emphasis is given to obtaining a light weight structure. These solutions are said to be lying on the Pareto optimal curve [49]. This 2D structure is analyzed primarily to formulate and study the optimization algorithm. The real interest is in 3D structures to be used in the design of wings, as described in chapter 2.
3.4.2 3D unit cells

A 2x2x2 cell 3D ground structure is formed by reflection as shown in Figure 3-13. The boundary conditions, and the output point indicated by the blue arrow, are shown in Figure 3-14. The external forces are shown by red arrows. The structure is connected to the neighboring structures at 9 points marked by red stars. Forces are applied only at 4 of these points, shown by the red arrows, because the structure is symmetric. If the structure can carry each of these loads individually in both directions then it can carry any combination of these loads or the loads applied at any of the other symmetrical positions. During the forcing cycle the structure is analyzed 24 times for the 12 forcing cases in both positive and negative directions.

Figure 3-13: A 3D 2x2 cell ground structure formed by reflection.
The program was first run for the 3D case with the following fitness function, where both the trusses and cables were penalized equally.

$$fitness = \frac{fitness}{(Vf_{cable} + Vf_{truss})^{1.5}}$$ \hspace{1cm} \text{(3.13)}$$

The result can be seen in the movie file in Figure 3-15 where the two sets of cables are actuated turn by turn. The unit cell expands in the z direction and compresses in the y direction. It has an effective Poisson’s ratio of 1 in the y-z plane and 0 in the x-z plane. The Poisson’s ratio in the y-z plane can be tailored as described in the previous chapter. The fitness function was changed to penalize the cables more than the truss elements as follows Eq. 3.14.

$$fitness = \frac{fitness}{(Vf_{cable} + Vf_{truss})Vf_{cable}}$$ \hspace{1cm} \text{(3.14)}$$

Figure 3-14: The 3D structural problem setup.
The result of optimization using this fitness function is shown in Figure 3-16. Here the displacement at the output point is associated with a shear mode of deformation in the x-z plane. There is no deformation in the y direction. The number of cables is far fewer than in the previous case.

### 3.4.3 Shape conformation

The displacement of a single point is maximized in both these cases. The central point deflects more than the corner points. When the unit cells are joined to form a structure this can lead to incompatibility between adjacent nodes.
Figure 3-16: The optimized 3D structural solution where the cables are penalized

To overcome this problem, the entire face or all the 5 points should deflect together. If the mean of the displacements of the 5 points was used instead of the single output point in the fitness function, the algorithm would maximize the displacements of all the 5 points. In order to capture the effect of all the points moving together, the fitness function was divided by the variance of the displacements of the 5 points. The contribution of the “displacement due to actuation” to the fitness function is shown in Eq. 3.15. The variance is divided by the mean ($\mu$) to non-dimensionalize the denominator. If all the points move together, then the variance is 0 and the fitness function would become a very large value even if the mean displacement was not very large. In order to prevent this, a small number is added to the non-dimensional variance in the denominator.
The above equation was substituted in Eq. 3.8 to find the fitness. The result from the optimization algorithm using this fitness function is shown in Figure 3-17. This result is essentially an octahedral unit cell. The algorithm was run six times using different sets of initial values. The solution shown below was obtained three times among the six times and had the best fitness value among the six solutions.

\[
\mu = \frac{\sum_{i=1}^{5} U^d (\text{output point}_i)}{5}, \quad \sigma = \sqrt{\frac{\sum_{i=1}^{5} (U^d - \mu)^2}{5}}
\]

\[
(fitness)_{a} = (fitness)_{a} + \frac{\mu}{0.01 + \frac{\sigma}{|\mu|}} \quad \text{when the first set of cables is actuated}
\]

\[
(fitness)_{a} = (fitness)_{a} - \frac{\mu}{0.01 + \frac{\sigma}{|\mu|}} \quad \text{when the second set of cables is actuated}
\]

Figure 3-17: Result for the 3D structure with uniform displacement of the unit cell.
The octahedral unit cell can also be obtained by reasoning from the basic unit cell shapes [50]. Since the same result is obtained from the genetic algorithm, it confirms that the octahedral unit cell is an optimum configuration for the current design requirement. Other unit cells are developed in chapter 5 using the genetic algorithm. The octahedral unit cell is used to build a wing structure shown as shown in Figure 2-6. The details of the construction and the strength and weight considerations are discussed in the next chapter.
Chapter 4

THE OCTAHEDRAL UNIT CELL

The design obtained by using the topology optimization approach was essentially an octahedral unit cell with central tendons. The octahedron can be easily seen in Figure 4-1. The peripheral members in the figure are drawn in lighter lines for clarity. The unit cell shown in Figure 4-1 would be the same as shown in Figure 3-17 if it were rotated about the y-axis by 90°. The central cable lying in the same plane as the other cables has also been removed.

4.1 Orientation of the Unit Cell

The orientation of the unit cell depends on the strain characteristic of the location it is placed. As proposed in chapter 2, the HECS wing structure is made using a layer of unit cells each for the top surface and the bottom surface. The orientation of the basic octahedral unit cell used for the top surface is shown in Figure 4-2. In the figure, the thicker blue lines represent truss...
members and the lighter red lines represent cables. By assumption, the cables can be actuated independently. If the cable connecting nodes 5 and 6 is shortened and the remaining cables are lengthened simultaneously, then points 2 and 4, as well as points 1 and 3, will move further apart. Here the strains in the x and y direction increase or decrease simultaneously when the cables are actuated. The ratio of these strains can be changed by varying the lengths of truss members in the yz plane and the xz plane. The truss members of the octahedron shown in Figure 4-2, in the yz plane are shorter than the truss members in the xz plane. When the cables are actuated the strain of the unit cell in the y direction will be more than the strain of the unit cell in the x direction. These unit cells could be aligned along the directions of the principal strain in the top surface or they could be aligned along the spanwise and chordwise directions and the in plane cables could be actuated differently to match the shear deformations. A bench-top demonstration model of the top-surface unit cells is shown in Figure 4-3. The in-plane cables in the model have been removed and compliant joints are designed to allow movement in a plane, for ease of construction.

Figure 4-2: Unit cell for top-surface of the HECS wing
The orientation of the unit cell used for the bottom surface is shown in Figure 4-4. In this case, as the strain in the y direction decreases the strain in x direction increases and vice-versa.

4.2 Size of Unit Cells

The size of the unit cell in the three directions can be limited by the morphing strain required and the available space. Consider the parallelogram formed by the truss members in the x-z plane of Figure 4-2. Two configurations of this parallelogram are shown in Figure 4-5: the solid lines show the initial configuration, and the dashed lines show the configuration after...
deformation. The expansive or compressive strains in the unit cell are due to the change in lengths of the diagonals of the parallelogram in the plane.

![Diagram of parallelogram deformation]

**Figure 4-5**: Deformation of the parallelogram of truss members of an octahedron

In this deformation, the length of the truss members is fixed. Depending on the initial slope of the truss member w.r.t. the x axis (Figure 4-5), the effective strains in the x and z direction vary. For small deformations, Eq. 4.1, holds.

\[ xdx = zdz \]  

**4.1**

For large deformations, the deformations \( dx \) and \( dz \) as shown in Figure 4-6, are related by the following equation

\[ (x - dx)^2 + (z + dz)^2 = l^2 \]  

**4.2**

![Diagram of large movement in x and z directions]

**Figure 4-6**: Large movement in the x and z directions
A change in cell size (the distance between opposite vertices of the octahedron) in one direction also results in some change in cell size in the other directions. If the size of the cell is limited in the thickness or z direction, then the maximum change in length in the other directions can be obtained as the change in z direction goes from the limit length \((z_d)\) to 0. When \(z = 0\), \(x\) is equal to \(l\) (length of the truss member). However, the structure would lose its shear stiffness if the angle of the truss member is allowed to go 0. Hence, the final angle is limited to some small value (10°). The relation between the maximum strain required, \(\varepsilon\), and the initial angle, \(\theta\), of the truss member (Figure 4-6) can be calculated as follows:

\[
\varepsilon = \frac{\text{change in length}}{\text{original length}} = \frac{(l \cos(10) - x)}{x} \\
\Rightarrow \varepsilon = l \cos(10)/(x - 1) \\
\Rightarrow x/l = \cos \theta = \cos(10)/(1 + \varepsilon) \\
\Rightarrow \theta = \cos^{-1}\left(\frac{\cos(10)}{1 + \varepsilon}\right)
\]

This relation is plotted in Figure 4-7. The figure shows that for a 15% strain, as is required by the HECS wing, the initial angle of the truss member should be 30°. Therefore, if the maximum thickness of the airfoil is 4\(\text{"}\), the allowable displacement in the thickness direction \((z_d)\) for a unit cell in the top-surface would be 2\(\text{"}\). With the initial angle of the truss member as 32°, the size of the unit cell in x direction is \((z_d \ast \cot(32)) = 3.2\text{"}\). The size of the unit cell obtained above is acceptable for the HECS wing since the maximum strain of 15% is only seen in a small region near the tip. Hence there would be a reasonable number of unit cells in the wing (10 – 30 cells in the spanwise direction and 4-7 in the chordwise direction). However, if the required strain is large, say 100%, for example, as in the TSCh wing concept of NextGen (Figure 4-8) [19], then the size of the unit cell is 1\(\text{"}\), and a large number of such cells would be required per unit area.

Each of the octahedral unit cell has 15 calbes. So if the number of unit cells increase the number of actuators required to morph the wing structure also increases. This would
tremendously increase the complexity and weight of the wing. Hence the octahedral unit cell is most effective when the morphing strains are not too large so that fewer unit cells are used.

Figure 4-7: Initial angle required for a given strain

Figure 4-8: Tsch Wing [19]
Another aspect that could limit the size of the unit cell on the upper end, is the accuracy of the airfoil geometry approximated by the unit cells. The airfoil geometry is a approximation obtained by discrete unit cells as shown in Figure 4-10. As the size of the unit cell increases the smoothness of the airfoil section of the truss structure decreases. If a flexible skin is chosen then the smoothness of the airfoil geometry could limit the maximum size of the unit cell used.

Having obtained the topology and size of the unit cell, the strength and weight of a wing structure made of these unit cells is analyzed in the next section.

4.3 Strength and Weight Considerations

If they are to see use in practice, such morphing aircraft structures must be capable of satisfying all the design requirements for primary aircraft structure: carrying basic structural loads, as well as those associated with active morphing.

4.3.1 Representative vehicle and wing

A model wing structure is sized for a vehicle weighing 3000 lbs, representative of a military unmanned air vehicle. The weight was obtained after surveying a few aircraft of the same category. The wing is assumed to be rectangular with a span of 15’ and chord of 2.5’. The wing is to be constructed using the octahedral unit cells described in the previous section. As mentioned before in section 4.2, the shear stiffness of the unit cell decreases as the angle, $\theta$, of the truss members (Figure 4-6), decreases. The unit cell has least shear stiffness in its extended configuration. The size of the unit cells along the span was chosen so that they are in their expanded configuration. 10 cells are used in the span-wise direction. The angle, $\theta$, of the truss members, is between $5^\circ$ and $8^\circ$ for the span-wise members. 7 cells are use in the chord-wise
direction to capture the smoothness of the airfoil geometry. The angle, $\theta$, of the truss members in the chordwise direction, is between $10^\circ$ and $16^\circ$. Different configurations of the unit cells are used for the top and bottom layers. The negative in-plane Poisson’s ratio unit cells (Figure 4-2) are used in the top layer and the positive in-plane Poisson’s ratio unit cells (Figure 4-4) are used in the bottom layer. The top and bottom layers are connected with truss members near the leading edge and trailing edge. The structural layout is shown in Figure 4-9 and a cross section of the wing is shown in Figure 4-10.

Figure 4-9: A wing made of the octahedral unit cells

Figure 4-10: Cross section of the wing structure
4.3.2 Wing loading and member sizing

The wing is designed for a loading condition with a limit load factor of ±4 and a factor of safety of 1.5. The load factor was chosen to be in between those used for cruising aircraft (2-3) and combat air vehicles (5-6). The load is distributed spanwise using the Schrenk’s approximation [34] and chordwise using the pressure distribution of the Clark Y airfoil. Schrenk’s approximation assumes that the span-wise load distribution at a span station is proportional to the mean of the chord of the original wing and chord of an elliptical wing with the same area and span as the original wing. This is illustrated in Figure 4-11.
c) Draw an ellipse with the same area enclosed as the area under the curve of chord distribution shown in (b), such that
\[
\frac{\pi}{4} sz = s(c_r + c_l)
\]
\[
\therefore z = \frac{2}{\pi}(c_r + c_l)
\]

b) Chord distribution along semi-span of length
\[
s = \frac{h}{2}
\]

\[
\text{area} = \frac{s}{2}(c_r + c_l)
\]

d) Chordwise pressure distribution and hence lift distribution.

a) Lay the ellipse in (a) upon the chord distribution, (b), and construct a mean line of lift distribution.

Figure 4-11: Figures illustrating the Schrenk’s approximation. Source: [34]
The ability of the wing structure to carry load is analyzed using linear finite element analysis. The truss members are assumed to be tubular, made of aircraft-grade aluminum alloy (Brush Wellman AM 162 AlBeMet® extruded bar[35]), and the cables are made of a high strength polymer such as Spectra®.

As discussed in chapter 3, during the finite element analysis, the stiffness of any cable having a compressive load is set to a small value to effectively eliminate such cables, and the analysis is repeated. In the next iteration some of the cables that were previously slack might be in tension as the structure is different. So the stress in entire set of cables is checked including those cables for which the stiffness was set to a small value. This process is continued until the same set of cables is carrying compressive loads as calculated in the previous iteration [36].

The areas of the truss members are also adjusted so that local buckling of cross-section or column buckling or material failure occurs at the ultimate load. The thin tubular truss members can fail either due to local buckling or as a column, which ever happens at a lower load. The stress at which local buckling occurs is given by Eq. 4.4 [37] and the Euler buckling load of the column with simply supported boundary conditions is given by Eq. 4.5.

\[
\sigma_t = \frac{1}{\sqrt{3}} \sqrt{\frac{E}{1-\nu^2}} \frac{t}{r} \tag{4.4}
\]

\[
2\pi rt\sigma_E = \frac{\pi^2 EI}{L^2} \tag{4.5}
\]

In these equations, \( t \) is the thickness of the tube, \( r \) is the radius and \( L \) is the length of the tube. \( E \) is the Young’s Modulus and \( \nu \) is the Poisson’s ratio. These two stress values are equated to get a relation between \( t \) and \( r \) as shown in Eq. 4.6

\[
t = \frac{\pi^2 \sqrt{3(1-\nu^2)r^3}}{2L^2} \tag{4.6}
\]
The force \( (N_i) \), carried by a particular truss member \( (i) \), at the ultimate load, can be obtained from the results of the finite element analysis. And using Eq. 4.4 and Eq. 4.6 the radius is calculated as shown in Eq. 4.7

\[
r_i = \left[ \frac{2L^4_i N_i}{\pi^4 E \sqrt{3(1 - \nu^2)}} \right]^{1/6}
\]

The areas of the tendons are adjusted so that tensile material failure occurs at the ultimate load. The stiffness of the structure and hence the stress in the members changes as the areas of the members are changed. The stresses in the members are recalculated and the members are sized again. This process is repeated a few times until the sizes and stresses converge. The areas of the members thus calculated were found to be reasonable even at the root where the members see the maximum load.

4.3.3 Flexibility and weight

If the members are sized only to prevent local failure under the applied loads, the deflections are relatively high; that is, the over-all structure is relatively soft. To increase the stiffness, the cables and truss members are resized by multiplying the ultimate load of each member \( (N_i) \) with an additional factor \((SR>1)\). The sizing process is repeated for different values of \(SR\) and the weight of the wing is calculated each time.

An additional 18 lbs \((8.16\ \text{kg})\) is added to all the configurations uniformly to represent the weight of the nodes, based on the assumption that the nodes are solid and spherical. The weight of the actuators and skin is not included.

Figure 4-12 shows the results in terms of weight of the wing structure vs. maximum (tip) displacement at the ultimate load. As the wing is made stiffer, \(i.e.,\) the areas of the truss members
and cables are increased, the tip deflections decrease but the weight of the wing structure increases. Also plotted in the figure is the weight of a conventional stiff wing of a light fighter aircraft, designed for the same loading conditions, i.e., a gross weight of 3000 lbs, load factor of 4 and factor of safety of 1.5. It has the same size, same planform, unswept wing and no variable sweep capability. This weight was obtained using the statistical weights method [41], and does not include the weight of the control surfaces or the actuators. This weight is an estimate of the weight of a non-morphing conventional stiffened-skin wing. The optimal weight of the truss wing is not known since the limiting deflection affects the weight of the truss. However, this analysis shows that the weight of the truss wing can be comparable to the weight of a conventional wing at this geometric scale.

Figure 4-12: Weight comparison of a truss-wing and a conventional wing at the ultimate load

The deflection of the wing of the military unmanned vehicle was calculated assuming a box beam with the dimensions obtained from the airfoil section. The deflections of the conventional wing are relatively small compared to those obtained for the truss wing. The control surfaces on a typical fighter aircraft are hinged and large deflections of the wing would cause the hinges to bind, making the control surface inoperable. This acts as one limiting factor for the
deflection of such a wing. In the truss wing concept, however, this is less of a concern since the whole wing or part of it can morph to provide control.

The torsional stiffness, $GJ$, of the wing was calculated by applying an end moment of the wing. The deflection at 80% span was used to calculate the change in slope of the wing on application of end moment. The value of the torsional stiffness was $1.68 \times 10^5$ N-m. The position of elastic axis was also estimated in a similar manner by applying vertical tip loads over a section of the chord and observing the deflection at 80% span. The location of the aerodynamic center for a Clark Y airfoil was obtained from a paper Noyes [43]. Using these data the divergence speed, $V_d$, for a finite wing was calculated as $1.28\times 10^5$ m/s from and Eq. 4.8 [44].

$$V_d = \sqrt{\frac{\pi^2 GJ}{2\rho ec^3b^5 \frac{\partial C_l}{\partial \alpha}}}$$

4.8

In the above equation, $\rho$ is the density of air at sea level; $b$ is the semi-span; $ec$ is the distance between the aerodynamic centre and the elastic axis of the wing; and $(\partial C_l/\partial \alpha)$ is the slope of the lift coefficient w.r.t the angle of attack. The truss wing is also highly adaptable, it could be actively morphed to avoid divergence, for example, by washout. Flutter speed could also be increased through the use of high-bandwidth active control [45]. The lateral shear stiffness of the structure decreases as the angle of the truss members decreases. However, even in the fully-extended configuration, truss members are at a small angle. The unit cells in the above analysis are closer to the fully-extended configuration.

As noted above, the weight of the truss wing does not include the weight of the skin. The details of the skin have not yet been established. The skin in a morphing truss wing will not necessarily add significantly to the bending stiffness of the wing, however, it does need to transfer the air loads to the underlying truss structure. If the skin is made of a lightweight polymer like Vectran® (200 denier, 50 x 50 plain weave), the additional weight of a skin is around 0.7 lb.
If segmented skins (discussed in chapter 6) are used, the additional weight could be somewhat higher.

The truss wing has a lower stiffness and a slightly higher weight penalty when compared to a stiffened skin structure. If the truss structure could only be used in the regions of the wing where the morphing is large then the weight penalty could be reduced. For example in the HECS wing it would need to be used in a small region near the tip where the morphing strains are large. The loading in this region is also much lower so the deflections would be smaller.

4.3.4 Nonlinear finite element analysis

Since the deflections are quite large, a nonlinear finite element analysis was also carried out and compared to the linear analysis. A higher-order strain measure which is often used in nonlinear analysis known as Green’s strain was used, which is defined as in Eq. 4.9

\[ \varepsilon_G = \frac{l_n^2 - l_o^2}{2l_o^2} \]  

where, \( l_n \) is the original length and \( l_o \) is the length after deformation. The corresponding stress, \( \sigma_G \), is referred to as the second Piola-Kirchhoff stress. Using a linear elastic material with modulus \( E \), the stress and strain are related as Eq. 4.10.

\[ \sigma_G = E \varepsilon_G \]  

The principle of virtual work is used to derive the equilibrium equations for a truss element. The equations for each element are then assembled to get the global equations of equilibrium. The load is increased in small steps and at each step an updated Newton-Raphson method is used to get the corresponding deflection [42].
The nonlinear analysis was done using member cross-section areas corresponding to a stiffness factor of SR=5 (Section 4.3.3). The maximum displacement obtained was higher than that obtained using linear analysis. The tip deflection using the nonlinear analysis was 68% of the chord whereas the linear analysis gave a tip deflection of 50% chord for the same loads. A movie of the deflection is shown in Figure 4-13.

Figure 4-13: Deflection of truss wing using nonlinear finite element analysis
4.4 Conclusions

The truss structure made of octahedral unit cells could be heavier than a conventional design, but has the additional ability to provide smooth and continuous deformations. While it might not be effective or necessary to design an entire wing from such cells, they could perhaps be used in those sections of the wing where morphing is especially important. For example, in the HECS wing, the morphing truss structure might be used mainly in a small spanwise section near the tip to bend and twist the wing tip, the wing tip itself being a stiff structure that acts somewhat like a control surface. Deciding exactly where to place morphing structure could be addressed more generally through a comprehensive trade study that compares the benefits of morphing with the penalties of weight and complexity.
Chapter 5

DESIGN USING THE GENETIC ALGORITHM

The octahedral unit cell was obtained as an optimum unit cell for a particular design requirement or fitness function in chapter 3, but it has a few drawbacks as seen in the previous chapter. A cellular structure made of the octahedral unit cells has many cables and the truss structure cannot be made adequately stiff without having a higher weight penalty. This chapter describes a few improvements made to the genetic algorithm and a few more unit cell designs are presented. A few other design strategies are also investigated using the algorithm developed.

5.1 Modifications to the Genetic Algorithm

The octahedral unit cell was obtained as an output three times from six independent runs made using the genetic algorithm. Since different results (octahedral and non-octahedral) were obtained as optimal solutions from the genetic algorithm, the mutation parameter, $mu$, was changed to see if it affected the repeatability solution in any way. Two values (0.01 and 0.1) were used for the mutation parameter. Each digit (gene) of a particular chromosome can undergo mutation. A random number is generated and, if this number is less than the mutation value, then the particular gene is mutated to any of the other three possibilities with equal probability. So the probability that at least one digit (gene) of a chromosome will change is

$$\left[1 - (1 - mu)^{ng}\right]$$
where, \( n_g \) is the number of genes in the chromosome. So if the mutation value is small, the probability of a chromosome changing is small. Hence the child population is almost the same as that formed entirely by crossover, and converges faster since favorable patterns prevail in the child population, and the entire population becomes uniform after some time. But if the mutation value is large then the population does not become uniform, which is good to keep diversity for a wider search but is not optimal when convergence is required. In order to capture both these contradictory requirements, a high value (0.1) was chosen initially for the mutation parameter. And this was multiplied by a factor less than one (0.3-0.5) every eighty to one hundred generations thus gradually reducing the mutation value. The mutation value would be large in the beginning so that more search area is explored but, towards the end when optimal patterns are found, smaller mutation values would help in obtaining the best solution faster. Another technique that was used to increase the diversity further, is known as the micro-GA [51]. After a fixed number of generations (400-1000) a new set of population was generated using a random number generator, also keeping a few of the better individuals from the previously-converged generation. The mutation value was simultaneously reset to the high number (0.1). This increased the robustness of the genetic algorithm in finding the global optimum. The choice numbers described above depend on the size of the problem. For the 3D unit cell with 28 independent variables with a population size of 1000 and calculations up to 2000 generations the following values were used: The mutation value was multiplied by 0.5 every 80 generations. A new random population was generated every 200 generations with 5 chromosomes being carried over to the new population.
5.1.1 Variable forces

All the cables were actuated with the same force in the calculation of the fitness due to actuation, in chapter 3. But it might be possible to get a better fitness if the cables were actuated using different forces. For a single output point, the forces in the cables can be the same since each active cable would cause the displacement at the output point to either deflect in the same direction as other cables in the set or in the opposite direction. If the displacement is in the same direction as the remaining cables then it would add to the required displacement increasing the fitness. The system is linear so these forces can be considered to be acting individually one by one, and to get the maximum output, it would be preferable to have a maximum value for all the forces. But if the force in a particular active cable causes it to deflect in the opposite direction to the remaining active cables then it should not be there in the first place – it should be a part of the antagonistic set. The GA takes care of such structural layouts, since such a cable decreases the fitness of the structural layout, and that particular layout is not preferred in the next generation.

The optimization of the magnitude of the forces is effective in obtaining a better solution if there are contradictory requirement(s), for example, a limit on the total amount of force that could be used in all the cables or, as in the current case, a requirement that the structure should deform into a particular shape. In the case of deforming the structure into a particular shape, the requirement would be to reduce the error between current output shape and a desired shape. In the unit cell addressed here, the requirement is to have a planar output face i.e., have a uniform displacement in all the output points.

There are two ways to find the optimal forces for a structure. One way is to include them as an additional variable in the GA. Modified genetic algorithms exist that can handle continuous as well as discrete variables. There would be one force variable for each of the members and it would be taken into consideration if the particular member happens to be an active cable. So there
are two variables associated with each member: one, a discrete variable which determines whether the member is a strut, void, cable1 or cable2; and the other a continuous variable which is associated with the force in the cables. The second way is to have an inner optimization loop which finds the best configuration of forces for the particular structural layout.

The first method is inefficient for the current algorithm for because: 1) It would tremendously increase the total number of possible structural layouts, as, the number of variables is doubled. And, the number of possible structural layouts increases exponentially with increasing variables. 2) The optimal configuration of forces is independent of the calculation of displacement due to the external loading. And the calculation of the displacement due to loading is more time-consuming than the calculation to find the fitness due to actuation. Including the variables externally would require the time-consuming calculation of displacement due to loading to be done for any change in the actuation force variables too.

The optimal force set combination was found for each of the two sets of cables using an inner optimization loop which finds the best configuration of forces for the particular structural layout. The resulting force vector would give the best (fitness)\(_a\) for the current structural layout. The function that is to be optimized, \(P_{\text{disp}}\), is shown in Eq. 5.2, the mean, \(\mu\), and variance, \(\sigma\), are calculated as shown previously, in Eq. 3.15. \(U^d\), the displacement output from the linear finite element analysis, is found by solving Eq. 3.3. The global nodal force vector, \(F\), is formed from the forces acting due to the cables, \(F_c\), as shown in Eq. 5.3, where \(G_F\) is a geometric matrix relating the nodal forces to the forces in the cables.

\[
P_{\text{disp}} = \frac{\mu}{0.01 + \frac{\sigma}{|\mu|}}
\]  \hspace{1cm} 5.2
Using Eq. 3.3 in conjunction with Eq. 5.3 we get the following equation Eq. 5.4

\[ U^d = K^{-1} * G_F * F_c \]  

\( K^{-1} * G_F \) remains constant for a particular structural layout. Hence while the optimal forces are being calculated, the stiffness matrix needs to be inverted only once. The displacement \( U^d \), is linearly related to the force in the cables and the function to be optimized, \( P_{disp} \), is a continuous function of the output displacements. Hence a constrained gradient-based optimization approach is used to find the optimal solution. A library of optimization routines from Sandia National Labs was used [39]. A quasi-Newton method was used which required the first derivatives of \( P_{disp} \). These are calculated as shown below in Eq. 5.5. The forces in the cables were constrained to be positive and lower than a limit value, so that the deflections don’t become very large.

\[ \frac{\partial P_{disp}}{\partial F_c} = \frac{\partial P_{disp}}{\partial U^d} \cdot \frac{\partial U^d}{\partial F_c} = K^{-1} * G_F \]

\[ \frac{\partial U^d}{\partial F_c} = \frac{\partial P_{disp}}{\partial U^d} = \left( \frac{0.01 + \sigma/|\mu|}{\sigma} \right) \cdot \frac{\partial U^d}{\partial U^d} \]

\[ \frac{\partial P_{disp}}{\partial U^d} = \frac{1}{5} \left( \frac{U^d - \mu}{\mu^2 - \sigma^2} \right) \frac{\partial \sigma}{\partial U^d} = \frac{1}{5} \left( \frac{U^d - \mu}{\mu^2 - \sigma^2} \right) \]

The possibility of variable forcing increased the fitness of all the different optimal solutions obtained, but the change was not very large. Symmetry about three planes is enforced
on the 3D ground structure, so all the solutions obtained are symmetric. And, the output function corresponds to a uniform displacement of the entire top face. So the optimal forces for all the cables are nearly equal. However significant improvements in the output fitness functions were observed in non-optimal solutions. So optimizing the forces for each candidate helped in comparing the best results of each candidate.

5.1.2 Checking for pseudo-mechanisms

The analysis of the structure is carried out using linear finite element analysis. In the actuation phase the stiffness of all the cables is set to a small value, hence essentially only the structure formed by truss members is analyzed. The structure formed by the truss members should ideally be a mechanism and not a stiff structure so that it deforms without storing strain energy in the structure. However, some of the optimal solutions obtained in the genetic algorithm had mechanisms similar to one shown in Figure 5-1

![Illustration of a “pseudo-mechanism”](image)

Figure 5-1: Illustration of a “pseudo-mechanism”

The structure shown in the above figure does not have stiffness at the center node (pin joint) in the perpendicular direction, shown by the arrow, and is essentially a mechanism for very small deflections. However, as the structure is actuated using the forces in the cables, the deflections increase and the structure ceases to be a mechanism. A linear analysis fails to capture this increase in stiffness, and hence, the fitness due to actuation, \( (fitness)_a \), calculated using the linear analysis for such a structure, will not correspond well with its ability to morph. A full
nonlinear analysis of the structure would solve this problem, but this would considerably increase the computation time. Hence, an intermediate approach is adopted.

The fitness due to actuation for each set of cables is calculated in two steps, using linear analysis in both steps. The first step is the same as the previous linear analysis, except that the actuation force limit is much smaller. In the second step, the deformed geometry obtained by adding the displacements, calculated in the first step, to the original geometry, is analyzed. If the structure shown in Figure 5-1 is analyzed in step 2 it will look as shown in Figure 5-5. Hence if there are any “pseudo-mechanisms” present in the structure they will be eliminated in the second step. The forces were optimized in each step to give the best fitness value.

In the second step, the geometry of the structure is slightly different, but the stresses in the members are 0. Including the pre-stress in the second step would involve an iterative solution procedure, which is computationally expensive. It is in this regard that it differs from a nonlinear finite element analysis.

---

![Figure 5-2: Deformed geometry of the pseudo-mechanism shown in Figure 5-1](image-url)

The modifications were made to the genetic algorithm. The results from the modified algorithm are discussed in the next section.
5.1.3 Results for the modified genetic algorithm

The octahedral unit cell also had a large number of cables so the fitness function was modified from Eq. 3.14 to have a higher penalty on the cables as shown in Eq. 5.6. The corresponding unit cell obtained from the genetic algorithm is shown in Figure 5-3

\[
\text{fitness} = \frac{\text{fitness}}{(Vf_{cable} + Vf_{truss})Vf_{cable}^2}
\]

5.6

Figure 5-3: An optimal unit cell for the modified genetic algorithm

The unit cell shown above has 11 cables. Upon actuation the displacement in the z direction is accompanied by an equal and opposite displacement in the x direction. There is no...
displacement in the y direction. The ratio of these displacements can be modified as was done in the octahedral case by changing the lengths of members in the two directions.

A solution obtained for a higher weight penalty on the cables is shown in Figure 5-4 and the corresponding equation for fitness function is Eq. 5.7

\[
\text{fitness} = \frac{\text{fitness}}{(Vf_{\text{cable}} + Vf_{\text{truss}}) Vf_{\text{cable}}^3}
\]

Figure 5-4: An optimal unit cell for the modified genetic algorithm with a higher weight penalty for the cables
The unit cell shown in Figure 5-4 had 10 cables. The displacement pattern upon actuation is similar to the previous case, i.e., the displacement in the z direction is accompanied by an equal and opposite displacement in the x direction and there is no displacement in the y direction.

The fitness values for all the three optimal unit cells (Figure 3-17, Figure 5-3 and Figure 5-4) calculated using the three different fitness functions (Eq. 3.14, Eq. 5.6 and Eq. 5.7) are shown in Table 5-1. Each unit cell has the highest fitness value among the three when its corresponding fitness function is used. The unit cells developed on this chapter have fewer cables than the octahedral unit cell. But even these unit cells are considerably complicated and hence different approaches are investigated that are described in the following sections.

<table>
<thead>
<tr>
<th>Table 5-1: Fitness values of different optimal solutions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure</td>
</tr>
<tr>
<td>---------</td>
</tr>
<tr>
<td>Figure 3-17</td>
</tr>
<tr>
<td>Figure 5-3</td>
</tr>
<tr>
<td>Figure 5-4</td>
</tr>
</tbody>
</table>

5.2 Alternative Approach to Designing a Morphing Structure

A cellular approach using truss members and cables can be designed to build a morphing wing but there are penalties to be paid in increasing complexity. Also, the change in area achievable by such a structure is limited as seen in the previous chapter. A 2D approach is investigated to reduce the overall complexity of the problem. The wing is assumed to be made up of beams which carry the lift load as shown in Figure 5-5. The wing needs to be stable under drag and lift loads and have the ability to morph in-plane into a required shape when the cables are actuated.
A three dimensional frame element was used to model a member of the structure. The stiffness matrix of a 3D frame element is shown in Eq. 5.8 [40]. The frame element has 12 degrees of freedom, with three translational and three rotational degrees of freedom at each end. The six degrees of freedom at node $j$ are shown in Figure 5-6.

---

$K(9,9)$ is misprinted in the book.
\[ K = \begin{bmatrix}
AS & 0 & 0 & 0 & 0 & 0 & -AS & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\]

The terms in this equation are defined as follows in Eq. 5.9

\[ AS = \frac{EA}{l}; \quad TS = \frac{GJ}{l}; \quad a_z = \frac{12EI_z}{l^3}; \quad b_z = \frac{6EI_z}{l^2}; \quad c_z = \frac{4EI_z}{l}; \quad d_z = \frac{2EI_z}{l}; \]
\[ a_y = \frac{12EI_y}{l^3}; \quad b_y = \frac{6EI_y}{l^2}; \quad c_y = \frac{4EI_y}{l}; \quad d_y = \frac{2EI_y}{l} \]

where, \( A \) is the area of the beam; \( E \) is the Young’s Modulus; \( I_x, I_y \) and \( J \) are the area moments of inertia of the cross-section of the beam about the Y, Z and X axes respectively; and \( l \) is the length of the beam. The X axis is along the beam, the Z axis is perpendicular to the plane of the structure, and the Y axis is perpendicular to both the X and Z axes.

In the current case, all the beams are assumed to be pinned at the nodes. So, the degrees of freedom corresponding to rotation about the \( z \) axis are eliminated from the equations. The 6th and 12th rows and columns of the stiffness matrix are deleted. The bending stiffness \( a_z \), is set to zero. The cables and voids also use the same stiffness matrix as the beams but the properties are
changed to suit each member. In case of a void or a slack cable, \textit{i.e.}, in compression, all the stiffnesses are set to a small value. In case of a cable in tension the bending terms are multiplied by a small number.

5.2.1 Problem setup

An entire wing made of the beams is designed instead of designing only a unit cell, since, it is possible that a combination of different cells would behave differently than the cells individually. And, there could be a better solution possible if different cells are used in the entire wing geometry rather than a single type of cell. The analysis and problem setup is same as in the truss examples. The ground structure used for the entire wing is shown in Figure 5-7.

![Figure 5-7: The ground structure for the 2D beam design of the wing](image)

The wing deforms in plane when the cables are actuated. The required shape that the wing deforms to is shown in Figure 5-8. This shape was chosen based on the TSCh wing shown in Figure 4-8. The wing sweeps back by 30° and reduces in length by 25% which results in an area change of 51%.
Unlike in the previous case, where the displacement of the entire face was maximized, in the current case the required deformed shape is specified. Hence, the distance between the actual deformed shape and the required deformed shape is minimized as shown in Eq. 5.10. The fitness due to actuation that needs to be minimized is the sum of the squares of the distances between the required and obtained deformed coordinates of a chosen set of points. The points on the leading edge and the x coordinates of the points on the tip are used to calculate the fitness due to actuation.

\[
(fitness)_i^{\text{Constrained}} = \sum_{i}^{n_{\text{Constrained}}} \left[ (x_{i}^{\text{def}} - (x_{i}^{\text{orig}} + u_{i}))^2 + (y_{i}^{\text{def}} - (y_{i}^{\text{orig}} + v_{i}))^2 \right] \tag{5.10}
\]

In this equation, variables with superscript \textit{def} correspond to the coordinates of deformed geometry, and those with superscript \textit{orig} correspond to the coordinates of the original configuration, and \(u_{i}\) and \(v_{i}\) correspond to the displacement of the coordinates on actuation. Only a part of the total points were chosen in the fitness calculation to give some freedom to the design.

Figure 5-8: Required deformed shape of the wing.
variables and increase the design space. If all the points were chosen the problem would be too constrained.

As before, two sets of cables are used. One set deforms the structure into a required shape and another set brings it back to the original shape from the deformed shape. When the second set of cables was actuated, the structure was formed in the required deformed shape and the actuation forces were applied. Equation 5.10 was used when the first set of cables was actuated and Eq. 5.11 was used when the second set of cables was actuated. The two fitness values were added to find the total \((fitness)_a\) due to actuation of cables.

\[
(fitness)_a^{\text{Constrained}} = \sum_{i} \left[ \left( x_i^{\text{orig}} - (x_i^{\text{def}} + u_i) \right)^2 + \left( y_i^{\text{orig}} - (y_i^{\text{def}} + v_i) \right)^2 \right]
\] 5.11

The geometry of the structure is different when the deflection due to the second set of cables is calculated. One might argue that the structure being analyzed is not the same. But this is a linear analysis which can only result in giving a required direction. The goal is to get the connectivity of the trusses and the cables that will give the best results when either set of cables is actuated and the connectivity is the same in both the cases.

The stability with external loads was checked using three load cases in both directions, positive and negative. The two in-plane load cases are shown in the Figure 5-9. The third load case was set of loads acting on all the nodes in the z direction, \(i.e.,\) perpendicular to the page.
The forces are optimized as described before in section 5.1.1. The function to be
optimized, $P_{\text{disp}}$, in the current case is the same as $(\text{fitness})^1_a$ or $(\text{fitness})^2_a$ for the first set or the
second set of cables respectively. The first derivative of $(P_{\text{disp}})$ w.r.t. the forces is calculated as
shown in Eq. 5.5 in terms of the derivative w.r.t. the displacement. The derivatives of $(P_{\text{disp}})$ w.r.t.
the displacement for the first set of cables is calculated as shown in Eq. 5.12. The corresponding
derivatives are then inserted into the appropriate position in the vector.

$$\frac{\partial P_{\text{disp}}}{\partial u_i} = 2\left((x_{i}^{\text{def}} + u_i) - x_{i}^{\text{orig}}\right)$$

$$\frac{\partial P_{\text{disp}}}{\partial v_i} = 2\left((y_{i}^{\text{def}} + v_i) - y_{i}^{\text{orig}}\right)$$

5.12

5.2.2 Results

The calculation of the final fitness function followed the same pattern as those used in the
3D truss structures. However, in the current case as the fitness due to actuation, $(\text{fitness})_a$, needed
to be reduced, hence, it was included in the denominator. Different fitness functions were tried
and the following fitness function shown in Eq. 5.13 yielded the best results. The other variables in the equation: \(compliance, V_{fcables}, V_{ftruss}\) were calculated as in chapter 3 (Eq. 3.6 and Eq. 3.9).

\[
\text{fitness} = \frac{1}{\left(0.1 + \left(\frac{\text{fitness}}{a}\right)^3\right)} \frac{1}{(V_{fcables} + 0.5V_{ftruss})} \tag{5.13}
\]

There are 106 elements in the structure, and, since reflection is not used, this results in 106 independent variables. The number of possibilities are much higher so, the algorithm was run for 10000 generations, compared to 2000 used before, using the same population size as before, i.e., 1000. The other parameters for the modified genetic algorithm are as follows: a new random population is generated every 500 generations, 2 genes are carried over from the older set, and the mutation value is multiplied by 0.5 every 80 generations. The result from the analysis is shown in Figure 5-10. The actuated configurations are shown in Figure 5-11.

---

Figure 5-10: Optimal beam solution for the morphing wing
The structures obtained from the optimization don’t perform as well as required. The choice of the ground structure greatly determines the quality of the output. The genetic algorithm can only chose among the possible designs that are described by the ground structure. In some cases it might be preferable to have longer beams and cables, which is not possible in the way the problem is designed above. A fully connected ground structure could solve the problem. But the number of elements increases significantly in a fully connected structure. An alternative approach is described in the next section.

5.2.3 End moment variables

The beam elements in the above case were assumed to be connected using pin joints. If the beams are now assumed to be connected at the ends with a stiff rotational spring $M$, then two beams connected with such springs would essentially behave as a single beam as shown in Figure 5-12. However if the stiffness of the spring $M$, is low then the two beams behave as though...
they were pinned. Hence if $M$, was included as a variable in the genetic algorithm, it would allow
the use of longer beams as well as short beams.

Two springs would be needed for each member, one at each end. Each spring can have
either a high stiffness or a very low stiffness resulting in $2n$ variables each having two
possibilities. Hence, if there are $n$ members in the ground structure each having 4 possibilities,
void, strut, cable 1 or cable 2, the total number of possible structural lay outs is $(4^n)(2^2^n) = 16^n$.
These springs would be useful only at the ends of truss members, since the cables and voids have
a low bending stiffness. Hence instead of including the springs as separate variables we could
include three new types of members: 1) a beam member which is pin connected at one end and
which is rotationally stiff at the other, 2) a beam member which is rotationally stiff at both the
ends, or 3) a beam member which is stiff at one end and pin connected at the other (type (1)
flipped over). The total number of possible structural layouts is now $7^n$.

The elemental stiffness matrices are developed for the new beam elements. The
derivation is given in Appendix C. The local degrees of freedom for a single beam element
represented by Eq. 5.14 are shown in Figure 5-13

$$u = \begin{bmatrix} u_1, u_2, u_3, u_4 \end{bmatrix}$$ 5.14
The local stiffness matrix for a beam with a pin joint on the right hand side is given by Eq. 5.15. In this equation \( I \), is the length of the element, \( I \), is the area moment of inertia of the cross-section of the beam, \( E \), is the Young’s Modulus of the material of the beam, and \( \delta \), is a very small number that represents the low rotational stiffness of the beam with respect to \( u_4 \).

\[
k = \frac{3EI}{l^3} \begin{bmatrix}
1 & l & -1 & \delta \\
l & l^2 & -l & \delta \\
-1 & -l & 1 & \delta \\
\delta & \delta & \delta & \delta
\end{bmatrix}^{5.15}
\]

Similarly the local stiffness matrix for a beam with a pin joint on the left hand side is given by Eq. 5.16

\[
k = \frac{3EI}{l^3} \begin{bmatrix}
1 & \delta & -1 & l \\
\delta & \delta & \delta & \delta \\
-1 & \delta & 1 & -l \\
l & \delta & -l & l^2
\end{bmatrix}^{5.16}
\]

The local stiffness matrix corresponds to the stiffness of the beam for in plane bending about the z axis. These local stiffness matrices for a beam element with a pin on the one end included in the stiffness matrix of a 3D frame element (Eq. 5.8) and the corresponding stiffness matrix corresponds to a 3D frame element with a pin about the z- axis on one end (Eq. 5.17).
where \[ a_x = \frac{3EI_x}{l^3}, \quad b_x = \frac{3EI_x}{l^2}, \quad c_x = \frac{3EI_x}{l} \]. If the right end is pinned then \( \delta_R = 1 \) and \( \delta_L \) is a small value; and, if the left end is pinned then \( \delta_L = 1 \) and \( \delta_R \) is a small value. The other terms in the equation are same as before.

### 5.2.4 Step wise actuation

A linear analysis does not do justice to the calculation of fitness due to actuation, \((fitness)_{a}\). The deformations are large and the geometry of the structure changes significantly as it is actuated. This could be another reason why the results from the genetic algorithm, using the 2D pin connected beam ground structure, did not yield good results.

The calculation of the fitness actuation could be improved by using a stepwise actuation as follows. The structure is actuated by a small amount in each step. The geometry of the
structure is updated in each step. Thus in each step a new structure is analyzed; the geometry of which is obtained from the deformed configuration of the previous step.

The actuation of the cable is modeled by using equal and opposite forces on the nodes at the ends of the cable and by setting the cable element to a void element (element with a really small stiffness). The amount of force used in the cables becomes critical in this case. The deformation of the structure formed by the beam elements and voids should be reasonably small in each step; even if the structure formed by the beams, is a mechanism. The forces in the cables are optimized in each step to reduce the error between the required configuration and the current deformation. The upper limit of the forces is set to a small value. The value of the force is such that when applied on a single void element the deformation of the element is of the order of length of the element divided by the number of steps.

The geometry of the structure would change significantly upon actuation, if the structure formed by the beam elements was a mechanism, but the stresses and deformations in the beams would be small. Thus analyzing the deformed geometry from a zero stress state in each step would not lead to significant errors. One does not need an iterative calculation of a full nonlinear analysis.

The first set of cables is actuated the structure from the current configuration to the deformed configuration. The actuation is done in steps. The error is calculated as the sum of the squares of difference between the deformed configuration after the final step and the required configuration of a chosen set of points as described in section 5.2.1 The calculation of this error is shown in Eq. 5.18. In this equation, variables with superscript final, correspond to the coordinates of deformed geometry at the final step of actuation using first set of cables, and those with superscript def, correspond to the coordinates of the required deformed configuration.
The deformed geometry obtained in the final step of the actuation using the first set of cables, is used as the starting geometry in the first step of actuation using the second set of cables. This is a more accurate representation of the actual case. The fitness function due to actuation using the second set of cables is calculated as shown in Eq. 5.19

\[
(fitness)^2 = \sum_{i=1}^{n_{\text{Constrained}}} \left[ (x_i^{\text{def}} - x_i^{\text{final}})^2 + (y_i^{\text{def}} - y_i^{\text{final}})^2 \right]
\] 5.19

The forces are optimized in each step so that the error between the required configuration and the current configuration is minimized. The function that is minimized \( P_{\text{disp}} \), is calculated as shown in Eq. 5.20

\[
P_{\text{disp}} = \sum_{i=1}^{n_{\text{Constrained}}} \left[ (x_i^{\text{req}} - (x_i + u_i))^2 + (y_i^{\text{req}} - (y_i + v_i))^2 \right]
\] 5.20

where, \( x_i^{\text{req}} \) and \( y_i^{\text{req}} \), are coordinates of the required configuration, deformed or original as the case might be; \( x_i \) and \( y_i \) are the undeformed coordinates of the structure in the current step; and, \( u_i \) and \( v_i \) are the deformations obtained in the current step. The partial derivatives are calculated as before (Eq. 5.12).

### 5.2.5 Factors affecting designs obtained from the genetic algorithm

The algorithm can only chose among the possible designs which are described by the ground structure. Sometimes, obtaining a required deformation might not be possible from any combination of the ground structure. A more comprehensive ground structure spans a greater
design space and hence there is a better probability of obtaining a good solution. A method for improving the pin connected 2D beam ground structure was suggested in section 5.2.3.

Apart from the formulation of the ground structure, the quality of the design obtained also depends on choice the fitness function. The fitness function chosen for the 2D beam analysis primarily represents how well the structure could morph into a required shape while carrying the loads. While there might be cases where one might want to design an aircraft that morphs to a required shape, the goal currently is to find a structural concept that can achieve significant changes in area while carrying the air-loads. An effective way to achieve such a design might be to include the change in area into the fitness function, while letting go of the requirement of morphing into a specific shape. The design space would open up a little more and based on the results obtained the problem could then be constrained to guide the development of a reasonable wing.

5.3 Conclusion

The genetic algorithm developed in this research is an effective tool that could be used in the design of cable actuated morphing aircraft structures. Two new unit cells designs were developed using the parallel genetic algorithm using a slightly modified fitness functions in which the cables were penalized more. One of the unit cells had 10 cables and the other had 11 cables compared to 18 in the octahedral unit cell. Even the newer cells with fewer cables than the octahedral unit cell have a significant number of cables. Hence, another design approach using 2D beams was investigated using the parallel genetic algorithm. Preliminary results obtained from the genetic algorithm are presented. And suggestions were made to improve the design procedure.
Chapter 6
THE COMPLIANT JOINT AND SKIN

The designs for the internal support structure of a morphing wing have been discussed so far. The truss members were assumed to be connected through pin joints in the 2D case and through rotational ball joints in the 3D case. In reality if 3D rotational joints were used the weight penalty would increase significantly, and one of the issues with using 2D pin joints is that they could bind and resist motion on application of loads. As suggested in chapter 2, a compliant joint could be used to connect the truss members. This chapter describes some of the studies made on the compliant joint. The latter section of the chapter also addresses some of the issues regarding the skin.

6.1 Design of the Compliant Joint

Preliminary studies were made to see if a compliant joint could be designed using existing materials to repeatedly undergo the required deformation while carrying the required load. As described previously, a possible realization of the compliant joint could be a rod made of Shape Memory Alloy (SMA) operating in pseudoelastic mode.

A compliant joint needs to carry the required axial load while bending during morphing. The maximum angular deflection required of a truss member in the wing structure discussed in chapter 4 for the required shape change is around 30°. This angle is estimated from the required morphing strain in the wing as described in section 4.2. The change in angle can be achieved with an angular deflection of ±15° of the truss member. Hence the SMA joint designed for the wing
made of octahedral unit cells should be able to deflect through an angle of $15^\circ$ while carrying the required load. A load of 110 N was used to illustrate the design procedure. The loading and deflection of the joint is shown in Figure 6-1.

![Figure 6-1: Deflection and loading on the compliant joint.](image)

The design of the compliant joint made of SMA material is carried out in two steps. In the first step, the maximum stress in a beam made of linear elastic material is determined for an axial load of 110 N and a tip deflection of $15^\circ$. The variation of maximum stress in the beam with varying beam thickness, beam length and material Young’s Modulus is obtained. In the second step, an effective linear modulus is obtained for an SMA beam in bending, and the range of values for beam thickness and length are determined such that the maximum stress is less than the ultimate stress of the SMA material.

### 6.1.1 Deflection of a beam under combined bending and axial load

The compliant joint is designed as a beam of length, $l$, with a square cross-section of thickness, $b$. The analysis of a Bernoulli–Euler beam under combined bending and axial loads is done assuming that the deflections are small. The governing differential equation is shown in Eq. 6.1, where $w$ is the deflection of the beam, and $M_o$ and $N$ are the end moment and axial load
respectively, as shown in Figure 6-2. \( \omega_e \) is the tip deflection, \( I \) is the area moment of inertia of the beam and \( \theta_e \) is the tip deflection in radians.

\[
EI \frac{\partial^2 w}{\partial x^2} = M_o - N \sin \theta_e (l - x) + N \cos \theta_e (\omega_e - \omega)
\]  \( 6.1 \)

On simplifying, Eq. 6.1 becomes:

\[
\frac{\partial^2 w}{\partial x^2} + \frac{N \cos \theta_e \omega}{EI} \cdot w = \frac{M_o - N \sin \theta_e (l - x) + N \cos \theta_e \omega_e}{EI}
\]  \( 6.2 \)

Define \( p \) as:

\[
p^2 = \frac{N \cos \theta_e}{EI}
\]  \( 6.3 \)

Substituting Eq. 6.3 in Eq. 6.2 gives Eq. 6.4

\[
\frac{\partial^2 w}{\partial x^2} + p^2 w = \frac{M_o}{N \cos \theta_e} - \frac{N \sin \theta_e (l - x) + \omega_e}{N \cos \theta_e}
\]  \( 6.4 \)

The solution to the differential equation in Eq. 6.4 is

\[
w = C_1 \sin px + C_2 \cos px + \left( \frac{M_o}{N \cos \theta_e} - \frac{N \sin \theta_e (l - x) + \omega_e}{N \cos \theta_e} \right)
\]  \( 6.5 \)
Where the constants of integration \( C_1 \) and \( C_2 \) can be determined from the boundary conditions as shown:

\[
 w(0) = 0 \\
 \Rightarrow C_2 = \left(-\frac{M_o}{N \cos \theta_e} - \frac{N \sin \theta_e}{N \cos \theta_e} l + w_e\right) \\
 w'(0) = 0 \\
 \Rightarrow C_1 = -\frac{\sin \theta_e}{P \cos \theta_e}
\]

The tip deflection, \( w_e \), can be determined from Eq. 6.5 by substituting \( x = l \).

\[
 w_e = \frac{1}{\cos pl} \left[ \frac{M_o}{N \cos \theta_e} (1 - \cos pl) + \tan \theta_e \left( l \cos pl - \frac{\sin pl}{p} \right) \right] \quad 6.7
\]

The required tip slope, \( \theta_e \), is 0.262 rad \((15^\circ)\). If the axial load is zero then the moment, \( M_o \), required to achieve the required tip slope is given by Eq. 6.8

\[
 M_o = \frac{EI \theta_e}{l} \quad 6.8
\]

As the axial load is increased, the tip angle decreases, if the moment remains constant. And, if the tip slope is to be maintained constant the applied moment needs to increase. The value of the moment, \( M_o \), required to maintain a constant tip slope for increasing axial load is found using an iterative solver, \texttt{fzero}, in Matlab. The variation of the tip deflection with increasing axial load for a beam made of steel \((E = 200 \text{ GPa})\), with a square cross-section of thickness, \( b \), 0.8 mm and length, \( l \), 2 cm, is shown in Figure 6-4. The plot of the deformation of the beam with increasing load is shown in Figure 6-3. The deflection plotted in the figure may not correspond to the actual deflection in the beam for larger deflections, since a small deflection assumption is made in the derivation of the equations. The deflections calculated here are assumed to be approximately valid for smaller deflections with tip deflections, less than 25 % of the length,
shown by the dotted line in the figure. A plot of the corresponding moment with increasing axial load and tip deflection is shown in Figure 6-5.

Figure 6-3: Deformation of the beam

Figure 6-4: Plot of axial load vs. tip deflection (constant tip angle).
The maximum stress in the structure occurs near the tip and is given by Eq. 6.9

$$\sigma_{\text{max}} = \frac{M_s b}{I} + \frac{N}{b^2} \tag{6.9}$$

The maximum stress should be less than the material strength for the ultimate load and less than the yield stress when the maximum load factor is used. The following figure shows the variation of the maximum stress with varying thickness (1 mm – 3 mm) and length (1.5 cm – 3 cm) when the beam is subjected to an axial load of 110 N and the tip angle is maintained at 15° as before.
It is easier to bend the beam to the required angle as the length increases. $M_o$ decreases with increasing length. Hence, the maximum stress in the beam decreases with increasing length. The plot only shows those results when the deflections of the beam are less than 25% of the length. If the thickness is smaller than 1 mm the deflections of the beam increase and are greater than 25% of the length. And as the thickness is increased, the stress first decreases, since the tip deflection decreases and lower moments are required to maintain the required tip angle as shown in Figure 6-5 and, simultaneously, the stress due to the axial force also decreases since the area is increasing. And after a certain point the tip deflection does not vary much and the moment required only increases since, the area moment of inertia, $I$, is increasing.

### 6.1.2 Bending of shape memory alloy beam

In some temperature range, shape memory alloys exhibit pseudoelastcity. The idealized stress-strain relation of a shape memory alloy is shown in Figure 6-7. The stress increases linearly initially ($\varepsilon < \varepsilon_i$) and then there is a region where the alloy undergoes phase
transformation and the stress is almost constant in this region ($\varepsilon_1 < \varepsilon < \varepsilon_2$). After the phase transformation is complete, the material behavior is almost linear elastic ($\varepsilon > \varepsilon_2$). During the unloading phase the material does not exactly follow the loading curve, but the SMA material has a high recoverable strain. The behavior of the material is slightly different for tension and compression. But these differences are ignored in the current analysis as the corresponding errors are small [52]. The plateau stresses ($\sigma_1$ and $\sigma_2$) are a function of temperature. This effect is also not considered in the current calculation. The properties of the material used in the current calculation are assumed to be valid at operational temperature.

![Stress-Strain diagram for ideal pseudoelastic material.](image)

Figure 6-7: Stress-Strain diagram for ideal pseudoelastic material.

As in the previous section, a beam with a square cross-section of thickness, $b$, and length, $l$, is analyzed. Plane sections of an initially straight beam which are perpendicular to the neutral axis remain plane and perpendicular in the deformed state. The strain, $\varepsilon$, of a layer at a distance, $z$, from the neutral axis is given by Eq. 6.10

$$\varepsilon = \Phi z \quad 6.10$$
where $\Phi$ is the curvature of the neutral axis. The stress distribution for increasing load is shown in Figure 6-8 [53].

Figure 6-8: Stress distribution across the thickness of the SMA beam. [53]

$\Phi_1$ and $\Phi_2$ in this figure are related to the strains $\varepsilon_1$, $\varepsilon_2$ shown in Figure 6-7 by Eq. 6.11.

When the load is increase the yielding starts in the outer layer of the rod when the curvature $\Phi = \Phi_1$. The yield region develops as the load and curvature increases up to the point when $\Phi = \Phi_2$. And after that the outer layer is in the elastic region beyond point 2 in Figure 6-7.

$$\Phi_1 = \frac{2}{b} \varepsilon_1$$
$$\Phi_2 = \frac{2}{b} \varepsilon_2$$

Using the preceding information the bending moment can be calculated for increasing curvature. The relation between the bending moment and the curvature as shown in Eq. 6.12

$$M_o = \begin{cases} EI\Phi & 0 \leq \Phi \leq \Phi_1 \\ EI \left\{ \Phi \left(\frac{\Phi_1}{\Phi}\right)^3 + \frac{3}{2} \Phi_1 \left[1 - \left(\frac{\Phi_1}{\Phi}\right)^2\right] \right\} & \Phi_1 < \Phi \leq \Phi_2 \\ I\Phi \left[ E \left(\frac{\Phi_1}{\Phi}\right)^3 + E \left[1 - \left(\frac{\Phi_2}{\Phi}\right)^2\right] \right] + \frac{3}{2} E \frac{I}{\Phi^2} \left[\Phi_2^2 - \Phi_1^2\right] \Phi_1 & \Phi > \Phi_2 \end{cases}$$
where \( I = b^2 / 12 \), and \( E \) is the modulus of elasticity in the linear regions.

The relation between the moment and curvature for different values of thickness, \( b \), is plotted in Figure 6-9. \( \Phi_1 \) and \( \Phi_2 \) decrease as the thickness, \( b \), increases. In a thicker beam the stress in the outer layer reaches the yield region, indicated by point 1 in Figure 6-7, at a smaller curvature. The properties of the material assumed for the following plot are as follows: \( \varepsilon_1 = 1\% \), \( \varepsilon_2 = 6\% \), and \( E = 45 \text{ GPa} \). [54].

![Figure 6-9: Variation of bending moment with increasing curvature.](image)

In a linear elastic material the moment is proportional to the curvature as shown in Eq. 6.13.

\[
M_o = EI\Phi
\]  

6.13

A plot of the effective linear modulus of the SMA beam in bending, obtained by dividing the moment by \( I\Phi \), is shown in Figure 6-10. The effective modulus is constant initially when the material is linear and, with increasing curvature, as the outer layers of the beam “yield” and the effective modulus decreases. Yielding of the outer layers occurs at a smaller curvature for thicker beams.
6.1.3 Design of the joint made of SMA

The objective is to design a beam made of SMA material to carry a load of 110 N with a tip deflection angle of 15°. The ultimate stress of the SMA beam was taken as 1150 MPa [54]. The length of the beam is fixed at 2cm and the variation of the maximum stress in beam with thickness, $b$, and Young’s modulus, $E$ is obtained as shown in Figure 6-11. The maximum stress in the beam increases with increasing modulus, and the variation with the thickness, $b$, is the same as before. The curvature at the tip, for different thickness values, is around 18. The effective modulus of the SMA beam for values of thickness ranging from 1.5 mm to 4 mm, was obtained from Figure 6-10. Using these values of modulus, the values of thickness for which the maximum stress in the beam was less than the ultimate strength of SMA (1150 MPa), were obtained from Figure 6-11. These values are shown in Figure 6-12 the rest of the values from the plot in Figure 6-11 are set to zero.

Figure 6-10: Effective linear modulus of the SMA beam for different thickness values.
Given an axial load of 110 N and a tip angle of 15° a compliant joint having a square cross-section can have a thickness ranging from 2 mm – 4 mm for a 2 cm length. Increasing the length decreases the curvature at the tip. The effective modulus of the SMA material is higher at a lower curvature. This results in a smaller feasible range.

Figure 6-11: Variation of the maximum stress in the beam with changing thickness and Young’s Modulus for an axial load of 110 N

Figure 6-12: Feasible thickness values of the SMA beam with maximum stress less than 1150 MPa
This design procedure gives a conservative estimate. The calculation of the effective modulus does not take into account the axial load acting on the joint. If the axial load was considered, the stress in the joint would already be in the pseudoelastic zone and the effective modulus would be lower. Hence the moment required to maintain the tip deflection and the maximum stress in the joint would be lower.

6.1.4 Summary: Compliant joint

The specifications of a compliant joint for the required load carrying capability and deflections were obtained. The design-specific joint procedure can be used to calculate the cross-section and length for other configurations. While the calculations are not generic in nature, it is possible to design compliant joints given the loading conditions as described above. A square cross-section was used in the above case, if the loading was doubled two such joints could be used in parallel or a rectangular cross-section with twice the breadth and the same thickness could be used.

The max load in a truss member in the representative vehicle discussed in chapter 4 is 1.3 \(10^5\) N. A compliant joint designed for this load was either too long (6 cm) for the truss member to be used in (13 cm) or too wide for the truss member (1.9 cm dia). Assuming that the weight of an aircraft scales as the cubed of the wing span, total weight and span of a similar plane with a max load of 110 N, were calculated as 1.15 kg and 0.433 m respectively. The size of the truss member for the small plane was calculated as 4 cm. So the compliant joint was designed with the loading condition used in the previous sections (Figure 6-1), but with a smaller range for the length of the beam. The size of a feasible compliant joint had a length of 4 mm and a rectangular cross-section with a thickness of 0.8 mm and breadth of 4 mm. The moment required to deflect the compliant joint was around 0.2 N-m. Given the size of the unit cell The additional force in the
cable is 10 N. Hence there is a 10 % to 20 % increase in actuation for due to the compliant joint.

Some of the issues regarding a morphing skin are discussed next.

6.2 Skin

The development of a skin system that can accommodate large shape changes while carrying and transferring aerodynamic loads is an essential element of aircraft morphing technology, posing perhaps one of the most difficult technical challenges. While the skin might not be required to carry high in-plane loads due to bending or torsion of the wing, it would be required to transfer the lateral aerodynamic loads to the underlying truss structure.

Low (or releasable) membrane stiffness is desirable in order to reduce morphing actuation requirements, while high lateral stiffness is desirable in order to maintain an aerodynamic shape while transferring load effectively. Several approaches to morphing skin structure could be explored.

High strain-capable materials offer the possibility of directly accommodating or withstanding the required morphing strains. Depending on the magnitude of these strains, such materials might include superelastic SMA, high-strength elastomeric composites, impermeable or tensioned fabrics. Low stiffness membranes, while attractive in principle, might require closely-space lateral support in order to minimize lateral deflection under pressure loads. Those that are especially stiff, such as SMA, might be best considered when inextensional skin deformation (bending only) is a possible morphing deformation. Vertifex™, a shape memory polymer resin, is capable of reaching up to 200% elongation when heated and when cooled it becomes stiff in the

™ Cornerstone Research Group Inc.
deformed configuration. Sheets or composites made of the material could also be used for the morphing skin.

*Segmented skins*, made of multiple discrete elements that slide relative to one another as the structure deforms, somewhat like fish scales or feathers, offer the possibility of relatively high local lateral stiffness in combination with low-force membrane deformation. Individual elements would be fairly stiff, capable of transmitting aerodynamic forces to the underlying structure. This approach seems particularly suited for use with a compliant cellular truss, in that each segment might be naturally attached to the nodes of a cell.

6.3 Summary

A compliant joint is designed for a representative vehicle discussed in chapter 4, as a beam with a square cross-section and made of pseudoelastic SMA. The design is carried out in two steps. In the first step, the maximum stress in a beam made of linear elastic material is determined for the required loading conditions. The variation of maximum stress in the beam with varying beam thickness, beam length and material Young’s Modulus is obtained. In the second step, an effective linear modulus is obtained for an SMA beam in bending, and the range of values for the thickness of the cross-section and length are determined such that the maximum stress is less than the ultimate stress of the SMA material. A few ideas of morphing skin were presented. The designs so far have focused on unmanned air vehicles. The next chapter studies the applicability of the unit cell concept on aircrafts of different sizes.
Chapter 7

Effect of Scaling

The size and weight of the representative wing designed in chapter 4 were based primarily on the NASA HECS wing. The octahedral unit cell was sized to carry the required loads but the weight of the actuation mechanism was not addressed. The weight of the actuators can potentially play a significant role in the design and sizing of the aircraft. One can intuitively say that as the size of the aircraft increases the actuation forces will become larger and it might not be possible to design tendon actuated truss structures for morphing of aircraft beyond a certain size. If this is so, at what size does morphing using such structures, make the most sense? This question is addressed in this chapter.

7.1 Maximum Forces Seen by the Actuators

The cables in the octahedral unit cell truss structure are in the load path. Hence, the forces in the cables are proportional to the weight of the plane. Consider a plane of gross weight $W$ with a tapered wing of span $b$. If the spanwise lift distribution is assumed to be proportional to the chord, the total moment at the root due to the lift at steady level flight is given by Eq. 7.1

$$M = \frac{Wb}{2} \left( \frac{1}{6} + \frac{2\alpha}{3} \right) \frac{1}{1+\alpha}$$  \hspace{1cm} 7.1

where $\alpha$ is the taper ratio (tip chord / root chord) of the wing, which varies from 0 for a delta wing, to 1 for a uniform wing.
The following assumptions are made in the scaling calculations:

1) The aspect ratio, $AR$, of the wing, i.e., the ratio of the span to the mean chord of the wing, remains constant as the size of the plane changes. The aspect ratios of wings usually vary from 6 – 10. However, for high speed attack aircraft, aspect ratios are around 4 – 6, and for long endurance (high finesse) aircraft like a glider, aspect ratios are around 15 and higher.

2) The ratio of the maximum thickness to chord of the wing, $\tau_c$, is also assumed to remain constant with varying size. The ratio of maximum thickness to chord is a property of the airfoil geometry. It is typically higher for low-speed airfoils and lower for high speed airfoils. It usually varies from 10 % – 15 % for subsonic airfoils, and is lower for transonic and supersonic airfoils.

Some cables in the octahedral unit cell structure are on the top and bottom surfaces of the wing. If the wing is actuated against the lift loads or if the cables are in the load path as in the octahedral cell structure, then the sum of the spanwise forces in the cables at the root, $F$, balance the moment due to the lift loads at the root as shown in Eq. 7.2

$$F \times t = M \quad \text{7.2}$$

$t$ is the thickness of the wing at the root. The thickness, $t$, is written in terms of the wing span, $b$, as shown in Eq. 7.3

$$t = \tau_c \times c_R = \tau_c \times \frac{b}{AR(1+\alpha)/2} \quad \text{7.3}$$

Combining equations Eq. 7.1, 7.2 and Eq. 7.3 gives Eq. 7.4
The sum of the forces in the cables increases proportionately with weight of the aircraft as the size of the aircraft increases, if the other factors in the equation remain the same.

### 7.2 Weight of Actuator with Varying Force

The maximum force seen by an actuator is thus a function of the weight of the aircraft. Marden has suggested that the maximum force output of a generalized motor can be related to the weight of the motor by a simple equation \[ \left[55\right]\). He has obtained this equation by considering both biological and mechanical motors over a large scale with sizes ranging from \(10^{-20}\) kg to \(10^5\) kg. The motors are grouped into two general categories:

**Group 1:** Motors that push or pull a steady translational (linear) load quasi-statically. This group of motors comprises single molecules (myosin, kinesin, dynein), muscle cells, whole muscles, winches, linear actuators and rockets. The maximum force output of these motors scale as \(\text{motor mass}^{0.67}\). The maximum force in these motors is hypothesized to be limited by the ultimate stress of the materials used.

**Group 2:** Motors that cycle at a steady rate and are used to move bodies in a fashion more complex than translation, have maximal force outputs that scale as \(\text{motor mass}^{1}\). This set of motors include the locomotor apparatus of flying birds bats and insects, swimming fish, running animals, piston engines, electric motors and all types of jets. Inertial forces dominate in these motors, and the fatigue life of the motor is suggested as the limiting factor of the size of these motors.
The motors used to actuate the cables are assumed to belong to group 1, since the actuation is linear and occurs at a slow rate. The maximum force, $F_{\text{max}}$ in newtons, of such motors scales with the motor mass, $Mm$ in kilograms, as shown in Eq 7.5.

$$F_{\text{max}} = 891(Mm)^{0.67} \tag{7.5}$$

### 7.3 Airplane Sizing Based on the Actuator Weight

The maximum actuation force required by a morphing aircraft using tendon-actuated cellular cable trusses is directly proportional to its weight as seen in Eq. 7.4. Using Eq 7.5 the weight of the motor that can output the required force can be calculated as

$$Mm = \left(\frac{F_{\text{max}}}{891}\right)^{\frac{1}{0.67}} = \left[\frac{9.8W}{891} \frac{AR}{4\tau_c} \left(\frac{1}{6} + \frac{\alpha}{3}\right)\right]^{1.493} \tag{7.6}$$

Assuming a single motor is used to actuate all the cables, if the weight of the motor is limited to be 10% of the weight of the aircraft (likely a very optimistic bound), then the upper limit of the weight of the aircraft is calculated as

$$\left[\frac{W}{891} \frac{AR}{4\tau_c} \left(\frac{1}{6} + \frac{\alpha}{3}\right)\right]^{1.493} = 0.1W$$

$$\Rightarrow W = \left[0.1\left\{\frac{AR}{36\tau_c} \left(\frac{1}{6} + \frac{\alpha}{3}\right)\right\}^{1.493}\right]^{2.02} \tag{7.7}$$

The aspect ratio of different aircraft varies from 6 – 10, the taper ratio, $\alpha$, varies from 0 – 1 and the thickness ratio, $\tau_c$, varies from 10 % – 15 %. Nominal values are used for the different parameters to estimate the weight of a morphing aircraft. These results are shown in Table 7-1.
The sizes of the planes that can be built using such structures are relatively small as seen in the table. This analysis is an order-of-magnitude analysis- many details have been omitted in these calculations. The actuation force in the above analysis was calculated only under steady level flight without load factors and a factor of safety.

The equation relating the force and the motors (Eq 7.5) was obtained using regression of data from a logarithmic plot of max-force of the motor vs. motor mass. There is a significant scatter of data around the regression line, approximately a ten fold range of variation at any given motor mass; that presumably reflects differences in material properties and mechanisms that affect stress resistance.

If a value of 6 is used for the ultimate load factor and the constant relating the maximum force in a motor to the motor mass \(^{0.687}\) (891), is increased 10 fold, the upper limit of the weight of the aircraft in Table 7-1 is 8000 kg (10% aircraft weight, AR = 6, \(\tau_c = 0.15\) and \(\alpha = 0\)). Since the weight of the actuators in a morphing aircraft is significant, the following ideas could be used to reduce the weight of the actuators.

### 7.3.1 Distributed actuation

The total weight of the actuators can be reduced using distributed actuation. All the actuators were assumed to be actuated using a single actuator and the weight of the actuator was given by Eq 7.5. If \(N_c\) actuators, distributed in the chordwise direction, are used then the

<table>
<thead>
<tr>
<th>Weight of the actuator as a percentage of total weight</th>
<th>Using (AR = 7, \tau_c = 0.12) and (\alpha = 0.6)</th>
<th>Using (AR = 10, \tau_c = 0.1) and (\alpha = 1)</th>
<th>Using (AR = 6, \tau_c = 0.15) and (\alpha = 0)</th>
</tr>
</thead>
<tbody>
<tr>
<td>5%</td>
<td>4.14 kg</td>
<td>0.94 kg</td>
<td>408 kg</td>
</tr>
<tr>
<td>10%</td>
<td>16.79 kg</td>
<td>3.80 kg</td>
<td>1656 kg</td>
</tr>
</tbody>
</table>
maximum force carried by each of these actuators is divided by $N_c$, since they act in parallel. The total weight of the actuators these actuators is calculated using as:

$$\left( Mm \right)_{\text{distributed}} = N_c Mm_c = N_c \left( \frac{F}{N_c} \right)^{1.49} = \frac{1}{N_c^{0.49}} \left( \frac{F}{891} \right)^{1.49} \approx \frac{(Mm)_{\text{single}}}{\sqrt{N_c}} \quad 7.8$$

where $Mm_c$ is the weight of each actuator, $(Mm)_{\text{distributed}}$ is the total weight of the distributed actuators, and $(Mm)_{\text{single}}$ is the weight of a single actuator used to carry the same load. The total weight of the actuators is reduced by a factor of $\sqrt{N_c}$. However, if multiple actuators are used in the spanwise direction, and the required maximum force output of the actuators remains the same, i.e., the actuators act in series, then the total weight of the actuators increases.

### 7.3.2 In-plane morphing using beams

The actuators in the above case were either in the load path or the morphing was against the lift loads. So the actuation force in the cables was significantly high. However the morphing could be limited to in-plane morphing and the structure could be built of beams to carry the bending loads generated by lift. The actuating cables could be located at or close to the neutral axis of the beams. The forces in the cables would then be primarily due to the in-plane loads (drag forces) which are much lower than the lift loads. This would lead to a lower actuator weight. The parallel genetic algorithm could be used in the design of such structures as discussed in chapter 5.
7.4 Summary

The scaling of actuator requirements with aircraft size has been analyzed. If the actuators need to carry the lift loads of the plane, then the forces in the actuators can be related to the weight of the plane. The weight of the actuators is calculated based on the maximum force of the actuator using a scaling relation proposed by Marden. The weights of planes that can be built using such structures vary from tens to thousands of kilograms if the actuator weight is 10% of the aircraft weight. This estimate of scaling would be valid for any morphing technique that involves actuation of the wings against the lift loads. The weight of the actuators could be reduced using distributed actuators in parallel. Using 2D beams with in-plane actuation of the wings, so that the loads in the cables are mainly due to the drag forces, could significantly reduce the forces in the cables, hence reducing the actuator weight.
A new structural concept to build morphing aircraft structures using cables and truss structures was developed and explored. The truss structure is connected using compliant joints such that only modest bending moments are transmitted from one member to another. The structure formed by the truss members is essentially a compliant mechanism. Two sets of cables actuate the structure. One set morphs the structure into a required shape while the other set returns it to the original shape. When one set of cables is reeled in the other set of cables is released in a controlled manner. Hence the stability of the structure can be maintained in any intermediate position.

The morphing HECS wing from NASA was used as an example to guide the development of the structural concept. The deformation in the wing was primarily a bending type deformation. On morphing the span-wise and chord-wise strains on the top surface were expansive and on the bottom surface the span-wise strains were contractive and the chord-wise strains were expansive. An effective way of accomplishing the required deformation is to use a different layer of unit cells for each of the top and bottom surfaces.

A genetic algorithm was developed to design the unit cell using topology optimization. A reduced ground structure is used to formulate the structural layout. Each member of the ground structure has four possibilities: void; truss element; cable 1, which morphs the structure into the required shape; and cable 2, which returns the structure back to its original shape. The number of possible layouts increases exponentially with the number of structural members. For example, if the ground structure has 20 members, then the total number of possible structural layouts is $4^{20}$.
(1.0995e12). The problem was parallelized to increase the calculation speed, making it possible to analyze a large number of fitness functions in a reasonable time. The effective computation speed increased up to 80 times due to parallelization.

Another technique used to increase the size of the ground structure without increasing the problem size was enforcing symmetry in the ground structure. In this case only one-eighth of the structure needed to be designed in the 3D case, using symmetry about three orthogonal planes.

In the genetic algorithm developed, the variables corresponding to the structural members are concatenated to form a “chromosome.” Each chromosome is a sequence of numbers with \( n \) digits, where \( n \) is the total number of variables. Each chromosome represents a unique structural layout. A group of these chromosomes form a population. The chromosomes of the initial population are generated randomly. This initial population forms a “parent” mating pool from which “child” chromosomes are generated using processes of selection, crossover and mutation. Features of a genetic algorithm such as population size and mutation parameter determine its convergence characteristics. Micro GA, a technique to search a larger design space, was used to improve the performance of the algorithm. The mutation parameter was also varied between 0.1 and 0.001 to balance the diversification and selectivity characteristics of the algorithm. These parameters were determined after studying the convergence pattern of the results. If a population becomes uniform very early in the calculations, then those parameters that increase diversity, such as the mutation parameter, are increased. And other parameters that increase the selection pressure, such as the number of elite chromosomes chosen to be added directly to the child population in each generation, the number of chromosomes that are carried over into the next random population in the micro GA, or the number of generations after which a new population is generated in micro GA, are decreased.

A fitness function represents how well each chromosome meets the design requirements. The fitness function used was formulated to pursue the uniform morphing of all the points on the
face of a unit cell; it includes actuation using both sets of cables. All the cables are set to voids in the actuation phase. The forces in the cables are optimized so that the structure morphs “towards” the required deformation in the best way available. The fitness function due to actuation is a continuous, differentiable function of the actuation forces in the cables. Hence a gradient-based approach is used to calculate the optimal forces in the cables. The corresponding fitness, attributed to the morphing ability of the unit cell calculated using this optimized set of forces, represents the best value for that structure.

Stability of the cable-truss structure on application of external load is also addressed in the fitness function. The cables are not pre-stressed and cannot carry compressive loads. External loads are applied to the unit cell at points where the unit cell is connected to other unit cells. Stability is checked by applying a unit force along each of the three coordinate axes in both positive and negative directions at each connection point individually. Cables under compressive loading conditions are set to voids in an iterative calculation of fitness. If the unit cell is stable under all these external loads, then the unit cell will be stable under any combination of these loads.

An octahedral unit cell was obtained as a result from the genetic algorithm. The cell geometry and the orientation are determined from the morphing strain requirements.

A wing made of these octahedral unit cells was sized for a representative vehicle having a gross mass of 1360 kg (3000 lbs). The wing was analyzed using load factors of ±4 and a factor of safety of 1.5. The truss members were sized to prevent local failure due to buckling and ultimate stress in the material, and the cable members were sized to keep the stress in the cables below the ultimate tensile strength of the material. An additional 8.16 kg (18 lbs) was added to weight of the truss wing to represent the weight of the nodes, based on an assumption that the nodes are solid and spherical. The weight of the wing including the weight of the nodes was about 41 kg (90 lbs). The weight of the skin and actuators was not included in the above weight. The deflection of the
morphing truss wing was comparable to that of a conventional stiff wing without the actuators sized for the same loads. The deflection of the wing at the tip was around 68% of the chord. The deflections in the truss wing are probably larger than those of conventional wings. However, this might be acceptable since, in conventional wings the hinges of the control surfaces bind and limit the maximum allowable deflection; this will not be an issue in a morphing wing. Aeroelastic instabilities such as divergence and flutter are concerns which could perhaps be addressed using active control. However, the wing is not especially soft in torsion. The divergence speed of the wing calculated using an approximate method was found to be high ($1.28 \times 10^5$ m/s).

A procedure was developed to estimate the size of a compliant joint made of pseudoelastic shape memory alloy given the axial load it must transmit, along with the rotational deformation it must accommodate. The design is carried out in two steps. In the first step, the maximum stress in a beam made of a linear elastic material is determined for the required loading conditions. The variation of maximum stress in the beam with varying thickness, length and Young’s Modulus is obtained. In the second step, an effective linear modulus is obtained for an SMA beam in bending, and the range of values of thickness and length are determined such that the maximum stress is less than the ultimate stress of the SMA material. The maximum load in a truss member in the representative vehicle is $1.3 \times 10^5$ N. A compliant joint designed for this load was either too long (6 cm) for the truss member it was to be used with (13 cm) or too wide (6 cm) for the truss member (1.9 cm dia). If the length of the compliant joint was limited to 2.5 cm the maximum rotation angle it could accommodate was $4^\circ$. A compliant joint was successfully designed for a smaller vehicle weighing about a kilogram. In this case, the size of a feasible compliant joint had a length of 4 mm and a rectangular cross-section with a thickness of 0.8 mm and breadth of 4 mm.

An analysis of the scaling of actuator requirements with aircraft size was done assuming single actuator was assumed to balance the moment due to lift at the root. The load carried by the
actuator is function of the wing shape: aspect ratio, thickness to chord ratio and taper ratio. The aspect ratio was assumed to vary between 6 and 10, the thickness to chord ration between 10 % and 15 %, and the taper ratio between 0 and 1. This analysis showed that if the morphing of the wing works against the lift loads, then the size of the aircraft that can be built using a reasonable actuation weight fraction, is limited to something in the range of a few kilograms to a few thousand kilograms. The morphing wing for the representative vehicle, built using octahedral unit cells, falls in this category since the active cables are in the load path on the top and bottom surface of the wing. This analysis would also hold true for morphing concepts that involve changing the shape of the wing against the lift loads. The weight of the actuators could be reduced somewhat by using distributed actuators in parallel.

A design methodology that could be used for the development of morphing aircraft structures is as follows:

1. Develop a structural concept/design which can achieve the required morphing as well as carry the air loads. This was done both intuitively using reasoning and using topology optimization. The parallel genetic algorithm tool developed in this thesis could be used to investigate more concepts using topology optimization.

2. Determine the shape and size of the structure from the exact morphing requirements and loading condition of the vehicle.

3. Design the skin locking and actuation mechanisms.

4. Address aeroelastic and dynamic concerns.

5. Address failure and redundancy concerns.
8.1 Conclusions and Future Work

The octahedral unit cell was obtained as an optimum design for a 3D unit cell with an expansive morphing capability. A wing made of these unit cells sized for a representative vehicle showed that the weight of a morphing truss wing is comparable to that of a conventional wing. The weight and complexity could be reduced by using these unit cells primarily in regions where the morphing strains are large. However, analyses of compliant joints and actuator weight indicate that a 3D cellular truss structure might not be efficient when only in-plane morphing is required; since, the size of the unit cells becomes small when large changes in area are required and the actuation of the truss structures involves high actuation forces in the cables.

A better way to design morphing structures for large changes in area is to use pin/compliant joint-connected beams to carry the lift loads, but cables for in-plane actuation. The actuating cables then act only against the in-plane loads which are much lower (approximately 10% of those associated with lift). The genetic algorithm can easily be modified to design such structures. Preliminary results using this new approach are shown for a morphing problem involving sweep and span reduction. The topology obtained does not morph into the required shape. It does move in the direction required.

The in-plane morphing change required is large and a linear analysis does not capture these large changes in configuration. Step-wise actuation could perhaps be used to analyze the deformation due to actuation. The fitness due to actuation calculated using this piece-wise linear analysis would be a better representation of the actual case than that calculated using simple linear analysis.

The solution obtained from the genetic algorithm in general depends on the form of the fitness function and the ground structure used. The genetic algorithm can only choose designs that can be described by the ground structure. Hence the ground structure should be as general as
possible in order to include many possible types of structures. This would improve the results obtained from the ground structure. In the 2D case analyzed in this thesis, only small beams were used which might not be the most efficient ground structure. A method to include longer beams using spring elements at the end was also described. The ground structure does not yet include the possibility of having linkages. It might be preferable at times to have longer cables. Even this feature could be included in the ground structure to improve the results.

In the 2D structure, the wing was required to morph into a specific shape. This requirement could perhaps be relaxed and an area change included in the fitness function. The problem could then be constrained to guide the development of a feasible design.

Apart from the structural concept, there are additional issues that need to be addressed before morphing aircraft become feasible. A skin is needed that can transfer air loads to the substructure as well as have a low in-plane stiffness to morph easily. Detailed designs of the different mechanisms involved such as the actuation mechanism and the locking mechanism also need to be developed. The aeroelastic and dynamic issues of the wing also need to be addressed. These issues can be addressed fully when weight distribution of the wing is known including the actuators, skin, locking mechanisms etc. Redundancy and failure should also be investigated. Redundancy is a greater challenge when morphing structures are involved since redundancy and morphing are opposing requirements. In cable actuated trusses additional cables of fixed length could be used to provide redundancy, these cables are slack in normal operation and become taut when failure occurs and the deflection increases beyond a certain limit.

This investigation into smoothly morphing aircraft structures has yielded insight into the general problem, design procedures and a powerful tool for the design of morphing aircraft structures using cables, truss elements and beams. There are still many challenges to be addressed before such structures become practical.
Bibliography


6. Linden, A W; et al “Variable diameter rotor study (Aerodynamic characteristics of three types of variable diameter rotary wings with telescoping blades, folding rigid blades, and flexible blades) [Final Report, Jan. - Jul. 1971]”


19. Information provided by Dave Cowan of NextGen Aeronautics, Inc., 2780 Skypark Drive, Suite 400, Torrance, CA 90505-7519.


34. Stinton, D., The design of the aeroplane : which describes common-sense mechanics of design as they affect the flying qualities of aeroplanes needing only one pilot, Van Nostrand Reinhold, New York, 1983.


Appendix A

Mechanism Modes

The octahedral unit cell is a statically determinate structure. However when the octahedral unit cells are linked to form an extended structure, the structure can have an inextensional mode of deformation, i.e., it can distort without any change in the length of the members, even though a unit cell is statically determinate by itself. For example, the structure in Figure 4-3 can fold hinging about the axis denoted by the arrow. The existence of the inextensional modes can verified using the method provided by Pellegrino and Calladine [33]. Equations of kinematics of small displacements of the assembly are written as follows. For each bar there is one equation relating its elongation, $e$, to the components of displacements of the joints, $d$, at either end; and the resulting equations may be written as Eq. 2.9.

$$B \cdot d = e$$ \hspace{1cm} \text{A1.1}

The nullspace of the kinematic matrix, $B$, gives the inextensional modes of the structure, i.e., a set of vectors of the displacements at nodes that do not cause the extension of members. This method does not account for cables being slack under compression. In this approach the unit cell is formed by taking a unit cube and placing the octahedral unit cell in the center and adding additional members appropriately. A cable or a truss is added following the octahedral truss pattern, after calculating the mechanism modes and observing the mode of deformation. The unit cell required to form a stable single-layer truss plate includes additional members along the top and bottom surface of the unit cell, as shown in Figure A1-11. The bottom-surface unit cell shown in Fig. 6 is seen in the centre of the cell in Figure A1-11.
The solid lines are truss members and dashed lines are the cables. The inner octahedral cell is represented using thicker lines for clarity.

Figure A1-1: Unit Cell that is integrated into the structure
Appendix B

The C++ Code for the Parallel Genetic Algorithm

There are two parts to this code, one that does the Genetic Algorithm calculations and the parallelization (parallelGA.C); and the other that does the fitness calculation (chromosomeMean.C).

Parallel GA.C

#include "ga_routine.h"
#include <assert.h>
using namespace std;
#define fileName "output_test195.txt" /*Output File*/
#define fileName1 "166gene.txt" /*start seed values*/
#define fileName2 "195Continue.txt" /*All chromosomes in each generation are stored in this file in case the program hangs. This file could be used to restart the calculation*/

enum {DO_WORK = 0, DO_EXIT = 1};
bool first_time1;

int main (int argc, char *argv[])
{
    time_t startTime = time (NULL);

    /* Declaration of Variables common to all the nodes ie that all the nodes need */
    int numProcs; // number of Processors
    int myRank; // My processor's rank

    MPI_Status status;

    /* End of Declaration */

    /* MPI START UP */
    cout << "\t\t ** Genetic Algorithm Implementation **\n";

    MPI_Init (&argc, &argv);

    /* Get my processor's rank */
    MPI_Comm_rank (MPI_COMM_WORLD, &myRank);

    /* Find out total number of processors */
MPI_Comm_size (MPI_COMM_WORLD, &numProcs);

if (myRank == 0)
{
    /* ################ MASTER NODE CALCULATIONS ################ */
    /* The master node does all the GA calculations. Sends the genes to the worker nodes for them to calculate the fitness */

    cout << fileName << endl; // to check the output

    /* Declaration Section -- Start */
    const int populationSize = 1000;
    const int noOfGen = 2000;    /* Number of Generations */

    const int elitism = 1;    /* Number of parents sent directly to the child pool */
    const int numCrossover = 2;    /* Number of crossover points: can be 1 or 2 */
    double mutValue = 0.1;    /* Probability that mutation occurs on a gene */

    Chromosome parent[populationSize];    /* The parent pool*/
    Chromosome *sortedParent = new Chromosome[populationSize];    /*The sorted parent pool*/
    Chromosome *child = new Chromosome[populationSize];    /* The child pool*/

    int typesOfGenes = parent[0].typesOfGenes;    /* The number of possibilities for a gene*/
    int calcCounterI = 0, calcCounterG = 0;    /* The variables that keep track of the number of chromosomes generated initially and then in each generation to estimate the number of feasible chromosomes*/

    MPI_Request *rq;
    MPI_Status status;
    MPI_Datatype reqd, ackd;    /* Data types to facilitate communication*/
    struct request *req;    /* The structure corresponding to the datatypes*/
    struct ack *ack;

    int completed = 0, pn;    /* completed genes and processor number*/

    rq = (MPI_Request *) calloc(numProcs, sizeof(MPI_Request));
    if (rq == NULL) {
        cout << "Could not allocate\n";    // fprintf(stderr, "Could not allocate\n");
        return -1;
    }

    req = (struct request *) calloc(numProcs, sizeof(struct request));
    if (req == NULL) {
        cout << "Could not allocate\n";    // fprintf(stderr, "Could not allocate\n");
        return -1;
    }

    ack = (struct ack *) calloc(numProcs, sizeof(struct ack));
    if (ack == NULL) {

cout << "Could not allocate\n";  // fprintf(stderr, "Could not allocate\n");
return -1;
}
compose_dtypes(reqd, ackd, noOfElemG);

/* Declaration Section -- End */

/*Opening file for output */
ofstream FileOutput;
FileOutput.open (fileName, ofstream::out | ofstream::app);
ofstream FileContinue;

/*Read connectivity and nodal data for the class */
int retVal = parent[0].ReadFiles (myRank);
if (retVal == 1)
{
    for (int dest = 1; dest < numProcs; dest++)
    {
        req[dest].command = DO_EXIT;
        MPI_Send (&req[dest], 1, reqd, dest, 0, MPI_COMM_WORLD); /* done =1 so it
            tells the workers to quit */
    }
    MPI_Finalize ();
    return 1;        /* Exit the program if error*/
}

/*Calculate the element lengths and direction cosines for the class */
// parent[0].ElLengthCalc ();

/* Initial Population Generation -- Start */

srand((unsigned)time(NULL));

Array1D < int >genest (noOfElemG, 0);    /* temporary variable to store the randomly
    generated gene */
Array1D < double >fitness (populationSize, 0.0);     /* array of fitnesses */
Array1D < int >address (populationSize, 0);    /* Array to store the addreses of the presorted
    parents after sorting */

/*Reading the first gene from a file*/
ifstream FirstGene(fileName1);

/* Feed the data to the slaves */
int assign = numProcs;// <populationSize ? numProcs : populationSize;
if (assign > 0) {
    for (pn = 1; pn < assign; pn++) { /*Sending the first set of chromosomes to all the processors. Note it is important that the population size should be larger than the number of processors. It will work otherwise too but it's waste. */
        req[pn].command = DO_WORK;
        //    if(pn==1)
        //      {
        //      //    for (int j = 0; j < noOfElemG; j++)
        //    //      {      //    FirstGene >> genest[j] ; //temporaryVariable >>
        //      //    req[pn].tmpgen[j] = genest[j];
        //      //    }
        //    //else
        //    }
        //  */Creation of a random chromosome*/
        for (int j = 0; j < noOfElemG; j++)
        {
            genest[j] = (int) (((double)typesOfGenes + 3.0)*rand()/(RAND_MAX+1.0));
            //rand ()% (typesOfGenes+3);
            //if ((genest[j]==typesOfGenes)||(genest[j]==typesOfGenes+1))
            //genest[j]=0;
            //if (genest[j]==typesOfGenes+2)
            //genest[j]=1;
            //req[pn].tmpgen[j] = genest[j];
        }
        MPI_Send(&req[pn], 1, reqd, pn, 0, MPI_COMM_WORLD);  /*Sends the chromosome to the slave*/
        MPI_Irecv(&ack[pn], 1, ackd, pn, 0, MPI_COMM_WORLD, &rq[pn]); /*Implements a non blocking receive*/
        calcCounterI++;  /* Increments the initial chromosomes counter */
    }
    int completedProcessors=1; /*counter that keeps track of processor that have completed and returned the fitness value after the population limit of feasible chromosomes has been reached*/

while (completedProcessors < numProcs)
{
    MPI_Waitany(numProcs, rq, &pn, &status);
    //    cout << __LINE__ << " Checking pn "<<pn <<"\t"<< "completedProcessors =" << completedProcessors"<<" " completed=" << completed << endl;
if (ack[pn].feasibility == 1 && completed < populationSize)
{
    for (int i = 0; i < noOfElemG; i++)
    {
        genest[i] = req[pn].tmpgen[i];
    }

    parent[completed].SetGenes (genest);
    fitness[completed] = ack[pn].fitness;
    completed++;
}

/* Give pn some more work if needed */
if (completed < populationSize) /*completed is incremented only if parent is feasible*/
{
    req[pn].command = DO_WORK;

    for (int j = 0; j < noOfElemG; j++)
    {
        genest[j] = (int) (((double)typesOfGenes +
        3.0)*rand()/(RAND_MAX+1.0)); //rand ()% (typesOfGenes+1);
        if ((genest[j] == typesOfGenes) ||(genest[j] == typesOfGenes+1))
            genest[j]=0;
        if (genest[j] == typesOfGenes+2)
            genest[j]=1;
        req[pn].tmpgen[j] = genest[j];
    }

    MPI_Ssend(&req[pn], 1, reqd, pn, 0, MPI_COMM_WORLD);
    MPI_Irecv(&ack[pn], 1, ackd, pn, 0, MPI_COMM_WORLD, &rq[pn]);
    calcCounterI++;
    }
else
    completedProcessors++;
}

FirstGene.close();
/* Initial Population Generation -- End */
cout << "nCalcCounterI \t"<<calcCounterI << endl;
calcCounterG = calcCounterI;

for (int cGen = 0; cGen <= noOfGen; cGen++)
{
    pn = 1; completedProcessors = 1; completed=0;
/* Sort the population according to the fitness */
for (int i = 0; i < populationSize; i++)
    address[i] = i;
QuickSort (fitness, address, populationSize);

for (int i = 0; i < populationSize; i++)
    sortedParent[i].SetGenes (parent[address[i]].genes);

/*End of sorting */

/* Store the best gene and the corresponding fitness in a file */
FileOutput << eGen << 't' << time(NULL)-startTime << 't' << calcCounterG << 't'
<< fitness[populationSize - 1] << 't';
for (int i = 0; i < noOfElemG; i++)
    FileOutput << sortedParent[populationSize - 1].genes[i] << 't';
FileOutput << endl;
calcCounterG = 0;

if (cGen%50==1)
{
    FileOutput.close();
    FileOutput.open (fileName, ofstream::out | ofstream::app);
}

//***** Storing each generation in a file for continuous runs
{
    FileContinue.open (fileName2, ofstream::out | ofstream::trunc);
    for (int i = 0; i < populationSize; i++)
    {
        FileContinue << fitness[i] << 't';
        for (int j = 0; j < noOfElemG; j++)
            FileContinue << sortedParent[i].genes[j] << 't';
        FileContinue << endl;
    }
    FileContinue.close();
}

if (cGen==noOfGen) break;
if (cGen%80==0 && cGen>30)
{
    cout << "mutvalue = " << mutValue << endl; mutValue = mutValue*0.5;
}
if (cGen%200==0 && cGen>=100){
    / Micro GA */
    mutValue=0.1;
int counterDiffGene = populationSize;
Array1D<int> tempGene(noOfElemG,0);
int flag = 1;
for (int i = 1; i <= 5; )
{
  if (counterDiffGene == 0)
  {
    parent[populationSize-i].SetGenes(sortedParent[counterDiffGene].genes);
    break;
  }
  flag = 1;
  counterDiffGene--;
  if (i==1){
    parent[populationSize-i].SetGenes(sortedParent[populationSize-i].genes);
    i++;
  }
  else
  {
    tempGene = sortedParent[counterDiffGene].genes.copy();
    for(int ii=0; ii<noOfElemG;ii++)
      if (tempGene[ii]!=parent[populationSize-1].genes[ii])
        flag = 0;
    if(flag==0)
    {
      parent[populationSize-i].SetGenes(tempGene);
      i++;
    }
    else continue;
  }
}

/*Assigning random genes to the rest of the genes other than the top 2*/

for (pn = 1; pn < numProcs; pn++) {
  req[pn].command = DO_WORK;
  for (int j = 0; j < noOfElemG; j++)
  {
    genest[j] = (int) (((double)typesOfGenes + 3.0)*rand()/(RAND_MAX+1.0));
    if ((genest[j]==typesOfGenes) || (genest[j]==typesOfGenes+1))
      genest[j] = 0;
    if (genest[j]==typesOfGenes+2)
      genest[j] = 1;
    req[pn].tmpgen[j] = genest[j];
  }
  MPI_Send(&req[pn], 1, reqd, pn, 0, MPI_COMM_WORLD);
  MPI_Irecv(&ack[pn], 1, ackd, pn, 0, MPI_COMM_WORLD, &rq[pn]);
calcCounterG++;  
}

while (completedProcessors < numProcs) 
{

    MPI_Waitany(numProcs, rq, &pn, &status);

    if (ack[pn].feasibility == 1 && completed < populationSize-5) 
    {
        for (int i = 0; i < noOfElemG; i++)
        {
            genest[i] = req[pn].tmpgen[i];
        }  

        parent[completed].SetGenes (genest);
        fitness[completed] = ack[pn].fitness;
        completed++;
    }

    /* Give pn some more work if possible */
    if (completed < populationSize-5) /*completed is incremented only if parent is feasible*/
        req[pn].command = DO_WORK;

    for (int j = 0; j < noOfElemG; j++)
    {
        genest[j] = (int)(((double)typesOfGenes + 3.0)*rand()/(RAND_MAX+1.0)); // rand()% (typesOfGenes+1);
        if ((genest[j]==typesOfGenes)||(genest[j]==typesOfGenes+1))
            genest[j]=0;
        if (genest[j]==typesOfGenes+2)
            genest[j]=1;
        req[pn].tmpgen[j] = genest[j];
    }

    MPI_Send(&req[pn], 1, reqd, pn, 0, MPI_COMM_WORLD);
    MPI_Irecv(&ack[pn], 1, ackd, pn, 0, MPI_COMM_WORLD, &rq[pn]);
    calcCounterG++;
}

else
    completedProcessors++;
else { // Normal Ga procedure

    /*Elitism*/
    for (int i = 1; i <= elitism; i++)
        child[populationSize-i].SetGenes (sortedParent[populationSize - i].genes);

    /*Getting children*/
    while (completedProcessors < numProcs) /*completed is incremented only if parent is feasible*/
    {

        /*Parent Selection*/
        /* using the roulette wheel method*/

        int chosenParent[2];
        ParentSelection(chosenParent,populationSize);
        Array2D<int> parentGene(2,noOfElemG,0);

        for(int k=0;k<2;k++)
            for(int j=0;j<noOfElemG;j++)
                parentGene[k][j]=sortedParent[chosenParent[k]].genes[j];

        /*Crossover*/

        int r[numCrossover];

        CrossoverPoints(r,numCrossover,noOfElemG); // chose crossover points randomly

        for (int k = 0; k < 2; k++) /* two children from two parents*/
        {

            SwapGenes(k, numCrossover, r, noOfElemG, genest, parentGene);

            /*Mutation*/
            Mutation(noOfElemG,mutValue,genest);
            /*End of Mutation*/

            /*Sending to the worker nodes for fitness calculation*/

            if(calcCounterG<numProcs-1)
            {

                req[pn].command = DO_WORK;
            }
        }
    }
}
for (int i = 0; i < noOfElemG; i++)
{
    req[pn].tmpgen[i] = genest[i];
}
MPI_Send(&req[pn], 1, reqd, pn, 0, MPI_COMM_WORLD);
MPI_Irecv(&ack[pn], 1, ackd, pn, 0, MPI_COMM_WORLD, &rq[pn]);
calcCounterG++;
pn++;
}
else{
    /*receiving data Sending work to any processors that have completed work*/
    MPI_Waitany(numProcs, rq, &pn, &status);
    if (ack[pn].feasibility == 1 && completed < populationSize-elitism)
    {
        for (int i = 0; i < noOfElemG; i++)
        {
            child[completed].genes[i] = req[pn].tmpgen[i];
        }
        fitness[completed] = ack[pn].fitness;
        completed++;
        // cout << __LINE__ << " Checking pn " << pn << "t" << 
        "completedProcessors =" << completedProcessors << " completed= " << completed << 
        " numprocs= " << numProcs << endl;
    }
    /* Give pn some more work if possible */
    if (completed < populationSize-elitism) {
        req[pn].command = DO_WORK;
        for (int j = 0; j < noOfElemG; j++)
        req[pn].tmpgen[j] = genest[j];
        MPI_Send(&req[pn], 1, reqd, pn, 0, MPI_COMM_WORLD);
        MPI_Irecv(&ack[pn], 1, ackd, pn, 0, MPI_COMM_WORLD, &rq[pn]);
calcCounterG++;
    } else {
        completedProcessors++;
        break;
    }
for (int i = 0; i < populationSize; i++)
    parent[i].SetGenes(child[i].genes);

} /* End of non random children*/

} /* End for number of generations */

//cout << "About to send done \n";
for (int dest = 1; dest < numProcs; dest++){
    req[dest].command = DO_EXIT;
    MPI_Send (&req[dest], 1, reqd, dest, 0, MPI_COMM_WORLD); // done =1 so it
tells the workers to quit
}
delete[]child;
delete[]sortedParent;
free (rq); free (req); free (ack);
MPI_Type_free(&reqd); MPI_Type_free(&ackd);

FileOutput.close (); // Close the files
MPI_Finalize ();

} else
{
    Chromosome worker;
    Array1D < int >genest (noOfElemG, 0);

    MPI_Status status;
    MPI_Datatype reqd, ackd;
    struct request req;
    struct ack ack;

    compose_dtypes(reqd, ackd, noOfElemG);

    assert (noOfElemG != 0);

    /*End of Declaration*/*
/*Read connectivity and nodal data for the class*/
int retVal = worker.ReadFiles (myRank);
if (retVal == 1)
{
    MPI_Finalize ();
    return 1;  // Exit the program if error
}

/*Calculate the element lengths and direction cosines for the class*/
//worker.ElLengthCalc ();

int recvd;
//    int ttt = MPI_Get_elements (&status, MPI_INT, &recvd);
//    cout << "get elements " << ttt << "Mpi success = "<<MPI_SUCCESS << endl;
//    if (MPI_Get_elements (&status, MPI_INT, &recvd) != MPI_SUCCESS)
//    {
//        cerr << "Holy cow! MPI_Get_elements() bailed out'n";  
//        MPI_Finalize ();
//        exit (1);
//    }

while (1)
{
    //cout << __LINE__ << " Worker about to recv a gene" << endl;

    MPI_Recv(&req, 1, reqd, 0, 0, MPI_COMM_WORLD, &status);
    if (req.command == DO_EXIT) {
        break;
    }

    for (int i = 0; i < noOfElemG; i++)
        genest[i] = req.tmpgen[i];

    worker.SetGenes (genest);

    ack.fitness = worker.Fitness (myRank);

    ack.feasibility = worker.feasibility;
    //    cout << __LINE__ << " Worker parallel " << myRank << "feasibility " << ack.feasibility << endl;

    //    if (worker.nActive==0)
    //{
    //    cerr << "Holy cow! nActive is 0\n";
#include "chromosome.h"
#include "mpi.h"

inline void ParentSelection(int cParent[], int pSize);
inline void CrossoverPoints(int rr[], int nCrossover, int nElem);
inline void SwapGenes(int p, int nCrossover, const int rr[], int nElem, Array1D<int> tempGenes,
const Array2D<int> pGene);
inline void Mutation(int nElem, double mValue, Array1D<int> tempGene);

void QuickSort (Array1D<double>fitness, Array1D<int>address, int array_size);
void Q_sort (Array1D<double>fitness, Array1D<int>address, int left, int right);

const int noOfElemG = Chromosome::noOfElemG; //6 20 28 158
struct request {
    int command;
    int tmpgen[noOfElemG];
};

struct ack {
    double fitness;
    int feasibility;
    // int tmpgen[noOfElemG];
};

static int compose_dtypes(MPI_Datatype &reqd, MPI_Datatype &ackd, int nElemG)
{
    struct request {
        int command;
        int tmpgen[noOfElemG];
    };

    MPI_Finalize ();  // End of MPI commands
    return 0;
}
struct ack {
    double fitness;
    int feasibility;
    int tmpgen[noOfElemG];
}; */

int req_num[3], ack_num[4];
MPI_Aint req_off[3], ack_off[4];
MPI_Datatype reqtypes[3], acktypes[4];

struct request foo;

req_num[0] = 1; req_off[0] = 0; reqtypes[0] = MPI_INT;
reqtypes[1] = MPI_INT;
MPI_Type_struct(3, req_num, req_off, reqtypes, &reqd);
MPI_Type_commit(&reqd);

struct ack fui;

ack_num[0] = 1; ack_off[0] = 0; acktypes[0] = MPI_DOUBLE;
ack_num[1] = 1; ack_off[1] = (unsigned long) &fui.feasibility - (unsigned long) &fui;
acktypes[1] = MPI_INT;
acktypes[2] = MPI_INT;
MPI_Type_struct(3, ack_num, ack_off, acktypes, &ackd);
MPI_Type_commit(&ackd);
return 0;
}

inline void ParentSelection(int cParent[], int pSize)
{
    for (int k = 0; k < 2; k++)
    {
        double r = (((double) rand () / (RAND_MAX +1.0)) * ((double) pSize * ((double) pSize + 1.0) / 2.0 ));//rand ()% ( pSize * (pSize + 1) / 2);;//
        double sums = 0;
        for (int j = 1; j <= pSize; j++)
        {
            sums = sums + j;
            if (r <= sums)
            {
                cParent[k] = j - 1;
                break;
            }
        }
    }
if (k == 1 && cParent[0] == cParent[1])
    k = 0;
}

inline void CrossoverPoints(int rr[], int nCrossover, int nElem)
{
    for (int i = 0; i < nCrossover; i++)
    {
        rr[i] = (int) floor ((double) rand() * (nElem - 1) / (RAND_MAX + 1.0)); // rand()%nElem://
        if (nCrossover == 2 && i == 1 && rr[1] == rr[0])
            i = 0;
        /* if (rr[i] == nElem - 1)
            rr[i] = nElem - 2; */
    }

    if (nCrossover == 2 && rr[1] < rr[0]) /* sort(r) */
    {
        int temp = rr[1];
        rr[1] = rr[0];
        rr[0] = temp;
        /* swap */
    }
}

inline void SwapGenes(int p, int nCrossover, const int rr[], int nElem, Array1D<int> tempGenes,
const Array2D<int> pGene)
{
    int index1, index2;
    for (int i = 0; i < nCrossover + 1; i++) /* swapping of genes for formation of a child */
    {
        if (i == 0)
        {
            index1 = 0;
            index2 = rr[i];
        }
        else if (i == nCrossover)
        {
            index1 = rr[i - 1] + 1;
            index2 = nElem - 1;
        }
        else
        {
            index1 = rr[i - 1] + 1;
            index2 = rr[i];
        }
}
for (int j = index1; j <= index2; j++)
{
    tempGenes[j] = pGene[p][j];
}

if (p == 0)
    p = 1;
else
    p = 0;
/* end of formation of a child */

inline void Mutation(int nElem, double mValue, Array1D < int >tempGene)
{
    for (int i = 0; i < nElem; i++)
    {
        double r = ((double) rand () / (RAND_MAX +1.0))/(rand ()%1000)/1000.0;
        //cout <<" rr ";
        if (r < mValue)
        {
            int m1, m2, m3;
            switch (tempGene[i])
            {
            case 0:
                m1 = 1;
                m2 = 2;
                m3 = 3;
                break;
            case 1:
                m1 = 0;
                m2 = 2;
                m3 = 3;
                break;
            case 2:
                m1 = 1;
                m2 = 0;
                m3 = 3;
                break;
            case 3:
                m1 = 1;
                m2 = 2;
            }
m3 = 0;
break;
}
int rr = (int)(3.0* rand () / (RAND_MAX +1.0));
//cout << rr << "\n";
if (rr ==0)
    tempGene[i] = m1;
else if (rr ==1)
    tempGene[i] = m2;
else
    tempGene[i] = m3;
}

/**
 * @defgroup Functions 
 * @brief Functions
 * @
 */

void QuickSort (Array1D < double >fitness, Array1D < int >address, int array_size)
{
    Q_sort (fitness, address, 0, array_size - 1);
}

void Q_sort (Array1D < double >fitness, Array1D < int >address, int left, int right)
{
    double pivot;
    int l_hold, r_hold, pivot_address;

    l_hold = left;
    r_hold = right;
    pivot = fitness[left];
    pivot_address = address[left];

    while (left < right)
    {
        while ((fitness[right] >= pivot) && (left < right))
            right--;
        if (left != right)
        {
            fitness[left] = fitness[right];
            address[left] = address[right];
            left++;
            l_hold = left;
            r_hold = right;
            pivot = fitness[left];
            pivot_address = address[left];
        }
    }
}

while ((fitness[left] <= pivot) && (left < right))
    left++;
if (left != right)
    {
        fitness[right] = fitness[left];
        address[right] = address[left];
        right--;
    }
fitness[left] = pivot;
address[left] = pivot_address;
int pivotPoint = left;
left = l_hold;
right = r_hold;
if (left < pivotPoint)
    Q_sort (fitness, address, left, pivotPoint - 1);
if (right > pivotPoint)
    Q_sort (fitness, address, pivotPoint + 1, right);
}

ChromosomeMean.C

#include "optVolt.h"
using namespace std;

Array1D<double> Chromosome::elLength(noOfElem,0.0);
Array1D<int>    Chromosome::xDoF(noOfNodes,0),
                Chromosome::yDoF(noOfNodes,0),
                Chromosome::zDoF(noOfNodes,0);  //variables for Dof.txt
Array1D<int>    Chromosome::node1(noOfElem,0),
                Chromosome::node2(noOfElem,0),
                Chromosome::geneType(noOfElem,0);  //variables for connect.txt
Array1D<double> Chromosome::xOrig(noOfNodes,0.0),
                Chromosome::yOrig(noOfNodes,0.0),
                Chromosome::zOrig(noOfNodes,0.0);  //variables for nodal_coord.txt
Array1D<double> Chromosome::le(noOfElem,0.0),
                Chromosome::me(noOfElem,0.0),
                Chromosome::ne(noOfElem,0.0);  //variables for direction cosines
Array1D<int>    Chromosome::cDoF(noOfcDoF,0);  // For variable voltage calculation need to specify or read
Array1D<double> Chromosome::dc(noOfcDoF,0.0);  //
Array1D<int>    Chromosome::appFDoF(nAppFDoF,0);  // For variable voltage calculation need to specify or read
Array1D<double> Chromosome::appF(nAppFDoF,0.0);  //

Chromosome::Chromosome()
{
feasibility=1; // A chromosome is feasible by default when defined if it violates the stress or strain constraints then it becomes infeasible
nActive=0; // Number of active cables.
nActiveSmall=0; // Number of active cables in 1/8 of the cell.

Array1D<int> genest(noOfElemG,0);
genes = genest.copy();
}

void Chromosome::SetGenes( Array1D<int> genest)
{
    genes=genest.copy();
    feasibility=1;
    Array2D<double> tempu(maxDoF,4,0.0),tempstr(noOfElem,4,0.0);
    u = tempu.copy();
    str = tempstr.copy();
}

int Chromosome::ReadFiles(int rank)
{
    int retVal=0;
    
    /* File Open -- Start */
    ifstream FileNodalCoord( "reflected3D_coords.txt" );
    ifstream FileConnect("reflected3D_connectivity.txt");
    ifstream FileDof("reflected3D_DoF.txt");                          //"Dof_test1.txt""DoF_3D.txt"
    ifstream FileAppFDof("reflected3D_appF.txt");                    //"appFDof_test1.txt""appFDof_3D.txt"
    ifstream FileCDof("reflected3D_output.txt");                    //"cDof_test1.txt""cDoF_3D.txt"
    assert(FileDof.is_open());
    assert(FileConnect.is_open());
    assert(FileNodalCoord.is_open());
    assert(FileCDof.is_open());
    assert(FileAppFDof.is_open());
    /* File Open -- End */
    
    /* Start :- Input from files in to variables for easy access */
    FileDof.seekg(0); //positioning the cursor at the beginning of the file
    for (int j=0; j<noOfNodes;j++)
    {
        FileNodalCoord >> temp >> xOrig[j] >> yOrig[j] >> zOrig[j];
        FileDof >> temp >> xDoF[j] >> yDoF[j] >> zDoF[j];
    }
void Chromosome::ElLengthCalc()
{
  double xdiff,ydiff,zdiff;
  for (int j=0;j<noOfElem;j++)
  {
    xdiff=(x[node2[j]-1]-x[node1[j]-1]);
    ydiff=(y[node2[j]-1]-y[node1[j]-1]);
    zdiff=(z[node2[j]-1]-z[node1[j]-1]);
    elLength[j]=(double) (sqrt(pow(xdiff,2)+pow(ydiff,2)+pow(zdiff,2)));
    le[j] = xdiff/elLength[j];
    me[j] = ydiff/elLength[j];
    ne[j] = zdiff/elLength[j];
  }
}

void Chromosome::GlobalMatrices( Array2D<double> K, const int &mode, double volume[])
{
  /* Declaration -- Start */

  int  11, 12;
  double A, E;
  double kel[6][6];

  /* End :- Input from files into variables for easy access */
  nLayers =node1.copy();
  nLayers=0;

  return retVal;
}
// double volume[3]={0,0,0}; // 0:- void 1:- strut 2:- cable
/* Declaration -- End */

/* Global Matrix Assembly -- Start */
for (int i=0;i<maxDoF;i++)
for (int j=0;j<maxDoF;j++)
K[i][j]=0.0;

nActive=0; nActiveSmall=0; // nActive is the total number of cables in the entire reflected truss and nActiveSmall is the total number of cables in the original cell
for (int j=0;j<noOfElem;j++)
{
    if (mode==activated)
    {
        if (genes[geneType[j]-1]==0)
        {
            A=aLow;
            E=eLow;
            nLayers[j]=0;
            volume[0] += aCab*elLength[j];
        }
        else if (genes[geneType[j]-1]==voids)
        {
            A=aLow;
            E=eLow;
            nLayers[j]=0;
            volume[2] += aCab*elLength[j];
        }
        else if (genes[geneType[j]-1]==1)
        {
            A=aStr;
            E=eStr;
            nLayers[j]=0;
            volume[1] += aStr*elLength[j];
        }
        else if (genes[geneType[j]-1]==active)
        {
            A=aLow;
            E=eCab;
            nActive++;
            if(j<noOfElemG) nActiveSmall++;
            nLayers[j]=(int) (elLength[j]/layerThickness);
            volume[2] += aCab*elLength[j];
        }
    }
    else if (mode== forced)
    {

volume[0]=1;
volume[1]=1;
volume[2]=1;
if (genes[geneType[j]-1]==0)
    {
        A=aLow;
        E=eLow;
    }
else if (genes[geneType[j]-1]==voids)
    {
        A=aCab;
        E=eCab;
        if (cableCheck[j]==0)
            A=aLow;
    }
else if (genes[geneType[j]-1]==1)
    {
        A=aStr;
        E=eStr;
    }
else if (genes[geneType[j]-1]==active)
    {
        A=aCab;
        E=eCab;
        if (cableCheck[j]==0)
            A=aLow;
    }
}

double mult=A*E/elLength[j];

/* el stiffness matrix */

kel[0][0]=mult*pow(le[j],2);
kel[0][1]=mult*le[j]*me[j];
kel[0][2]=mult*le[j]*ne[j];
kel[0][3]=-kel[0][0];
kel[0][4]=-kel[0][1];
kel[0][5]=-kel[0][2];
kil[1][0]=kel[0][1];
kil[1][1]=mult*pow(me[j],2);
kil[1][2]=mult*me[j]*ne[j];
kil[1][3]=-kil[1][0];
kil[1][4]=-kil[1][1];
kil[1][5]=-kil[1][2];
kil[2][0]=kel[0][2];
kil[2][1]=kil[1][2];
kil[2][2]=mult*pow(ne[j],2);
kil[2][3]=kil[2][0];
kil[2][4]=kil[2][1];
kil[2][5]=kil[2][2];
\begin{verbatim}
kel[2][4]=kel[2][1];
kel[2][5]=kel[2][2];
kel[3][0]=kel[0][3];
kel[3][1]=kel[1][3];
kel[3][2]=kel[2][3];
kel[3][3]=kel[3][0];
kel[3][4]=kel[3][1];
kel[3][5]=kel[3][2];
kel[4][0]=kel[0][4];
kel[4][1]=kel[1][4];
kel[4][2]=kel[2][4];
kel[4][3]=kel[4][0];
kel[4][4]=kel[4][1];
kel[4][5]=kel[4][2];
kel[5][0]=kel[0][5];
kel[5][1]=kel[1][5];
kel[5][2]=kel[2][5];
kel[5][3]=kel[5][0];
kel[5][4]=kel[5][1];
kel[5][5]=kel[5][2];

/* el stiffness matrix */

/* Insertion into global stiffness matrix */

int dof[6] = {xDoF[node1[j]-1],yDoF[node1[j]-1],zDoF[node1[j]-1],xDoF[node2[j]-1],yDoF[node2[j]-1],zDoF[node2[j]-1]};

for (int p=0;p<6;p++)
{
    l1=dof[p];
    if (l1!=0)
    for (int q=0;q<6;q++)
    {
        l2=dof[q];
        if (l2!=0)
            K[l1-1][l2-1]=K[l1-1][l2-1]+kel[p][q];
    }
}

/* Insertion into global stiffness matrix */

} // end of j==noOfEl

/* Global Matrix Assembly -- End */

double Chromosome::Fitness(int rank)
{

Array1D<double> V, force(maxDoF, 0.0), U(maxDoF, 0.0);
Array1D<double> stress(noOfElem, 0.0);
double fitness=0, fitnessDis=0, volumeMaterial[3]={0, 0, 0}, percentActive=0, percentTruss=0;

double A, E;
Array2D<double> ktemp(maxDoF, maxDoF, 0.0);
K = ktemp.copy();
Kinw = ktemp.copy();
int uCounter=0;

Array1D<double> prevU(noOfcDoF, 0.0), uOld(noOfcDoF, 0.0);
double uNew;

/* Feasibility Check -- Start */
/*-------Activated case------*/

double magnifier=0.005;
for (int j=0; j<noOfCables; j++) // using two sets of cables one is active while the other is void
{
    int mode = activated;
    if (j==0)
    {
        active=2;
        voids=3;
    }
    else
    {
        active=3;
        voids=2;
    }
    for(int phase=0; phase<2; phase++) // checking phase=0 and calculate phase = 1;
    {
        if (phase==0)
        {
            lower=0.3;
            upper=3.1;
            x = xOrig.copy(); y = yOrig.copy(); z = zOrig.copy();
        }
        else
        {
            lower=2.9;
            // upper = 6;
            int jj=0;
            for (int ii=0; ii<noOfNodes; ii++)
            {
                if (xDoF[ii]!=0)
{ x[\text{ii}] = x[\text{ii}]+U[\text{jj}]; \text{jj}++; }
if (yDoF[\text{ii}]!=0)
{ y[\text{ii}] = y[\text{ii}]+U[\text{jj}]; \text{jj}++; }
if (zDoF[\text{ii}]!=0)
{ z[\text{ii}] = z[\text{ii}]+U[\text{jj}]; \text{jj}++; }
}

ElLengthCalc();
GlobalMatrices(K,mode,volumeMaterial);


\text{if((percentActive >0.63)|| (percentTruss>0.60))}
\text{// if((percentActive >0.93)|| (percentTruss>0.90))}
\{  
\text{feasibility = 0;}
\text{//cout << __LINE__ << "\text{percentActive } " << percentActive} 
\text{<< "\text{percentTruss } " << percentTruss} 
\text{<< endl;}
\text{// cout << "returning percentactive>40\%\n";}
\text{return fitness;}
\}

\text{if ((nActive < 2)||(percentTruss < 3e-2))}
\{  
\text{//cout << __LINE__ << "nActive or percentTruss is 0. Returning 0 as feasibility\n";}
\text{feasibility = 0;}
\text{return fitness;}
\}

F= Fvector();
Kinv = Inverse(K);
if (V[i]<0.3) V[i]=0.3;
if (V[i]>3.1) V[i]=3.1;
V[i]=magnifier*V[i];
    cout << V[i]<<' ';
}

force = F*V;
U = Kinv*force;

for(int i =0;i<maxDoF;i++)
    u[i][uCounter] = U[i]; // Storing the disp for plotting in a 2d array
    uCounter++;

/* fitness calculation*/
double meanDisp=0.0, varDisp=0.0;
if (j==0)
{
    for(int i = 0;i<noOfcDoF;i++)
    {
        meanDisp = meanDisp + U[cDoF[i]-1];
        prevU[i] = prevU[i]+ U[cDoF[i]-1];
    }
    meanDisp = meanDisp/noOfcDoF;
    // prevU[0] = meanDisp;

    for(int i = 0;i<noOfcDoF;i++)
        varDisp = varDisp + (U[cDoF[i]-1]/meanDisp-1)*(U[cDoF[i]-1]/meanDisp-1)/noOfcDoF;
    varDisp = sqrt(varDisp);

    fitness = fitness + meanDisp/(0.01+varDisp); //initialize fitness to 0 above
}
else
{
    for(int i = 0;i<noOfcDoF;i++)
    {
meanDisp = meanDisp + U[cDoF[i]-1];
meanDisp = meanDisp/noOfcDoF;

for(int i = 0; i < noOfcDoF; i++)
    varDisp = varDisp + (U[cDoF[i]-1]/meanDisp-1)*(U[cDoF[i]-1]/meanDisp-1)/noOfcDoF;
    varDisp = sqrt(varDisp);

fitness = fitness - meanDisp/(0.01+varDisp); //initialize fitness to 0 above

/* Checking if both positive and negative disps are nearly equal and opposite */

if (phase==0)
    for(int i = 0; i < noOfcDoF; i++)
        uOld[i]=U[cDoF[i]-1];
else
    for(int i = 0; i < noOfcDoF; i++)
    {
        uNew = uOld[i]+U[cDoF[i]-1];
        if( (fabs(prevU[i]/uNew)>10) || (fabs(uNew/prevU[i])>10))
        {
            feasibility=0; fitness=0; // not necessary to make fitness=0
            cout << fabs(prevU[i]/uNew)<< "unequal displacements \n";
            return fitness;
        }
        if( (prevU[i]/uNew)>0 )
        {
            feasibility=0; fitness=0; // not necessary to make fitness=0
            cout << " same side displacements \n";
            return fitness;
        }
    }
    //    if( (fabs(prevU[0]/meanDisp)>3) || (fabs(meanDisp/prevU[0])>3) ||
    ((prevU[0]/meanDisp)>0) )
    //    {
    //        feasibility=0; fitness=0; // not necessary to make fitness=0
    //        cout << "unequal disps"<<endl;
    //        return fitness;
    //    }
    //cout << __LINE__ <<'t' << fitness << endl;

} // End of for loop through two cable types
} // End of loop for different phases
unlink("/tmp/myscheme");

/*--------------------------------------------------------------------------*/
// Array1D<int> genest(noOfElem,0);
// genest = genes.copy();

x = xOrig.copy(); y = yOrig.copy(); z = zOrig.copy();
ElLengthCalc();

for (int j = 0;j<2;j++) // for both positive and negative directions of forcing.
{
    int mode = forced;

    for (int i = 0; i<nAppFDoF; i++) // for (int forceCase = 0; forceCase<2; forceCase++)
    {
        Array1D<int> prevCableCheck(noOfElem,1);
        cableCheck = prevCableCheck.copy();

        Array1D<double> force(maxDoF,0.0),U(maxDoF,0.0);
        // int forceStart, forceEnd;

        // if (forceCase == 0)
        //   { forceStart = 0; forceEnd = 3;}
        // else
        //   { forceStart = 3; forceEnd = nAppFDoF;}

        force[appFDoF[i]-1] = pow(-1.0,j)*appF[i];

        int iterationCounter=0;
        do
        {
            flag1=0;
            GlobalMatrices(K,mode,volumeMaterial);
            QR<double> slvU(K);
            // Cholesky<double> slvU(K);

            // if (slvU.is_spd())
            if (slvU.isFullRank())
                U = slvU.solve(force);
            else
                cout << "factorization was not complete.\n";
        } while (flag1==0); // while flag1==0;
    }
}
// Stress calculation for checking cables in compression ............// If in doubt refer to page 104 FEM by Chandrupatla and Belegundu
for(int i =0;i<noOfElem;i++)
{
    if (genes[geneType[i]-1]==0)
    { A=aLow;  
      E=eLow;
    }
    else if (genes[geneType[i]-1]==1)
    { A=aStr;  
      E=eStr;
    }
    else if (genes[geneType[i]-1]==active || genes[geneType[i]-1]==voids)
    { A=aCab;  
      E=eCab;
      if (cableCheck[i]==0)
        A=aLow;
    }

    double directionCosine[6] = {-le[i],-me[i],-ne[i],le[i],me[i],ne[i]};
    int dof[6] = {xDoF[node1[i]-1],yDoF[node1[i]-1],zDoF[node1[i]-1],xDoF[node2[i]-1],yDoF[node2[i]-1],zDoF[node2[i]-1]};
    double uu;

    stress[i]=0;
    for(int p = 0;p<6;p++)
    {
        if(dof[p]==0)
            uu=0;
        else {
            assert((dof[p] - 1) >= 0 && (dof[p] - 1) < maxDoF);
            uu = U[dof[p]-1];
        }
        stress[i] = stress[i] + A*E/elLength[i]*uu*directionCosine[p];
    }

    prevCableCh eck[i]=cableCheck[i];
cableCheck[i]=1;

    if ((genes[geneType[i]-1]==active || genes[geneType[i]-1]==voids) &&
        stress[i]<0)
        cableCheck[i]=0;

    if(prevCableCheck[i]!=cableCheck[i])
        flag1=1;

    // Checking for the strength of truss members
    // if(genes[geneType[i]-1]==1 && fabs(stress[i])>stressLim)
    //   feasibility =0; // if infeasible make it 0
// End of strength check

} // End of Stress Calculation

iterationCounter++;
} while (flag1==1 && iterationCounter<50); // Checking the voids in cables
//cout << iterationCounter<<endl;
// genes = genest.copy();

if (j==0)
fitnessDis = fitnessDis + U[appFDoF[i]-1]; //initialize fitness to 0 above
else
fitnessDis = fitnessDis - U[appFDoF[i]-1]; //initialize fitness to 0 above

} // End of force cases

} // End of two force signs
// cout<< "fitness due to actuation " << fitness<<endl;
fitness =fabs(fitness/fitnessDis)/(pow((percentActive+percentTruss),1.5)*pow(percentActive,3));
return fitness;

} // End of Fitness

Array2D<double> Chromosome::Fvector()
{

Array2D<double> FF(maxDoF,nActiveSmall,0.0);

/*.....Calculating the Force matrix due which mu ltiplies the voltage vector to give the force vector.... */
int m=0;
Array1D<int> posV(noOfElemG,0);
long double f;
for (int j=0;j<noOfElem;j++)
{

if (genes[geneType[j]-1]==active)
{
    if(j<noOfElemG)
    {
        posV[j] = m;
        m++; // increment number for the column number goes to the active cables
    }
    f=(aCab*eCab/elLength[j])*d33*nLayers[j];
double localF[6] = {f*le[j],f*me[j],f*ne[j],-f*le[j],-f*me[j],-f*ne[j]};

int dof[6] = {xDoF[node1[j]-1],yDoF[node1[j]-1],zDoF[node1[j]-1],xDoF[node2[j]-1],yDoF[node2[j]-1],zDoF[node2[j]-1]};
for (int p=0;p<6;p++)
{
    if(dof[p]!=0) FF[dof[p]-1][posV[geneType[j]-1]] = FF[dof[p]-1][posV[geneType[j]-1]]+localF[p];
}

/*..End of calculation of Force Matrix */

Array2D<double> Chromosome::Inverse(const Array2D<double> K)
{
    Array2D<double> X(K.dim1(),K.dim1(),0.0);
    if(K.dim1() != K.dim2())
        cout << "******Not a square matrix for inverse calculation******\n";
    else
    {
        QR<double> inv(K);
        assert(inv.isFullRank());
        Array2D<double> II(K.dim1(),K.dim1(),0.0);
        for (int i=0;i<K.dim1();i++)
            II[i][i]=1;
        X = inv.solve(II);
    }
    return X;
}

Array2D<double> Chromosome::Transpose(const Array2D<double> K)
{
    Array2D<double> X(K.dim2(),K.dim1(),0.0);
    for(int i = 0;i<X.dim1();i++)
    for(int j = 0;j<X.dim2();j++)
        X[i][j]=K[j][i];
    return X;
}
Optvolt.h

#include "chromosome.h"
#include "NLF.h"
#include <unistd.h>
#include "BoundConstraint.h"
#include "CompoundConstraint.h"
#include "OptppArray.h"
// #include "OptBaQNewton.h"
#include "OptBCQNewton.h"
#include <cassert.h>

void init_hs65(int ndim, ColumnVector& v);
void hs65(int mode, int ndim, const ColumnVector& v, double& fx, ColumnVector& gv, int& result);

int maxDoFOptV = Chromosome::maxDoF;
Array2D<double> KinvOptV, FOptV, dxidvOptV;
int nOutputPoints = Chromosome::noOfcDoF;
Array1D<int> cDoFOptV;
extern bool first_time1;

Array1D<double> Chromosome::VoltOpt()
{
    double fitness=0, fitnessDis=0;
    int ndim = nActiveSmall;
    KinvOptV = Kinv;
    FOptV = F;
    dxidvOptV = KinvOptV*FOptV;
    cDoFOptV = cDoF;
    assert( nActiveSmall > 1 );
    Array1D<double> volt(ndim,0.0);
    ColumnVector lowerOptV(ndim,0.0);
    ColumnVector upperOptV(ndim);
    lowerOptV = lower;
    upperOptV = upper;
    Constraint c1 = new BoundConstraint(ndim, lowerOptV, upperOptV);
    CompoundConstraint* constraints= new CompoundConstraint(c1);
    NLF1 nips(ndim, hs65, init_hs65, constraints);
// nips.initFcn();
//nips.eval();
// OptBaQNewton objfcn(&nips);
OptBCQNewton objfcn(&nips);

/* The "0" in the second argument says to create a new file. A "1"
   would signify appending to an existing file. */

// objfcn.setOutputFile("voltage149.out", 1);
first_time1=true;
objfcn.setFcnTol(1.0e-06);
objfcn.setMaxIter(10);
objfcn.setMaxFeval(5000);
objfcn setSearchStrategy(TrustRegion); //LineSearch); //TrustPDS);

objfcn.optimize();
//objfcn.printStatus("Solution from Bound Constrained Newton");

double fval = nips.getF();
ColumnVector xVal;
xVal = nips.getXc();
for(int i=0;i<ndim;i++)
  volt[i]=xVal(i+1);

objfcn.cleanup();
//unlink("/tmp/myscheme");
delete constraints;
return volt;
}

#include "NLP.h"

void init_hs65(int ndim, ColumnVector& v)
{
  double factor = 0.0;

  // ColumnVectors are indexed from 1, and they use parentheses around
  // the index.
  for(int i=1;i<= ndim;i++)
    v(i)= 3;//
  // for(int i=3;i<= ndim;i++)
  //    v(i)= 2.5;
}

void hs65(int mode, int ndim, const ColumnVector& v, double& fx, ColumnVector& gv, int& result)
Array1D<double> forceLocal(maxDoFOptV,0.0), Ulocal(maxDoFOptV,0.0), Vlocal(ndim,0.0);

for(int i=0; i<ndim;i++)
    Vlocal[i]=v(i+1);
forceLocal = FOptV*Vlocal;
Ulocal = KinvOptV*forceLocal;

/* fitness calculation*/
double meanDisp=0.0, varDisp=0.0, varDispSq=0.0;

for(int i = 0;i<noutputPoints;i++)
{
    meanDisp = meanDisp + Ulocal[cDoFOptV[i]-1];
}
meanDisp = meanDisp/noutputPoints;

for(int i = 0;i<noutputPoints;i++)
    varDispSq = varDispSq + (Ulocal[cDoFOptV[i]-1]/meanDisp-1)*(Ulocal[cDoFOptV[i]-1]/meanDisp-1)/noutputPoints;
varDisp = sqrt(varDispSq);

double minimizingMultipler=1.0;
if(meanDisp>0) minimizingMultipler = -1.0;

if (mode & NLPFunction) {
    fx  = minimizingMultipler*meanDisp/(0.1+varDisp);
    result = NLPFunction;
}

if (mode & NLPGradient) {

double dnumerator, ddenominator;
double numerator, denominator, dfdx, dfdv;
numerator = meanDisp;
denominator = 0.1+varDisp;
meanDisp=meanDisp - 1e-15*minimizingMultipler;
varDisp = varDisp + 1e-15*fabs(meanDisp);

for(int j=0;j<ndim;j++)
{
    dfdv=0;
    for (int i=0;i<noutputPoints;i++)
    {
        dnumerator = 1.0/noutputPoints;
        ddenominator = 1.0/(noutputPoints*varDisp)* ((Ulocal[cDoFOptV[i]-1]-
meanDisp)/(meanDisp*meanDisp)- varDispSq/meanDisp)
;
\[ \text{dfdxi} = \frac{(\text{dnumerator} \times \text{denominator}) - \text{numerator} \times \text{ddenominator})}{(\text{denominator} \times \text{denominator})}; \]

\[ \text{dfdv} = \text{dfdv} + \text{dfdxi} \times \text{dxidvOptV[cDoFOptV[i]-1][j]}; \]

\[ \text{gv} = \minimizingMultiplier \times \text{dfdv}; \]

\[ \text{result} = \text{NLPGradient}; \]

\[ \]

**Chromosome.h** (header file of the parameter values)

```cpp
#include <fstream.h>
#include <cstdlib>
#include <ctime.h>
#include <cassert>
#include <math.h>
#include <iostream.h>
#include "jama_qr.h"
#include "jama_cholesky.h"
#include "tnt.h"
using namespace TNT;
using namespace JAMA;

class Chromosome
{
    public:
        Chromosome();
        void SetGenes(Array1D<int> genest);
        void GetGenes();
        double Fitness(int rank);
        int ReadFiles(int rank);
        void ELLengthCalc();

        Array1D<int> genes;
        Array2D<double> u,str; // variable to store the displacements
        int feasibility; //infeasible=0; feasible=1;
        double lower,upper;

        static const int noOfElem  =  158; // 20 72 158 1036
        static const int noOfElemG = 28; //28 6  20 28 158
        static const int noOfNodes = 27;//27 125
        static const int typesOfGenes  = 4;
        static const int maxDoF    = 72;//72 69 366
        static const int noOfcDoF  = 5;
        static const int nAppFDoF  = 12;// 150 2 2; 2 0 2; 2 2 2;
};
```
private:
static const double pi=3.14159;
static const int noOfCables=2;         // no of sets of cables ( 1 or 2)
static const double layerThickness=67e-6; // piezo stack variable
static const double eStr=193e9;         // Young's modulus of Brush Wellman AM 162
AlBeMet® Extruded Bar,
static const double eCab=72.236e9;       // Young's modulus of Stainless Steel 306
static const double eLow=72.236e9;       // Young's modulus of Voids
static const double aStr=1e-4;           //3.14159*0.5*0.5;
static const double aCab=1e-4;           //3.14159*0.5*0.5;
static const double aLow=1e-10;          // area of voids
static const double stressLim=2.9635e9;  // Ultimate tensile stress for cable

int active, voids, flag1;  //flag1 is a checking variable to check for cables under compression
int nActive, nActiveSmall;
double temp;
static const double d33=290e-12;
static const int forced=2;
static const int activated=1;

static Array1D<double> elLength;
static Array1D<int> xDoF,yDoF,zDoF;     //variables for Dof.txt
static Array1D<int> node1,node2,geneType; //variables for connect.text
static Array1D<double> xOrig,yOrig,zOrig;  //variables for nodal_coord.txt
static Array1D<double> le,me,ne;          //variables for direction cosines
static Array1D<int> eDoF, appFDoF;        //constrained dof, applided force dof
static Array1D<double> dc,appF;           //constrained displacement, applied force

Array1D<int> nLayers;
Array1D<int> cableCheck;
Array1D<double> x,y,z;
Array2D<double> K,Kinv,F;

void GlobalMatrices( Array2D<double> K, const int &mode, double volume[]);
Array2D<double> Inverse(const Array2D<double> K);
Array2D<double> Transpose(const Array2D<double> K);
Array1D<double> VoltOpt();
Array2D<double> Fvector();

};
Appendix C

Local Stiffness Matrix for Beam with a Pin Joint

The local stiffness matrix of a beam element with a pin joint on the right hand side is developed here. The degrees of freedom corresponding to this element are shown in Figure 3-1.

![Figure 3-1: Local degrees of freedom of a beam element](image)

The beam is assumed to be pinned on the right hand side so any change in the deflection $u_4$, does not affect the strain energy stored in the beam. The three shape functions $H_1$, $H_2$, and $H_3$ corresponding to the deflections $u_1$, $u_2$ and $u_3$ respectively are shown in Eq. 3.1

\[
H_1 = 1 - \frac{\xi^2}{2}(3 - \xi) \\
H_2 = \frac{l_e}{2} \xi \left(2 - 2\xi + \xi^2\right) \quad 0 < \xi < 1 \\
H_3 = \frac{\xi^2}{2}(3 - \xi)
\]

$l_e$ is the length of the element. $H_1$ and $H_3$ were derived from the deflection of a fixed free beam, fixed on the left hand side. $H_1$ has a unit deflection on the left hand side ($\xi = 0$) and $H_3$ has a unit deflection on the right hand side ($\xi = 1$). $H_2$ was derived from the deflected shape of a pinned-pinned beam with a moment applied on the left hand side. The slope ($dH_2/d\xi = 1$) of $H_2$ is 1 on the left hand side ($\xi = 0$). The deflection $\nu$ of the beam is written in terms of the end displacements $u_1$, $u_2$, and $u_3$ as Eq. 3.2.
where $\mathbf{u}$ is the nodal displacement vector $[u_1, u_2, u_3]$ and $H$ is the shape function vector $[H_1, H_2, H_3]$. The strain energy of the element in bending is calculated from Eq. 3.3

$$U_e = \frac{1}{2} EI \int_e \frac{d^2 \nu}{dx^2} dx$$

The chain rule relates the derivative w.r.t. $x$ to the derivative w.r.t. $\xi$ as

$$\frac{d\nu}{dx} \frac{d\xi}{dx} = \frac{1}{l_e} \frac{d\nu}{d\xi}$$

Using equations 3.4 and 3.2 in Eq. 3.3 gives Eq. 3.5

$$U_e = \mathbf{u}^T \frac{3EI}{2l_e^3} \begin{bmatrix} 1 & l_e & -1 \\ l_e & l_e^2 & -l_e \\ -1 & -l_e & 1 \end{bmatrix} \mathbf{u} = \frac{1}{2} \mathbf{u}^T \mathbf{k}' \mathbf{u}$$

Hence the local stiffness matrix $\mathbf{k}'$ is written as Eq. 3.6

$$\mathbf{k}' = \frac{3EI}{l_e^3} \begin{bmatrix} 1 & l_e & -1 \\ l_e & l_e^2 & -l_e \\ -1 & -l_e & 1 \end{bmatrix}$$

The rotational degree of freedom on the right hand side $u_4$ does not contribute to the strain energy. The stiffness matrix can be modified to include this degree of freedom by including a small value $\delta$ as shown in Eq. 3.7.

$$\mathbf{k} = \frac{3EI}{l_e^3} \begin{bmatrix} 1 & l_e & -1 & \delta \\ l_e & l_e^2 & -l_e & \delta \\ -1 & -l_e & 1 & \delta \\ \delta & \delta & \delta & \delta \end{bmatrix}$$

This is the local stiffness matrix of a beam element with a pin joint on the right hand side.
VITA

Deepak Shyamlal Ramrakhyani

EDUCATION

PhD in Aerospace Engineering, August 05
The Pennsylvania State University (University Park)
Thesis title: “Morphing Aircraft Structures using Tendon Actuated Compliant Cellular Trusses”

MS in Aerospace Engineering, May 02
The Pennsylvania State University (University Park)

BS in Aerospace Engineering, May 99
Indian Institute of Technology, Madras, India.

TECHNICAL PUBLICATIONS

Journal Publications:

Conference Proceedings: