MULTIDIMENSIONAL DESIGN VISUALIZATION IN MULTIOBJECTIVE OPTIMIZATION

John Eddy
Graduate Research Assistant, Student Member AIAA
Department of Mechanical and Aerospace Engineering
University at Buffalo, SUNY
Buffalo, NY USA

Kemper Lewis
Associate Professor, Member AIAA
Department of Mechanical and Aerospace Engineering
University at Buffalo, SUNY
Buffalo, NY USA

Abstract

As our ability to generate more and more data for increasingly large engineering models improves, the need for means of managing that data becomes greater. Information management from a decision-making perspective involves being able to capture and then represent significant information to a designer so that he or she can make effective and efficient decisions. However, most visualization techniques used in engineering, such as graphs and charts, are limited to two-dimensional representations and at most three-dimensional representations. In this paper, we present a technique used to capture and represent engineering information in a multidimensional context. In this paper, we present an overview of the technique and provide details regarding two specialized functions for use in multidimensional and multiobjective optimization. This method is part of an effort to develop a comprehensive visualization-based optimization tool for multi-objective and robust design problems.

Background

Engineers are increasingly able to generate more and more data for large system models, and as a result, the need for methods for managing and visualizing that data becomes greater. From a design perspective, large-scale analysis can result in a huge number of potential design configurations existing in complicated multi-dimensional spaces. This information could be presented to a designer as page after page of printed data. However, it is very difficult to make sense of large data structures in print form.

Some of the first attempts at using visualization methods to aid decisions in design and optimization are found in [1]. More recent advances in computer visualization and Virtual Reality (VR) [2-4] are allowing designers and scientists to interact and manipulate vast amounts of data. Until these innovations, computers were solely relied on to interpret results and compute answers based on programs written. It is now possible to interact with these large datasets even while they are being used in running analyses [5-11]. Users have the ability to compress large amounts of data into a visual format, to investigate trends and relationships that could not be seen otherwise, and then make informed decisions regarding a product or process design.

Commercial companies such as Raytheon [12] and Boeing [3], among others, have attempted to improve their own design processes by taking advantage of Virtual Reality and Scientific Visualization. Virtual Reality and Scientific Visualization offer methods and concepts to produce graphical representations (pictures, graphs, etc.) of complex data. The enormous computing power readily available today has made these technologies more and more useful in recent years. Tools and techniques based on these technologies are being developed that can further improve the efficiency and accuracy of the solutions to these complex design problems. However, there is still a need for process improvement and multidimensional data representation, as well as for better incorporation of heuristics and design knowledge. One approach is through the implementation of Computational Steering concepts. Computational Steering [7-10] is the implementation of Virtual Reality and Scientific Visualization into a process or analysis, so as to give a researcher or practitioner the ability to view how a solution

procedure is progressing. The researcher has the ability to alter parameters while the analysis is running to interactively "steer" it to a solution. This ability to interact visually with design and optimization processes has the potential to benefit decision making greatly, as estimations indicate that approximately 70% of a human's attention is dedicated to visual input [13]. To this end, visualization can and should be considered a solution tool rather than simply a way to present results [14].

Because of the significant dependence that engineers have on visual cues and information representation, it seems sensible to provide visually enhanced design steering and optimization methodologies. Easily interpreted visual cues such as color, shape, relative size, etc. can be used effectively to convey trends or large amounts of perhaps imprecise information quickly. They can also be used to add dimension to data beyond the three spatial dimensions. In this paper, some of these cues are integrated into a visualization framework, which acts as the designer interface while single or multiobjective optimization techniques operate at a lower, processing level. This framework not only provides insight into the nature of the problem and the optimization algorithm, but also helps prevent wasted analysis and supplies the designer with an opportunity to interact with the analysis.

Real-time visualization methods can be classified into two general categories: Artifact-Based Visualization (ABV) schemes and Non-Artifact-Based Visualization (Non-ABV) schemes [11]. Artifact-based schemes are those that are typically imposed on a physical object with a prescribed geometry, such as representing the stress in a beam by color contours imposed on the beams geometry. Non-Artifact-based schemes are those that are not constrained by a physical geometry such as visualization of process optimizations.

Non-ABV can also be further divided into two subcategories [11]. The first includes those methods that are based on a topographical view of the design space whereby a fit is made to the performance objectives based on the results of many a priori system evaluations. The results are usually viewed as contour plots. As pointed out in [11], approaches of this type require that system evaluations be performed before the optimization begins in order to create the contours, they do not accommodate multi-dimensional data very well, and the representation may not be meaningful due to inadequacy of the system model and/or approximations used to generate the contour plots.

The second subcategory contains non-ABV methods that do not require a topographical representation of the problem, such as tabular data, 2D plots, parallel coordinate based, and physical programming based methods. The method presented here, Cloud Visualization (CVis), is a category 2b visualization method meaning that it is a Non-ABV method that does not begin with a topographical view of the design space. Readers interested in more detailed discussion of the advantages and disadvantages of ABV and non-ABV are directed to [11] for more information.

The next section begins a description of the features of CVis most relevant to this paper.

Cloud Visualization Description

Cloud Visualization is a means by which a designer can view all previously generated design information in both the design and the performance spaces simultaneously. Its implementation resembles that of the Visiview method developed by Bangay [15] with some significant differences. Design spaces are defined by the design variables of the problem while performance spaces are defined by the performance objectives. All spaces are displayed in separate windows that can be linked (as described later). The following sections describe some of the features of CVis.

General Information

Design information is presented as a cloud of design points plotted along three or less axes that represent either design variables or objective functions. The color of the cloud may vary throughout its volume to reflect the properties of each area. The data represented may be single or multi-objective but this paper will focus only on multi-objective problems. If design variables are displayed, then the design space is bounded by a transparent box, indicating side constraints of the problem. If objective functions are displayed, then the performance space will not include a bounding box. Regardless of which space is plotted, CVis starts with blank three dimensional spaces until the user specifies the number of variables or objectives in the problem.

Figure 1 shows a screen capture of the CVis environment with some arbitrary data being displayed. The design space is on the left (including the side constraint box). The two dimensional performance space is on the right (Z axis is not used).

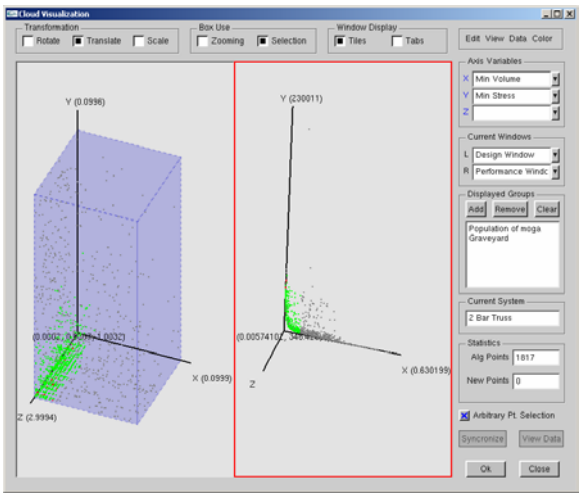

Figure 1 – The CVis Environment

Cloud Visualization is a module that can be used by any of the optimizers in our in-house optimization toolbox. It can be used to monitor an optimization continuously, periodically, or upon completion. The above data shows design information developed during the run of a multi-objective genetic algorithm (MOGA).

Coding

CVis is coded entirely using OpenGL and ANSI C++. It is therefore portable to a number of platforms. All of the runs in this paper are performed on a PC with a 1.3 GHz Pentium IV processor.

Interaction

The spaces can be freely translated, rotated, and scaled to aid in exploration. The window colors (axes, background, bounding box) can be freely changed to improve clarity. For a close up look of a particular region of a space, the user can use the zoom box to zoom in on a windowed. Further discussion of the interactive capabilities is presented in the following sections.

Point Selection

To further aid in discovering useful information about the problem, points may be selected in the spaces (groups of or individual points). The selected points will then be displayed by themselves. More information about point selection will be presented in later sections.

Use of Color

The use of color in visualization techniques can be very effective, but it can also be overwhelming, decreasing the effectiveness of the presentation and the value of the information [16]. Too many colors can only confuse users and the simple use of gray scales can often be more effective than elaborate color schemes [17].

To this end, the color scheme used in this paper shows all points colored according to a blocked color scheme whereby all feasible, non-optimal points are green, infeasible points are gray, and all optimal points are red. Since this paper focuses on multiobjective optimization problems, the term "optimal" means optimal in a Pareto optimal sense.

Space Linking

The optimization toolkit may be managing many optimization algorithms simultaneously each of which may be working on the same or different systems. Because of this, there may be many potential groups of points to visualize at any given time. Each CVis environment can only display one system at a time but may display the data from many algorithms simultaneously. To account for this, when a space is created within the environment (two of which are visible in Figure 1) it is independent of all other spaces. As alluded to previously, spaces present in CVis may be linked together. For example, the two windows shown in Figure 1 are linked and as a result, they display the same groups of points from the same algorithms. Many spaces of either type may be linked simultaneously.

The different possible links are useful for different reasons. The user can choose to link windows of the same type together. This is useful if there are more than 3 dimensions (defined by the design variables or objectives). The user can then see how groupings in some dimensions are dispersed in others.

The user may also choose to link windows of different types. This is useful for example to see how groups of points in the design space map into the performance space and visa versa.

Figure 2 shows the same spaces as Figure 1. A group of points has been selected in the design space and the selected points are then displayed alone in each window.

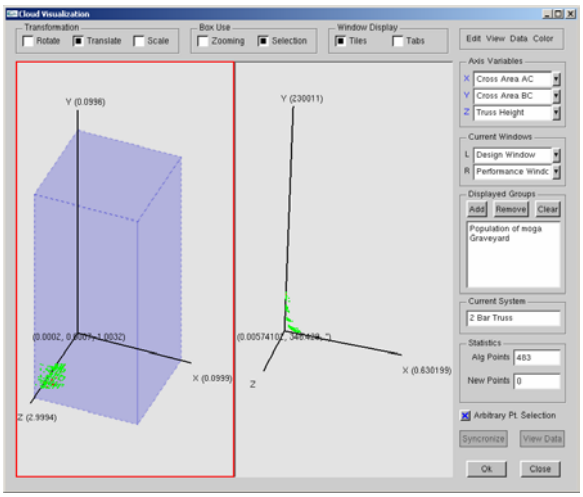

Figure 2 – Point Selection in Linked Windows

Clipping Planes

When large amounts of data appear on the screen, regions of the cloud can become very thick and it can be difficult to determine what is happening in those regions. For this reason, CVis allows the user to invoke cutting planes to section the cloud as shown in Figure 3. There are 6 planes so that the cloud can be clipped from either direction on each axis. Figure 3 shows the design space from Figure 1 (left) after having been transformed, zoomed, and clipped.

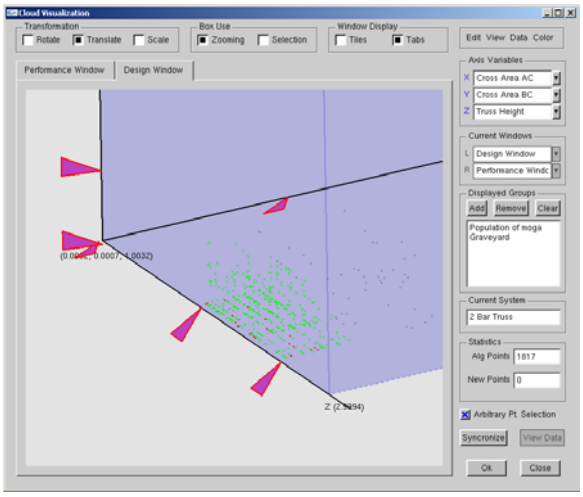

Figure 3 – Clipped Design Space of Figure 1

Each pink triangle indicates the location of a clip plane. The planes are moved by selecting the desired triangle and dragging it with the mouse.

Hidden Dimensions

While viewing three dimensions at a time in a multidimensional problem, it is possible to select a singular point that actually represents multiple unique design configurations. Consider the following two design points in a five design variable problem.

	X_1	X_2	X_3	X_4	X_5
P_1	3.59	6.47	1.25	8.79	9.12
P_2	3.59	6.47	1.25	4.23	7.77

If plotting X_1, X_2, and X_3 along the three axes of a design space, the two points above will both reside in the exact same location in the space. However, they are clearly two distinctly different designs.

In the case of selection of such a point, another window is created with three or less new design variables or objective functions (depending on how many remain unused). The newly created window is the child of the original window (parent). The points in the parent window are then locked and cannot be changed unless the child window is destroyed. The variables that can be used as axis variables in the child window are then restricted by its heritage. For instance, if a child's parent window has no parent window, then only the three variables of the child's parent are unusable. If it has grandparents, etc., then all previously used variables are held constant. The windows can be cascaded out until all the variables or objectives are represented on a plot as (conceptually) demonstrated in Figure 4.

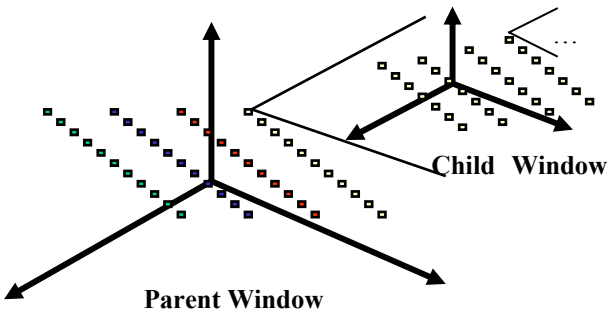

Figure 4: Expansion of Hidden Dimensions

Real Time Visualization and Design Steering

Many authors [7-11] indicate that it is desirable to visualize the progression of and interact with an optimization tool as it is running. For example, Physical Programming by Messac [11] provides the user with real time visualization of the progression of an optimization problem. The user is able to discover trends and relationships that may not be evident from a report of only the final results.

Computational Steering is the implementation of Virtual Reality and Scientific Visualization into a process or analysis, so as to give a researcher the ability to view how a solution procedure is progressing. The researcher has the ability to alter parameters while the analysis is running to interactively "steer" it to a solution. Winer and Bloebaum [9-10] present a Virtual Design Steering (VDS) method which employs a visualization tool called Graph Morphing. The user is able to visualize a design space three dimensions at a time and alter the remaining parameters via slider bars. After visualization, the resulting design configuration may be fed into an analysis or optimization routine.

The algorithm manager of the optimization toolbox provides intimate control over the algorithms that are running. It provides the user the ability to pause, cancel, or change the configuration of an algorithm at any time. CVis can use these abilities to implement various design steering actions.

The current implementation of Cloud Visualization supports some relatively simple steering operations although it has a great deal of potential for implementation of further design steering operations. The next section will present the most significant design steering operation supported by CVis.

Arbitrary Point Selection

Just as a user may wish to see how existing points map between different spaces, they may also wish to see how points that do not exist might perform. This choice would most likely be based on the user having identified a suspected trend or relationship in the data that leads him/her to the conclusion that a certain point in the space would perform well. However, selection of an arbitrary point in 3D space from a 2D display (computer monitor) is not trivial.

Consider the case of a problem for which there are five design variables. The user wishes to specify their values by mouse click selection in CVis spaces. This task is carried out as follows:

Consider the two design spaces in Figure 5 which together display all the design variables for a sample five design variable optimization problem (no data is shown for clarity). The space on the left contains three variables and the space on the right has two. Three axis are shown in both spaces as the initial default setting in CVis. The left space is the current target space as indicated by the red outline.

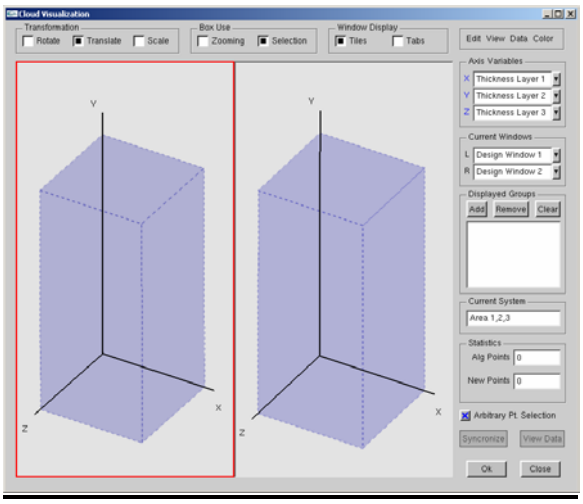

Figure 5 – Arbitrary Point Selection

The process for a space containing three variables begins with the selection of a point. The point is shown in Figure 6 below as a black X. The mouse click occurred at the vertex of the X.

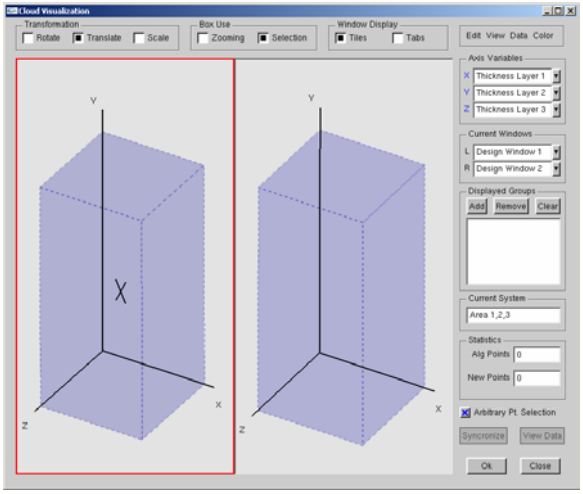

Figure 6 – Start of Arbitrary Point Selection

Having missed any displayed points (which there are none here), and with Arbitrary Point Selection enabled, CVis interprets this selection as an attempt at arbitrary point selection. The black X in the left space is actually the endpoint of a line segment that is exactly parallel to the view direction and terminates on whichever two planes of the bounding box it crosses. Rotating the space as in Figure 7 below reveals the line created (the bounding box color has been changed to reveal the line more clearly).

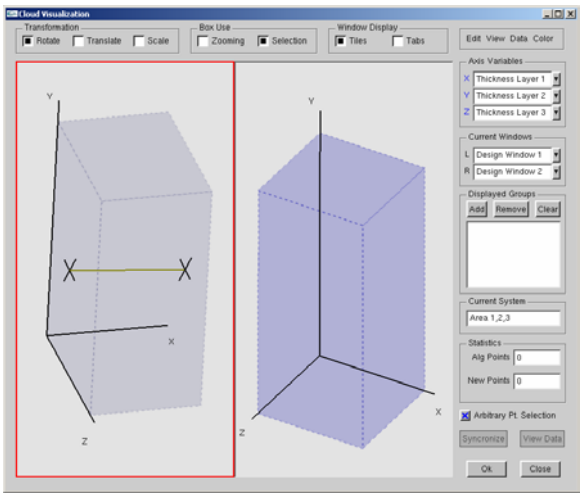

Figure 7 – Line Created by First Mouse Click

At this point, there is insufficient information to indicate the three desired coordinates. It is now necessary to select a point along the line to finish the selection of the first 3 dimensions. A mouse click that misses the line is interpreted as a cue to cancel the point selection operation. A strike on the line will bring up a window listing all the variables for the system and fill in the selected values for these first three variables (which in this case are not the first three in order as you can see). The window for this example is shown in Figure 8. No change to the CVis display is made at this time.

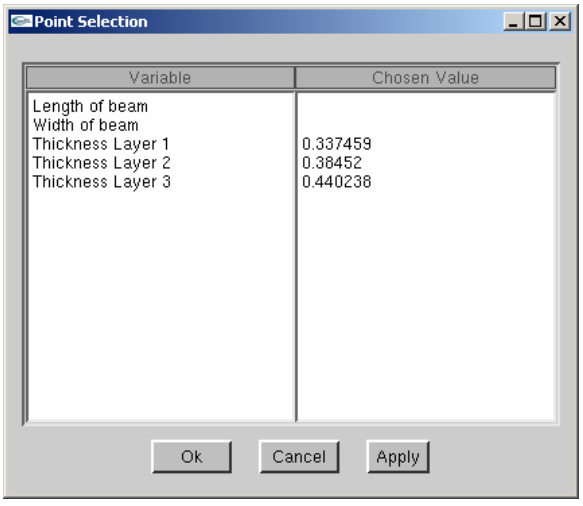

Figure 8 – Incomplete Point Selection Window

The coordinates are actually determined using a line-line intersection calculation for three dimensions. The two lines used are the one shown in Figure 6 (line 1) and another (line 2) created similarly by the second mouse click on line 1. The result of this calculation is the endpoints of the shortest line segment connecting the two lines. The endpoint on line 1 is chosen as the selected point. The two lines are guaranteed to come close to intersecting or no hit record would have been returned on line 1 for the second click. If the user makes their second selection on the same point as the first without rotating the space, the calculation will fail and an error will be presented (infinite intersection).

Further selections along the line will update the values shown in the window.

Having specified the first three dimensions, it is required to specify the remaining two. In two or less dimensions, a single point click is sufficient to determine the desired coordinates. To do this, a line is created (but not shown) in the same fashion as the three dimensional case. The intersection of that line with the plane that contains data is taken to be the desired point. As shown in Figure 9, the point selection is considered complete now.

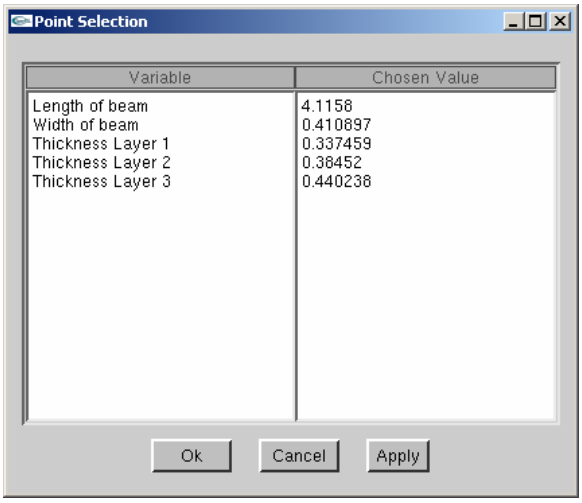

Figure 9 – Complete Point Selection Window

The user can now alter the values to their liking by clicking new points in the spaces.

Finally, to view the point in its new location in space, the user can strike the "Apply" button and the new point will be plotted in the original space as well as all linked spaces of the same type (design or performance). Many points can be entered by repetitive altering and applying. Figure 10 shows a few new points (in pink) after some new selections. Again, the background and bounding box colors have been changed to improve the visibility of the points.

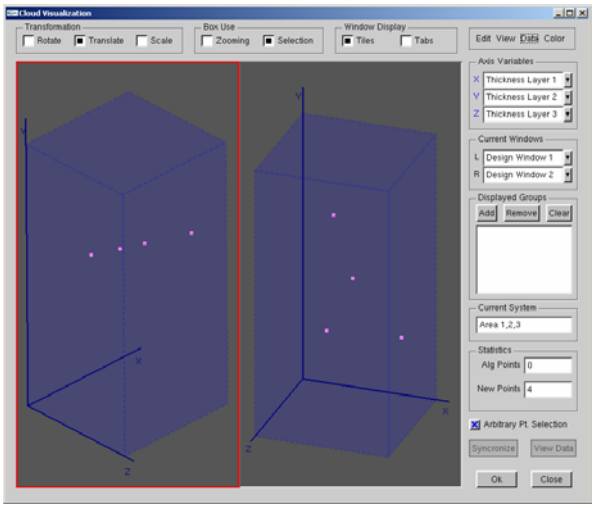

Figure 10 – Display of Newly Selected Points

The newly created points have similar properties to other points and can be selected, clipped, etc. In the preceding example, we specified only the design variable values. The corresponding performance values are not known at this time. CVis is capable of evaluating design points according to the system to which it is attached. However, upon re-entry into an algorithm, the point will have to be assessed for any specific metrics used in that algorithm.

The same process is applied to selection of points in the performance space except now, CVis is unable to do the mapping as explained in the next section.

Mapping Between Spaces

Relating the different spaces possible for a design problem has been a primary focus of our discussion of CVis so far. In this section, we continue with this discussion and move on to the next logical step of mapping newly selected points from the performance space into the design space.

For a given set of equations describing a system, there is a one-to-one mapping from the design space into the performance space. That is, given some set of design parameters, the performance can be singularly determined according to the objective functions. The opposite is not necessarily true, meaning that given some performance level, there may be many design configurations that will suffice. And even if there is only one, the solution is rarely closed form. Mathematically speaking, there are typically more variables (design parameters) than equations (objective functions) in a design problem and the presence of various constraints adds additional challenges. Consider this analogy: given some automobile design, the top speed can be singularly determined for that design based on engine horse power, drag coefficient, etc. However, given a desired top speed, there may be many automobile designs capable of achieving it. Figure 11 demonstrates the topological perspective, that the two domains have a surjective relationship.

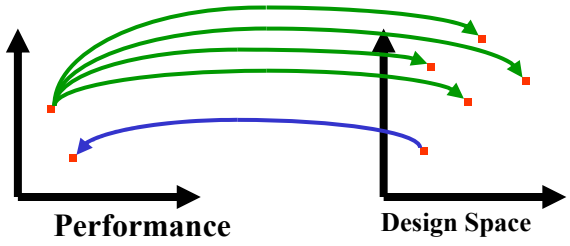

Figure 11: Mapping Between Design and Performance Spaces

As stated previously, CVis cannot perform this mapping. It can however provide the necessary information that may then be used in a goal driven optimization routine to discover designs that provide the desired performance. This approach is not yet automated but is part of the future plans for CVis. It is currently implemented in the following manner.

When a point is chosen in the performance space (as explained previously), the values of the objective functions are used as target values in the goal driven optimization formulation, as follows:

$$\begin{aligned} &\textit{Minimize:} && \sum_{i=1}^{n} \frac{|\, f_i(\bar{x}) - T_i \,|}{|\, T_i \,|} && \\ &\textit{Subject To:} && h_k(\bar{x}) = 0 && k = 1,l \\ & && g_j(\bar{x}) \le 0 && j = 1,m \\ & && x_r^l \le x_r \le x_r^u && r = 1,ndv \end{aligned} \qquad (1)$$

where T_i are the target values provided by cloud visualization for each of the n objectives. All of the constraints in this formulation are the same constraints from the original problem formulation (whatever they may be). Use of this formulation with an approach such as a genetic algorithm makes it possible to find *multiple* design points that satisfy the specified performance requirements. The disadvantage is that this could prove to be very computationally expensive.

It is also possible that the user may wish to specify only some target values and simply extremize the remaining functions. In this case, the unspecified

targets (T's) would be taken out and the absolute values dropped from those terms. Those objectives should however be normalized in some way to prevent them from receiving too much attention in the optimization.

In the next section, a demonstration of CVis for a relatively simple engineering problem is presented. The problem is sufficient to demonstrate the primary features presented in this paper.

Example Problem

The following is a 2-bar truss optimization problem adapted from [18].

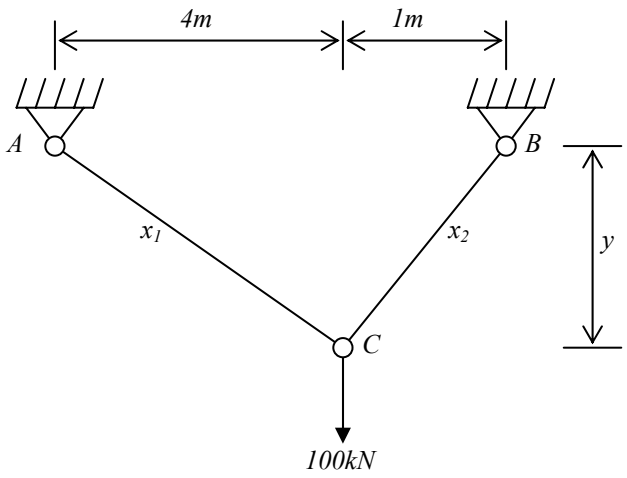

Figure 12: 2-Bar Truss Case Study

The problem is comprised of 3 design variables, 2 objectives, and 3 constraints. The three design variables are the cross sectional areas of bars AC and BC (labeled x1 and x2 respectively) and the overall height of the truss (labeled y). The two objectives are to minimize the overall volume of material used and to minimize the stress in bar AC.

The first two constraints limit the objective function values themselves. They are intended to limit the size of the Pareto set. The Pareto set is unbounded if these constraints are not in place. We have added side constraints to the variables because they are required by our optimizer. They are not part of the original formulation. The side constraints will limit the Pareto set but in a less concise manner than the inequality constraints. The third constraint limits the stress in bar BC to no more than 100,000 kPa.

The formulations are shown in Equations 2.

$$
\begin{aligned}
&\text{Minimize:} && f_{volume} && x_1\sqrt{16+y^2}+x_2\sqrt{1+y^2} \\
&\text{Minimize:} && f_{stress,AC} && \frac{20\sqrt{16+y^2}}{yx_1} \\
&\text{Subject to:} && g_1 && f_{volume} \le 0.1 \\
& && g_2 && f_{stress,AC} \le 100{,}000 \\
& && g_3 && \frac{80\sqrt{1+y^2}}{yx_2} \le 100{,}000 \\
& && && 1 \le y \le 3 \\
& && && 0.0001 > x_1 > 0.1 \\
& && && 0.0001 > x_2 > 0.1
\end{aligned}
\qquad (2)
$$

Figure 13 shows the state of the optimization after only a few seconds of running. All points are displayed, including those that have been discarded as inferior.

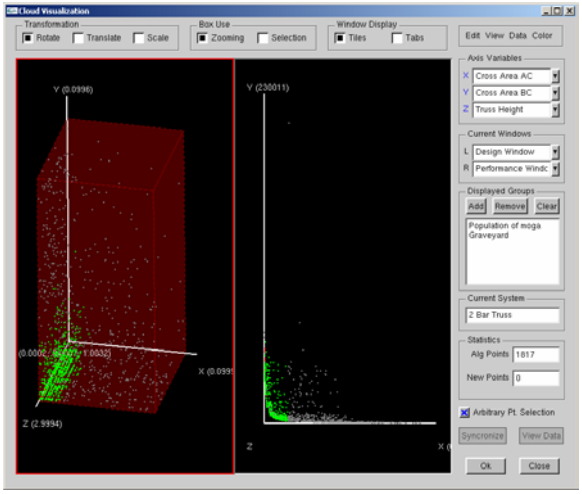

Figure 13 – All Design Points for 2-Bar Truss.

With only two dimensions in the performance space, it is easy to see the non-dominated Frontier and so the next figure shows the performance space zoomed in and the discarded points removed.

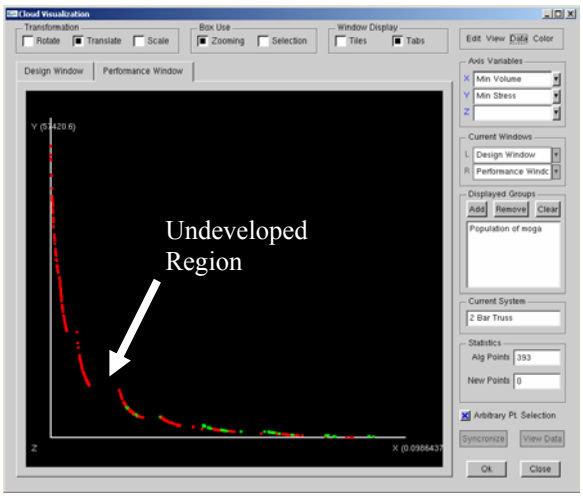

Figure 14 – Performance Space with Undeveloped Region Indicated

From Figure 14, it is apparent that there are regions on the non-dominated frontier that are sparsely populated. In particular, the region specified is clearly undeveloped. At this time, having a good idea of the shape and location of the non-dominated frontier, a user may decide that they would like to find points in that region but do not want to wait for the MOGA to find them. Perhaps they have decided what they would like their final performance values to be and no longer need to expend effort to populate the frontier further.

The pink point in Figure 15 is the newly selected point chosen using the arbitrary point selection capabilities.

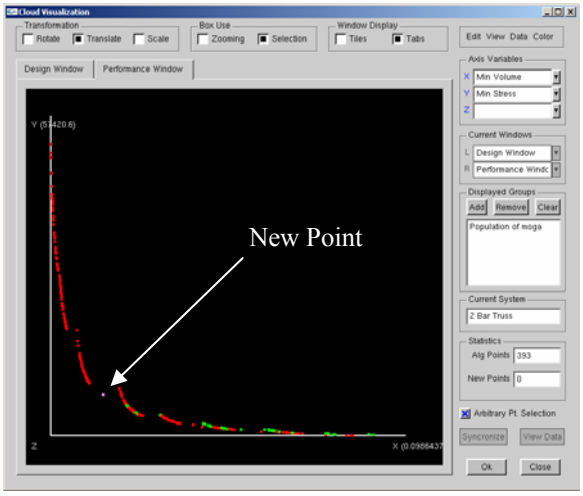

Figure 15 – Newly Selected Point Shown

The chosen performance values are a material volume of 0.021m^3 and a stress in bar AC of 9,700 kPa.

Feeding these values into the optimizer using the formulation of Equation 1 provides the following best (min weighted sum) feasible design configuration.

X_1	X_2	X_3	F_1	F_2
0.0041	0.0009	2.3141	0.02139	9824.5
% Error ->			1.9%	1.3%

This solution came at the expense of over 2,000 function evaluations which was about 1/3 of the total function evaluations performed to generate the points in Figure 13. Although the algorithm was seeking the minimum sum of deviations from the targets, it provided a range of possible designs to choose from.

Captures of the performance and design spaces for the mapping problem are shown in Figures 16 and 17 respectively. The controls have been cropped out to enlarge the spaces.

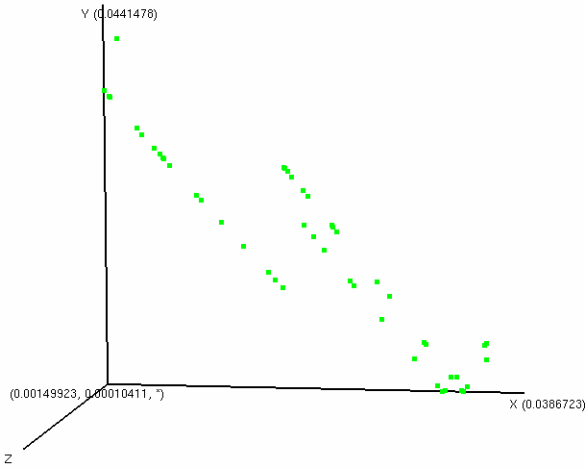

Figure 16 – Performance Space for Mapping Problem Optimization.

The performance space above shows us that there are 100 designs each of which satisfies the objectives by different amounts. If there is to be no more analysis, then the resulting points should be treated as solutions to a multi-objective problem whereby a set of non-dominated points can be identified. Doing this tells us that we have designs to choose from ranging from:

- 0.15% to 3.4% satisfaction of Target 1
- 0.01% to 3.7% satisfaction of Target 2

Looking at the design space below shows us that all the designs are tightly packed and that perhaps a precise solution is possible. If more analysis can be afforded, it may be worthwhile. Perhaps a method other than a GA would be more appropriate for this task.

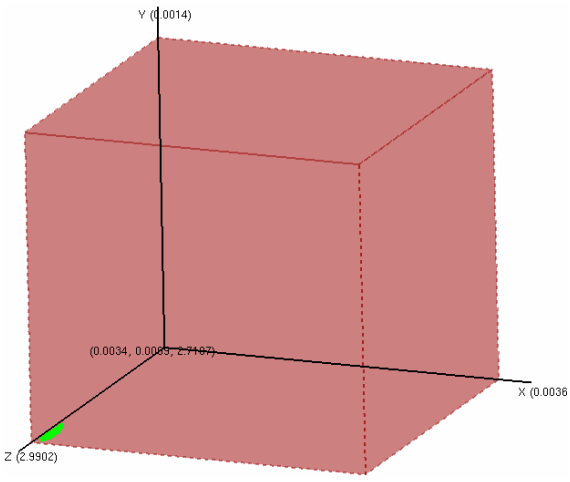

Figure 17 – Design Space for Mapping Problem Optimization.

Conclusions

In this paper, we have presented some of the capabilities of the Cloud Visualization module of our in-house optimization toolbox. They include the abilities to navigate, interact with, and customize the data being presented and the ability to steer an optimization process by providing desired points to investigate. Data from an optimization process can be viewed at any desired point during an optimization run or continuously as the algorithm runs. In addition, information from both the design and performance spaces can be viewed and interacted with at the same time. While the discussions of this paper have focused on multiple objective problems, the developments have been also applied to single objective problems. Single objective problems pose additional challenges of representing solution quality using color in an intuitive manner. However, single objective problems only operate in the design space and avoid the challenges with design and performance space mappings.

Future improvements to CVis include a number of steering operations some of which will be specialized for the algorithm being used. For example, the ability to perform manual selection during a Genetic Algorithm or suggest a search direction when using a gradient based optimization routine.

Acknowledgments

We acknowledge the support of the National Science Foundation, grants DMII-9875706 and DMII-0115444 in this work.

References

[1] Afimiwala, K.A., and Mayne, R.W., 1979, "Interactive Computer Methods for Design Optimization," *CAD Journal*, **11**, pp. 201-208.

[2] Stuart, R., 2001, *Design of Virtual Environments*, Barricade Books, Fort Lee, NJ.

[3] Mecham, M., 1997, "Aerospace Chases the Software Boom," *Aviation Week & Space Technology*, McGraw Hill, October 6, pp. 46-48.

[4] Stewart, P., and Buttolo, P., 1999, "Putting People Power into Virtual Reality," *Mechanical Engineering Design*, ASME, pp. 18-22.

[5] Furlong, T. J., Vance, J. M., Larochelle, P. M., 1999, "Spherical Mechanism Synthesis in Virtual Reality", *Journal of Mechanical Design*, **121**, pp. 515-520.

[6] Beazley, D.M., and Lomdahl, P.S., 1996, "Lightweight Computational Steering of Very Large Scale Molecular Dynamics Simulations," presented at *Supercomputing*, Pittsburgh, Pennsylvania.

[7] Parker, S., Weinstein, D., and Johnson, C., 1997, **"**The SCIRun Computational Steering Software System," *Modern Software Tools in Scientific Computing*, pp. 1-40.

[8] van Liere, R., Mulder, J.D., van Wijk, J.J., 1996, "Computational Steering," presented at *HPCN Europe*, Brussels.

[9] Winer, E.H. and Bloebaum, C.L., 2001, "Visual Design Steering for Optimization Solution Iimprovement," *Structural and Multidisciplinary Optimization*, **22**(3), pp 219-229.

[10] Winer, E.H. and Bloebaum, C.L., 2002, "Development of Visual Design Steering as an Aid in Large Scale Multidisciplinary Design Optimization - Part I and II," *Structural and Multidisciplinary Optimization*, to appear.

[11] Messac, A., and Chen, X., 2000, "Visualizing the Optimization Process in Real-Time Using Physical Programming," *Engineering Optimization Journal*, **32**(5).

[12] Mecham, M., 1997, "Raytheon Integrates Product Development," *Aviation Week & Space Technology*, McGraw Hill, October 6,

p. 50.
[13] Helig, M. H., 1992, “El Cine del Futuro: The Cinema of the Future,” *Presence*, **1**(3), pp. 279-292.
[14] Eddy, W.F. and Mockus, A., 1996, “Dynamic Visualization in Modeling and Optimization of Ill Defined Problems,” In *State of the Art in Global Optimization: Computational Methods & Applications*, eds. Floudas, C.A., and Pardalos, P.M., Kluwer Academic Publishers, Boston, pp. 499-520.
[15] Bangay, S., 1998, "Visiview, A System for the Visualization of Multi-Dimensional Data,” *IS&T/SPIE Conference on Visual Data Exploration and Analysis V*, Vol. 3298 - 0277-786X/86/98.
[16] Tufte, E.R., 1997, *Visual Explanations*, Graphics Press, Cheshire, Connecticut.
[17] Tufte, E.R., 2001, *The Visual Display of Quantitative Information*, Graphics Press, Cheshire, Connecticut.
[18] Azarm, S., Reynolds, B., and Narayanan, S., 1999, "Comparison of Two Multiobjective Optimization Techniques with and within Genetic Algorithms," Proceedings of the 25th ASME Design Automation Conference, Paper No. DETC99/DAC-8584, Las Vegas, NV.