Semi-Automatic Antenna Design Via Sampling and Visualization

A. Quigley, D. Leigh*, N. Lesh, J. Marks, K. Ryall, K. Wittenburg

TR2002-02 February 01, 2002

Abstract

Optimization-based approaches to antenna design have enjoyed limited success. The task is often computationally intractable. Moreover, it is also often difficult to capture all relevant design issues and tradeoffs in a single mathematical objective function. Therefore, human experts typically specify and refine antenna designs b̈y hand,üsing computers only to evaluate their candidate designs by simulation. In this paper we propose a middle ground between this traditional approach and fully automatic optimization. We use computation to sample a space of possible antenna designs, and we rely on human judgment to select good designs from the samples returned by the computer. The key elements of our computer system are a parallel algorithm for intelligently sampling a space of possible antenna designs and a graphical user interface for visualizing and exploring candidate designs and managing the sampling process.

IEEE Antennas and Propagation Society International Symposium (AP-S/URSI)

Submitted January 2002; revised and released March 2002.

Semi-Automatic Antenna Design Via Sampling and Visualization

A. Quigley, D. Leigh*, N. Lesh, J. Marks, K. Ryall, K. Wittenburg
Mitsubishi Electric Research Laboratories
201 Broadway, Cambridge, MA 02139, USA
{aquigley,leigh,lesh,marks,ryall,wittenburg}@merl.com

Abstract

Optimization-based approaches to antenna design have enjoyed limited success. The task is often computationally intractable. Moreover, it is also often difficult to capture all relevant design issues and tradeoffs in a single mathematical objective function. Therefore, human experts typically specify and refine antenna designs "by hand," using computers only to evaluate their candidate designs by simulation. In this paper we propose a middle ground between this traditional approach and fully automatic optimization. We use computation to sample a space of possible antenna designs, and we rely on human judgment to select good designs from the samples returned by the computer. The key elements of our computer system are a parallel algorithm for intelligently sampling a space of possible antenna designs and a graphical user interface for visualizing and exploring candidate designs and managing the sampling process.

Introduction

The idea of using computer-based optimization for design tasks has been applied to many problems, including antenna design. However, this idea does not always work well: the optimization problems are often intractable; and it often proves impossible to consider all relevant design criteria in the optimization process. In this paper we propose that the computer be used differently. Instead of having the computer search for a single optimal design, we program it to intelligently sample the large space of possible antenna designs, subject to user-supplied constraints. The task of choosing a final design from the computer-generated sampling is left to the human user, who can apply experience and judgment to recognize and then refine the most useful antenna design [1].

Our Approach

An overview of our system is shown in Figure 1. In Step 1 the user supplies an initial *antenna specification* that describes the antenna geometry and other physical-parameter inputs. This specification also serves as a starting point for the sampling process. In addition, the user also indicates which variables of the specification are free to be varied by the computer (the *free variables*) and minimum and maximum values for these variables. This information is specified in an XML file that can be edited manually.

In Step 2 the computer parses the XML file and creates an initial random sample of antenna specifications by sampling the free variables uniformly over their valid ranges. However, a uniform sampling of the free variables will rarely produce a representative sampling of antenna designs. To generate a representative sample of designs requires an intelligent sampling process that we call *dispersion*, which is Step 3.

A key requirement for dispersion is a mathematical function that quantifies the difference between two antenna designs. This difference metric is usually based on the performance characteristics of an antenna. We encode the performance characteristics of an antenna in a *performance vector* containing *m* real numbers that represent, for example, the antenna's gain, front-to-back ratio, front-to-sidelobe ratio, cost (total wire length), half-power beam width, and voltage standing-wave ratio (VSWR) for a given design impedance. In our prototype system, we compute the performance vector from an antenna specification with the NEC-2 simulator, though in principle any simulator could be used. We define the difference between two antennas to be the Euclidean distance between their two weighted *m*-dimensional performance vectors. The goal of the dispersion process is then to produce a sample of antenna designs for which the associated performance vectors are as broadly distributed as possible in the *m*-dimensional *design space*. The algorithm for accomplishing this is shown in Figure 2. In each iteration a new antenna specification is generated by perturbing the free variables of a specification already in the sample. The performance vector for this new candidate specification is computed by the simulator. If the new performance vector contributes more to the diversity of the sample than some antenna already there, then the former antenna replaces the latter in the sample. Our dispersion process can therefore be thought of as an evolutionary or genetic algorithm that is governed by a "maximal diversity" rule instead of a survival-of-the-fittest rule.

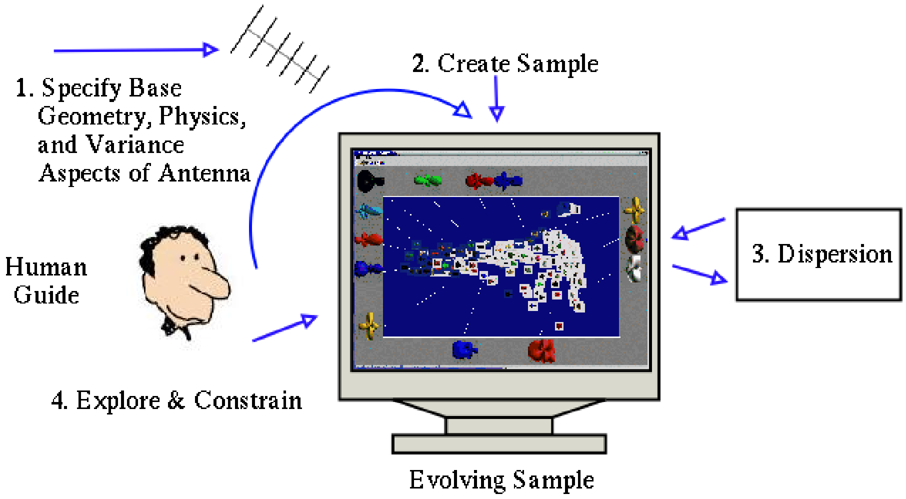


Figure 1. System overview.

The dispersion process requires many calls to the antenna simulator and is embedded in an interactive system, which mandates some degree of system responsiveness. We therefore parallelize the dispersion process by distributing simulator calls over the 132 nodes of our Beowulf cluster, with which we achieve near-linear speedup. The resulting parallel algorithm is a minor variation of the serial version described in Figure 2.

The first invocation of the dispersion process typically produces a wide variety of designs. In Step 4 the user then explores this sampling to locate and identify the most useful designs. This exploration process is facilitated by a graphical interface, shown in Figure 3. The central panel contains thumbnail images of gain plots for each antenna. The color of the plot indicates the value (low, medium, or high) of some significant scalar value, in this case the VSWR performance measure. The thumbnails are positioned so that antennas with similar performance vectors lie close to each other. In other words, distance in the display correlates with distance in the *m*-dimensional design space implied by the difference metric. This layout, which is computed using a technique called multidimensional scaling [1], allows the user to form a "mental map" of the design space.

The thumbnail display can be browsed by panning and zooming. The user can bookmark interesting antennas by moving them to the surrounding "gallery." Mousing over a bookmarked antenna causes its corresponding thumbnail to be highlighted, and vice versa. (The lines connecting bookmarked antennas to their thumbnails in Figure 3 do not appear in the actual interface, but are shown here for exposition.) A bookmarked antenna can be investigated further by clicking on it, which brings up additional windows in which details of the antenna's design can be examined.

Input:
A set S of antenna specifications; their corresponding performance vectors V; and an allowable region R of the space of performance vectors.

Output:
Modified sets S and V.

Algorithm:

```
DISPERSE($S$, $V$, $t$)
  for $i \leftarrow 1$ to $t$ do
    $j \leftarrow rand\_int(1, |S|)$;
    $s \leftarrow perturb(S[j],i,)$;
    $v \leftarrow simulate(s)$;
    if $v \in R$
      add $s$ to $S$
      add $v$ to $V$
      $k \leftarrow worst\_index(V)$;
      remove $S[k]$ from $S$;
      remove $V[k]$ from $V$;
```

Notes:

rand_int(1, *n*) returns a random integer in the range [1, *n*].

perturb(*S*[*j*], *i*) returns a copy of *S*[*j*] in which all the free variables have been perturbed. The magnitude of the perturbations is inversely proportional to *i*, meaning they decrease over time.

simulate(*s*) is a simulator code that computes a performance vector *v* given an antenna specification *s*. In our system it is the NEC-2 simulator.

worst_index(*V*) returns the index of the performance vector in *V* that has the minimum nearest-neighbor distance. Ties are broken using the average distance to all vectors in *V*.

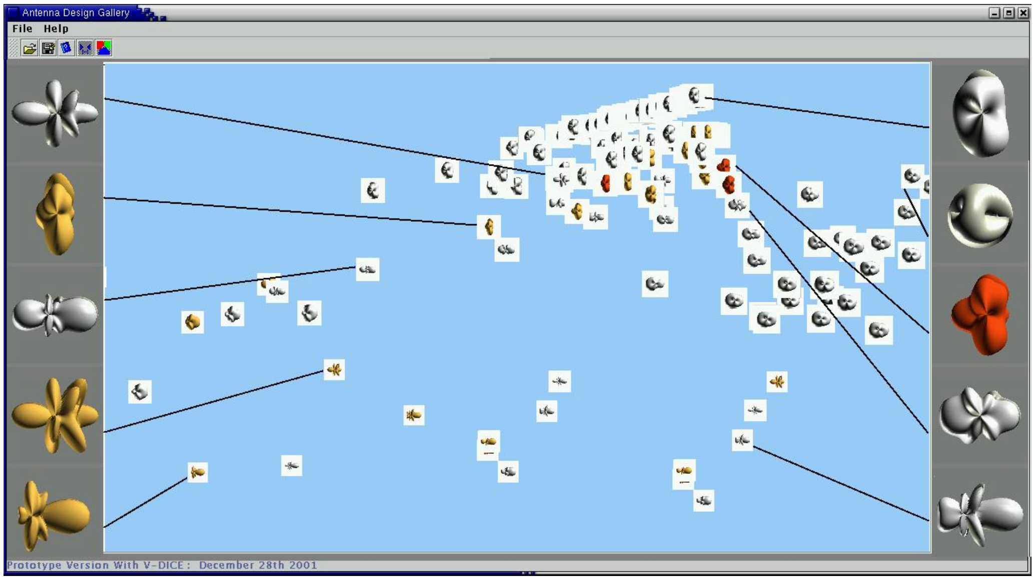
Figure 2. The dispersion process.

Besides presenting a visualization that clusters similar antennas, the system also affords users the opportunity to explore the tradeoffs between the antennas' numeric performance measures [2]. Each of the sliders in Figure 4 corresponds to one dimension in the performance vector. Each antenna in the current sample is shown on each dimension as a vertical line. Notice the relatively even distribution of values along each dimension: this is a consequence of the dispersion algorithm. Users can select subranges within the dimensions, thereby creating a visual query. The resulting selection is reflected immediately by highlighting in the thumbnail space. Furthermore, the selection is also shown in the dimension rows by fading the value bars of unselected antennas. This allows the user to perceive relationships between different performance measures and thus better understand design tradeoffs. For example, selecting the higher-gain antennas shows the expected clustering in cost (total wire length) as well as the low VSWRs these antennas would achieve when fed with the design impedance of 100 Ω (see Figure 4).

Having explored performance tradeoffs, the user can make another visual query to determine the starting sample to which the dispersion algorithm will be applied in the next round of sampling. Moreover, this query also delineates the region *R* of acceptable performance vectors (see Figure 2): any candidate design that falls outside *R* is rejected. The sample from the next round will therefore be concentrated in the regions of the design space of interest to the user, thus increasing the likelihood that it will contain a desirable antenna. Eventually, when the sample is sufficiently focused, the user may invoke a standard optimization algorithm to perfect some of the antennas in the sample by looking for their nearest locally optimal designs.

Current Status & Future Work

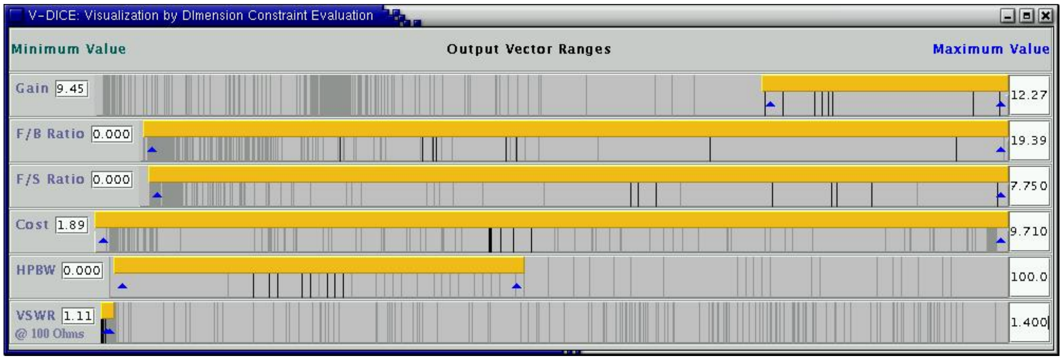
We have completed implementation of the system prototype above. We are testing the system on a variety of antenna design problems to ascertain its strengths and weaknesses. Future work will involve incorporating a more sophisticated simulator into the system.



Antenna Design Gallery

Prototype Version With V-DICE : December 28th 2001

Figure 3. Visualizing a sample of antenna designs.



V-DICE: Visualization by DImension Constraint Evaluation

Minimum Value | Output Vector Ranges | Maximum Value

Gain 9.45 | 12.27

F/B Ratio 0.000 | 19.39

F/S Ratio 0.000 | 7.750

Cost 1.89 | 9.710

HPBW 0.000 | 100.0

VSWR @ 100 Ohms 1.11 | 1.400

Figure 4. The interface for exploring performance tradeoffs via visual queries.

References

[1] J. Marks et al., "Design Galleries: A General Approach to Setting Parameters for Computer Graphics and Animation," in Proceedings of ACM SIGGRAPH 97, pp. 389-400, Los Angeles, CA, Aug., 1997.

[2] K. Wittenburg, T. Lanning, M. Heinrichs, M. Stanton, “Parallel Bargrams for Consumer-based Information Exploration and Choice,” in Proceedings of ACM UIST 01, Orlando, Florida, Nov. 2001.