Automatic Synthesis of an 802.11a Wireless LAN Antenna Using Genetic Programming

A Real World Application

Rian Sanderson

SiRiFIC Wireless
Santa Rosa, CA 95409
rsanderson@sirific.com

Abstract. This paper describes the application of genetic programming to synthesize a small form factor, 2 dimensional wire antenna for a 5.2 GHz 802.11a wireless LAN application. Utilizing basic genetic programming techniques and using a novel approach to discretizing the search space leads to a more simple problem representation and implementation. The paper presents simulation results exceeding expectations and that are competitive to commercial products.

1 Introduction

The enigmatic nature of radio frequency (RF) engineering, the problem domain's large search spaces, and the availability of simulators are all factors in the frequent choice of evolutionary algorithms [1,2,3] for synthesizing and optimizing antennas. Specifically, genetic programming (GP) is well suited to this domain when antenna geometries are represented with an implementation of the LOGO drawing language [2, 3].

This paper uses GP to create a 5.22 GHz antenna for an 802.11a wireless LAN application. An efficient antenna, one with good gain and low voltage standing wave ratio (VSWR), is an important part of an 802.11 system; it can extend the reach of signals, or prolong battery life by requiring less transmission power.

1.1 What Is Genetic Programming?

To paraphrase, [2] genetic programming is a technique for automatically creating computer programs which solve, or approximately solve, problems by utilizing a high level statement of the structure's desired characteristics. Genetic programming has been successfully used for sequencing proteins, synthesizing electronic circuits, creating robot control algorithms, and other difficult tasks. Genetic programming is further explained by Koza [5].

K. Deb et al. (Eds.): GECCO 2004, LNCS 3103, pp. 1201–1213, 2004.

1.2 Why an 802.11a Antenna?

The IEEE 802.11 standards are the most popular wireless networking standards for home and enterprise. IEEE 802.11b/g [6], in the 2.4 GHz band, has a low cost and is prevalent in homes, school campuses, airports and coffee shops. IEEE 802.11a [7] with its higher data rates, reduced range, and higher cost, is geared more for the enterprise and is not yet as prevalent.

Antennas for 802.11a are difficult to find, there are far more antennas for 802.11b/g. Standard, low gain, 5/8 wavelength, omni directional whip antennas are a feature on most access points. Yaego, a manufacturer of RF components, has a discrete ceramic 802.11a antenna [11] available from major parts suppliers. Murata also has a discrete ceramic antenna, the LDA131 [12], but only releases it and its data under non disclosure agreement. Due to their size and fixed position application, Centurion Technologies Whisper 5 GHz [13], at 6.4 x 6.4 x 1.7 cm, and the Hyperlink Technologies HG5308P and HG5311P [14], at 11.4 x 11.4 x 2.5 cm, are the most comparable antennas to this research. All these antennas work in the range from 5.18 GHz – 5.35 GHz.

1.3 Other Research

As its name implies, Electromagnetic Optimization by Genetic Algorithms, is a whole text describing genetic algorithm (GA) optimization of various RF problems, but the chapter by Altshuler and Linden, [1] describes automatic synthesis of “crooked wire antennas” using a GA. This successful antenna work focuses in the 1600 MHz range and uses the NEC2 simulator.

Lohn et al [3] use both a conventional GA encoded in real numbers, and a branching GA with elements of the LOGO drawing language. This branching GA is close to a GP representation with two functions, forward and rotate, each taking a real valued argument. Their successful 8470 MHz antenna, with a footprint about the size of a quarter, is scheduled for launch with NASA’s Space Technology 5 space craft.

Comisky et al [2] describe the rediscovery, by GP, of a 424 MHz Yagi Uda type antenna which fits in a bounding rectangle .4 meters by 2.65 meters. They utilize the LOGO drawing language to represent antenna geometries and directly compile in the source for a later version of the NEC2 simulator, NEC4. They use real valued quantities for wire lengths and angles, and have a more broad function set utilizing function argument constraints. To deal with simulator geometry restrictions they employ a complex wrapper which applies a lethal fitness to individuals violating geometry constraints, and which replaces wire intersections with 4 new wires.

To varying degrees these works preconceive what the final antenna will look like. While Altshuler and Linden [1] do not specify the configuration of wires, their GA implementation fixes the number of them at 7. Lohn et al [3] use the GA to design one arm, which will is symmetrically repeated, in a wire antenna with four arms separated at 90 degrees. Comisky et al [2] had a goal to not preconceive the solution as a Yagi Uda antenna, but the extra weight on elements to create separate wires, the choice of landmark functions and “no wire” terminals, and the symmetrical reflection of the generated geometry both lead toward a Yagi Uda type solution.

Based on this research we were confident we could make an antenna, but questioned: the quality of VSWR and gain we could find, if the targeted physical size was too small, and if our lack of predefined antenna structure or symmetry would hamper results.

2 The Problem

The problem is to synthesize a high gain, low VSWR, two dimensional, continuous wire antenna for the lower sub band (frequency range 5.18– 5.24 GHz) of the 5 GHz FCC Unlicensed National Information Infrastructure (U-NII) band which is no larger than 6 cm square, and made of American wire gauge (AWG) #19 wire. The antenna is simulated using the freely available NEC2 simulator at a single representative frequency of 5.22 GHz, in free space, assuming a lossless conductor.

Rather than using real valued quantities for wire sections and angles, the 6 cm square problem space is discretized into a matrix of 15 by 15 points. As Figure 1 shows, a continuous wire antenna can be viewed as a collection of small wires each joined at a point in the matrix.

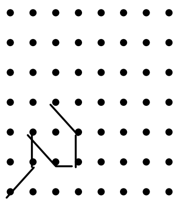

Fig. 1. An antenna viewed as the connection of six individual segments

NEC2 has many conditions regarding allowable geometries [8] with respect to wavelength, wire radii, and segment length. By spacing the matrix points appropriately the number of unsimulatable individuals can be minimized. For this problem, with its 5.7 cm wavelength signal on AWG #19 wire, 4 mm is the minimum spacing.

We discretized the search space in an intent to obviate the need for complicated geometry verification, as in [2]. However we ended up having to employ a simple geometry check to look for improper segment crossing, duplicate segments, and segments of zero length.

Segment crossing is the major simulator condition that the quantization does not protect against. Figure 2 demonstrates valid and invalid junctions.

NEC2 is based on the method of moments [8], and the simulator cannot compute the current on overlapping segments due to an indeterminate matrix calculation.

Discretizing the problem space puts few limits on what the end antennas will look like. While in this implementation all antennas are continuous and non branching, it is the choice of terminals which does not allow discontinuity, and the handling of duplicate segments which prevents branching. Furthermore, the problem space could readily be extended to 3 dimensions, where discretizing would help prevent even more geometry conflicts.

Fig. 2. Wires may only cross at end points, not in the middle of segments

Though bounded, the total size of the search space is difficult to calculate. Calculating the permutations of all possible wire segments is not correct; each segment must share at least one end point with another segment, leaving no stray unattached segments. Also, crossed segments, as discussed previously, must not be counted, nor can antennas with branches.

3 Methods

The concept and fitness measure are closely modeled after Comisky et al [2], though our simplifications render the architecture similar to Koza's artificial ant problem [5]. The methods are summarized in Table 1 and discussed in detail in the following sections.

3.1 Architecture

The simple architecture of one result producing branch, the connective functions progn2 and progn3, and terminals which actually do the work, bears similarity to Koza's implementation [5] of the Artificial Ant problem. The connective functions, progn2() and progn3(), evaluate each of their arguments in order, and return no value. The real work, output of LOGO instructions, happens as a side effect of evaluating the terminals at the leaves of progn trees.

The rotate terminals, rotxy 45, rotxy 90, rotxy 315, rotxy 270 change the direction of the turtle in the X-Y plane, and were chosen with the thought of expanding the experiment to 3 dimensions. While not strictly necessary, all four rotate terminals seem appropriate when looking at the problem from a human's point of view. In the actual results, rotate operators are often stacked five or more deep, causing the turtle to spin several revolutions to change perhaps 45 degrees.

The move terminal is the only operation that creates wires. It advances the turtle to the next matrix point in the current direction, creating a wire segment from the point at the start of the operation to the point at the end of the operation. Moves that will take the turtle outside of the matrix are ignored and become no-ops.

Table 1. Tableau sumarizing methods

Objective:	synthesize a two dimensional continuous wire antenna for the Lower UNII band (frequency range 5.18– 5.24 GHz) which is no larger than 6 cm square, and made of American wire gauge (AWG) #19 wire.
Architecture:	One result producing branch
Functions:	progn2(x,y), progn3(x,y,z)
Terminals:	move, rotxy 45, rotxy 90, rotxy 315, rotxy 270
Fitness Case:	simulation with NEC2 at a single frequency of 5.22 GHz, generating radiation data for a full 3 dimensional plot with 5 degree increments
Raw Fitness:	-maxgain + V(vswr) ; smaller fitness is better
Standardized Fitness:	-success predicate + Raw Fitness
Hits:	not used
Wrapper:	ignores out of bounds move instructions. Geometry checker: crossed segments, zero length segments, duplicate segments
Selection:	tournament with size of 7, .9 probability of cross over, .1 probability of mutation
Parameters:	M=1000,G=100; M=5000,G=40; M=6500,G=35; M=1000, G=30
Success Predicate	exit at fitness of -14

3.2 Fitness and Selection

Simulating at only one frequency cannot be called rigorous, but it speeds computation. A more robust fitness case, such as [2], would run the simulation over a frequency range and use the worst case.

The raw fitness measure (1) combines the two most important antenna criteria, the maximum gain and the voltage standing wave ratio (VSWR), to try and get the most negative number possible

$$\text{fitness} = -\text{maxgain} + V(\text{vswr}) \tag{1}$$

$$V(\text{vswr}) = \text{vswr} * C, \tag{2}$$

where

$$C = .1 \text{ for vswr} <= 2,$$
$$C = 1 \text{ for vswr} <= 3 ,$$
$$C = 10 \text{ for vswr} > 3 .$$

This fitness measure is based on work by Altshuler and [1] as well as Comisky et al [2], though an even higher penalty for poor VSWR was assessed because VSWR is critical in 802.11 systems. Early on in the process the VSWR term dominates, but to get a really negative (better) fitness the antenna must have a high gain.

The voltage standing wave ratio (VSWR) [9] is a measure of reflective wave interference, and can be thought of as how well matched an antenna input is to the transmission line feeding it. As its name suggests, VSWR is a ratio; a value of 1.0 means all energy is radiated from the antenna and it reflects nothing back to the feeding source. A VSWR of infinity radiates all energy back to the line feeding the signal and radiates no energy in its intended direction. The VSWR for a particular antenna is not a single quantity; it is a function which varies across the excitation frequency. When VSWR is mentioned as a single quantity in this paper it is at the simulated frequency of 5.22 GHz with a 50 ohm input impedance. A VSWR greater than 3 is unacceptable in a WLAN system.

Gain describes an antenna's ability to apparently amplify a signal [9]. The higher the gain the more distant a signal of a given power may be transmitted or received. Gain is measured in decibels relative to isotropic (dBi); where an isotropic antenna is an ideal antenna which radiates equally in all directions. Standard whip antennas have gains around 2 dBi, directional antennas such as the biquad have gains near 10 dBi.

Though not explicitly included in the measure of fitness, beam width is also an important characteristic. Beam width, measured in degrees, describes how wide the gain pattern is dispersed. Outside the arc of beam width, signals are transmitted at less than half the power as inside the beam width. Antennas with higher gain tend to have a more narrow beam width. The success predicate indirectly affects beam width, because it terminates the run before the antenna overspecializes with a super high gain and narrow beam width.

We employ a geometry checker as a wrapper to screen individuals to ensure they do not violate simulator constraints. The geometry checker looks for crossed segments, segments of zero length, and duplicate segments. Rather than try and fix each of these problems, as in the simple case of ignoring move instructions when they happen, the geometry checker flags the individual as a geometry error, giving it an enormously large, and lethal, fitness value.

Standardized fitness, a transformation of raw fitness so that 0 is the most fit, is necessary for the breeding process, and is achieved using the success predicate term.

3.3 Choice of Parameters

Based on results from Koza [10] tournament selection is used with .9 probability of crossover and .1 probability of mutation.

Initially we chose values for M and G, population and number of generations, with a 12 hour run as the target. After hypothesizing that genetic diversity was being lost due to excessive culling by the geometry checker we increased the population size and found faster servers to run on. Initial results ran in about .4 seconds per individual(M1000, G100 in approximately 11 hours), faster hardware yields evaluation times of .16 seconds per individual (M6500, G35 in approximately 10 hours). Improvement in large M runs started to level off around generation 30.

4 Structures Undergoing Adaptation

The trees of functions and terminals that actually undergo adaptation bear no resemblance to the antennas they generate. Figure 3 shows a portion of the program tree for a random individual of generation 0.

```
(progn2(progn2 (progn3 (progn2 (progn2 rotxy 90 rotxy 90)
(progn3 rotxy 90 rotxy 45 move))
(progn3 (progn2 (progn3 rotxy 315 rotxy 270 move)
(progn2 rotxy 90 move))
(progn3 rotxy 90 move rotxy 270 )
(progn3 rotxy 90 rotxy 270 rotxy 45))
```

Fig. 3. A portion of a program tree from an individual in generation 0

The individual in Figure 3, like most others, wastes effort spinning around in circles.

The antennas are much easier to view in their evaluated structure, as Figure 4 demonstrates. This average individual has a reasonable gain of 4.79 dBi, but its dismal VSWR of 240 contributes most to its raw fitness of 2395.21.

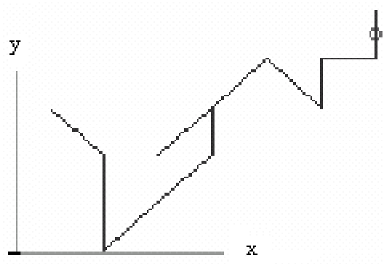

Fig. 4. Average individual from generation 0 with raw fitness of 2395.21. The branched portion is a sign of a duplicate segment, and the latest geometry checker would tag it with lethal fitness

The starting wire where the excitation signal is fed to the antenna is represented with a small circle. Note that in this run the individual started from the middle of the grid, later runs put the starting wire at the lower left. We changed the starting position for figure clarity rather than any performance difference.

As stated previously, a wire is created between two points upon the execution of a move instruction and rotate instructions don't actually create wires, they only select which point the next move instruction will connect to.

Figure 4 is generated from a tree that condenses into: rotxy 90, move, rotxy 270, move, rotxy 90, rotxy 45, move, rotxy 270, move, move, rotxy 90, rotxy 90, move, rotxy 90, rotxy 45, move, rotxy 45, move, move, rotxy 90, rotxy 45, move, move, rotxy 315, move.

The branched portion in the middle of figure 4 demonstrates an illegal duplicate segment. The turtle creates a segment by moving forward, rotating 180 degrees, then moving again. This is from an early run without the geometry check, which would tag it with lethal fitness.

5 Implementation

A foundation of this project is the quantizing of the problem space. In conjunction with the scaling feature of NEC, the quantization allows direct writing of matrix points to the NEC input file. This creating a one to one mapping from the theoretical points of the matrix and their actual mapping in the input file. In addition to bounding the search space, the discretization also fixes the two variables of wire size and segment length, further narrowing the dimensionality of the problem.

We use the lil-gp genetic programming toolkit for taking care of the grunt work involved in breeding, program tree representation, and program tree execution. We had little difficulty getting this C++ toolkit to run under Linux with gcc 2.9, WIN32 Cygwin with gcc 3.3.1, and native WIN32 with MSVC 6.0.

We ran NEC2 v2.3 for the simulator, compiling it for both Linux and WIN32 as a separate application. Under WIN32 the supporting 4nec2 program plotted antennas, drew gain fields, and gave instructional geometry warnings on models.

Our data flow is disk intensive, but writing to a file at each stage made modular development and discrete unit testing possible.

The evaluation of an individual is staged over several steps. The data flow starts with evaluating the GP tree to generate a LOGO instruction file as show in Figure 5.

```
rotxy 315
rotxy 90
rotxy 45
rotxy 45
move
rotxy 45
move
rotxy 270
```

Fig. 5. LOGO instructions generated from a GP tree

The LOGO instructions are easier for a person to understand than the original GP tree, but still difficult to visualize.

The parser converts from LOGO to NEC input, shown in Figure 6. The first wire is a fixed stub for the excitation connection and its second endpoint is the starting point for the first GP generated instruction. This fixed wire also allows for an individual which is made up of only rotate instructions.

```
CM Copyright (c) 2003
CM GP generated 802.11a antenna, 4 mm segs, #19 wire
CM TAG, #Segs, x1, y1, z1, x2, y2, z2, radius
CE
GW 1 1    0  0  0    1  1  0    .120
GW 2 1    1  1  0    2  1  0    .120
GW 3 1    2  1  0    3  2  0    .120
GW 4 1    3  2  0    2  3  0    .120
GW 5 1    2  3  0    1  4  0    .120
GS 0 0 .004
GE
EX0 1 1 0 1 0
FR0 1 0 0 5220
RP 0 73 73 1000 -180.0 0.0 5.0 5.0
EN
```

Fig. 6. GP generated NEC input file

In the lines starting with GW (generate wire card), notice the direct matrix coordinates in the NEC input file, as mentioned previously. The starting segment goes from (0, 0, 0) to (1, 1, 0). The direct coordinates are possible due to the problem space discretization and the 4 mm scaling operation of the GS card. Looking at this file a person can get a better sense of where the antenna is going: the first wire moves up and to the right, the second goes right, the third goes up and to the right again.

After translation, the geometry checker is run on the .nec input file. On a successful geometry check the input is sent to the NEC2 simulator via a system call to generate a .out file. Finally the fitness evaluator parses this .out file to compute a raw fitness.

The .out file can then be reviewed interactively outside of the GP run using 4nec2. The 4nec2 software will plot the antenna geometry for easy viewing, give a three dimensional radiation plot, and plot VSWR over a range of frequencies.

5.1 Problems Encountered

Error checking on file opening and closing, programming niceties usually shirked by most programmers, turned out to be important implementation details. Early runs failed due to a filled user disk quota and segmentation faults after opening too many file handles.

We added a duplicate segments check to the geometry screening after finding wildly successful data then discovering that duplicates were a serious error. 4nec2 does not warn about duplicate segments, and literature does not explicitly forbid them, however there is no physical way to represent them. Doubling the wire diameter for that segment violates the minimum spacing between wire segments rule, and also increases the VSWR. Removing the duplicate segments from a final model increases the VSWR above 10, invalidating results.

6 Results

All best of run individuals show convergence and improvement, with several runs producing excellent high gain, low VSWR antennas. One run exceeded our expectations with high gain and low VSWR in a frequency range doubling the problem specifications.

The best antenna synthesized, run 404803468, is pictured in figure 7 with its full radiation pattern and VSWR across the band. This antenna's performance far exceeds the Yaego chip antenna, it is within .6 dBi of the Hyperlink HG5308P, and 1.6 dBi of the Centurion Whisper 5GHz. Though a distance from the 11 dBi gain of the HG5311P, that antenna's narrow beam width of 60 degrees horizontal and 30 degrees vertical makes it more difficult to use. VSWR for antenna 404803468 creeps slightly above 2 at the bottom of the band but its average VSWR of 1.62, with a minimum of 1.27, is as good or better than the others. Its approximate size of 6 cm x 2.3 cm, one tenth the area of the HG5308P/HG5311P and one third the area of the Whisper 5GHz arguably makes it a portable or embeddable antenna.

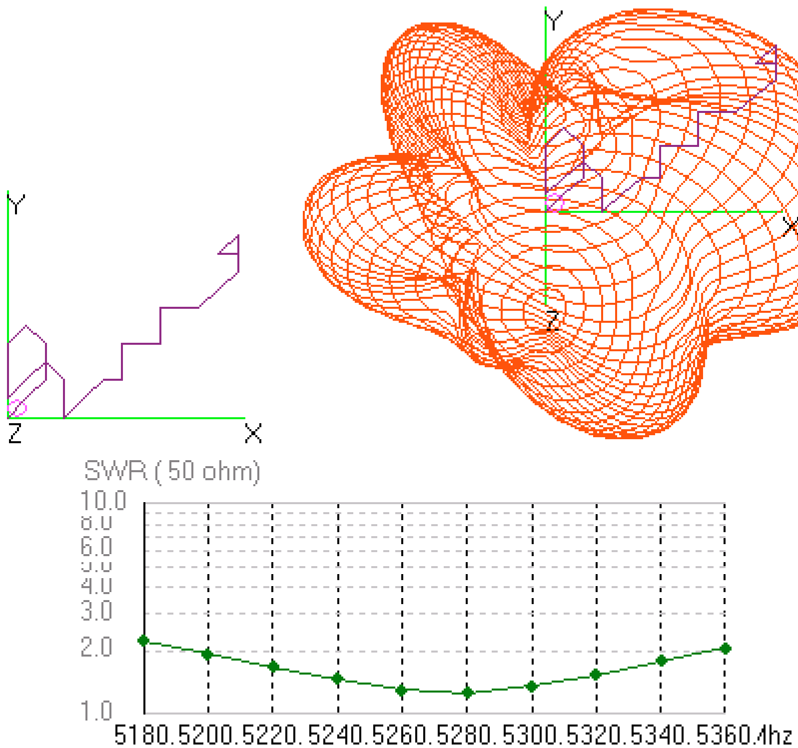

Fig. 7. The best antenna synthesized, from random seed 404803468. It has a max gain of 7.42 dBi, a VSWR of 1.68 at the fitness frequency, and a minimum VSWR of 1.27

Beyond 404803468's raw fitness, it has several desirable traits. The gain pattern is well focused, yet not overly narrow, with a beam width of 90 degrees vertical and 45 degrees horizontal. The simple geometry lends itself to more accurate construction. Most importantly, its VSWR is under 2.0 for twice the frequency range of the other GP generated antennas.

Along with antenna 404803468, Table 2 shows several other competitive and interesting antennas. Antenna 624971 (population 5000) has the second best fitness, but its complicated shape and VSWR past 5.24 GHz limit its use. Antenna 392561, though the worst in raw fitness, is actually a decent omni directional antenna with simple construction and a small 12mm by 12mm right isosceles triangle footprint. Antenna 274933 has the best minimum VSWR of 1.05 at 5.18 GHz, but this quickly rises above 3.0 after 5.24 GHz. Antenna 624971 (population 10000) looks good in the table, but its many lobed gain pattern makes it less desirable.

Table 2. Summary of results, with whip, Yaego, Centurian, and HyperLink antennas for reference

Random Seed	Population Size	Fitness	VSWR	Max Gain (dBi)	Comments
whip	for reference, not from GP		1.3	2	omni directional
Yaego	for reference, not from GP		2.5*	4	max VSWR
Centurion	for reference, not from GP		2.0*	9	max VSWR
HG5308P	for reference, not from GP		1.5**	8	avg. VSWR
HG5311P	for reference, not from GP		1.5**	11	60/30 beam width
5281999	1000	-13.86	1.46	14	invalid model, duplicate segments
8281999	1000	-3.23	2.52	5.75	
5647348	1000	-4.71	2.98	7.69	
392561	1000	-1.63	1.57	1.79	good omni pattern, trivial construction
624971	5000	-7.1	1.73	7.27	
624971	10000	-6.23	1.36	6.37	odd gain pattern
274933	6500	-6.57	1.362	6.71	narrow VSWR band, min VSWR 1.05
212519772	6500	-4.4	1.74	4.61	
404803468	10000	-7.25	1.68	7.42	best antenna

Though not an explicit goal, the fitness measure over optimizes high gain. Runs without the geometry checker, such as 5281999, produced extraordinarily high (and invalid) gain numbers, and when these runs were allowed to go for many generations the gain numbers kept getting higher while VSWR stayed just under 2.0. Increasing the weight of the VSWR term, or framing the fitness measure as a single objective problem with constraints could remedy this.

Culling out all antennas with geometry errors may be too drastic. While easy to implement, it seems to lead to a loss of too much genetic material. This hypothesis is validated by the success of runs with a greater initial population, and suggests that a less drastic strategy for handling geometry errors should be employed.

7 Conclusion

We achieved better than expected results which validate the methodology and create the opportunity for more investigation. The genetic programming architecture is proven and simple. Discretizing the problem space reduces complexity without sacrificing the quality of results, though in the end does not obviate all geometry errors. The fitness measure is successful in selecting low VSWR, high gain individuals, but can be improved.

Though antenna 404803468 is a competitive solution and could be used, its size is still prohibitive for the average 802.11a user. This exercise really serves as an important proof of concept. We chose the problem dimensions with manufacturability in mind; AGW#19 bent at 45 degree angles with sections no smaller than 4mm can be built with a careful hand, and then tested to verify model accuracy. If after further work, simulations and empirical data correlate well we will try to shrink the wire size down so the entire grid would be 3 mm square, or apply this technique to printed circuit board antennas.

The market for external antennas is small, but a printed circuit board antenna or a wire antenna less than 3mm square (and encased in resin) could then be used in an embedded wireless device, or placed into the top of the housing of a notebook computer.

Acknowledgements. Thanks to Dr. John Koza for his interest and assistance with this endeavor, Brad Marsh from Texas Instruments for making time in a demanding schedule for the initial project, and Bryan Bell at SiRiFIC wireless for its continued support.

References

1. Altshuler; Edward E. and Linden; Derek S. 1999. Design of wire antennas using genetic algorithms. In Rahmat-Samii, Yahya and Michielssen, Eric (editors). Electromagnetic Optimization by Genetic Algorithms. New York, NY: John Wiley & Sons. Pages 211 - 248.
2. Comisky, William, Yu, Jessen, and Koza, John. 2000. Automatic synthesis of a wire antenna using genetic programming. Late Breaking Papers at the 2000 Genetic and Evolutionary Computation Conference, Las Vegas, Nevada. Pages 179 - 186.
3. Lohn, Jason D; Linden, Derek S; Hornby S, Gregory; Kraus, William.F; Rodriguez-Arroyo, Adaan. Evolutionary Design of an X-Band Antenna for NASA's Space Technology 5 Mission. 2003 NASA/DoD Conference on Evolvable Hardware (EH'03) July 09 - 11, 2003 Chicago, Illinois. Page 155
4. Abelson,Harold and diSessa,Andrea.1980. Turtle Geometry. Cambridge,MA: The MIT Press.

5. Koza, John R. 1992. Genetic Programming: On the Programming of Computers by Means of Natural Selection.Cambridge, MA: The MIT Press. Chapter 1, Chapter 12 Pages 327-345
6. Std. IEEE 802.11b-1999, Wireless LAN Medium Access Control and Physical Layer Specifications:Higher Speed Physical Layer Ext. in the 2.4 GHz band.September 1999.
7. Std 802.11a-1999. Wireless LAN Medium Access Control and Physical Layer specifications High-speed Physical Layer in the 5 GHz Band. September 1999.
8. Burke, G. J.; Poggio, A. J. Numerical Electromagnetics Code (NEC) Method Of Moments Part II Program Description Code. UCRL- Livermore, CA: Lawrence Livermore National Laboratory. 1981
9. Sinnema, William; Electronic Transmission Technology, Lines, waves, and antennas. Pearson Education POD, 17 February, 1998. Pages 76-78, 275
10. Koza, John R. 1994a. Genetic Programming II: Automatic Discovery of Reusable programs. Cambridge, MA: MIT Press.
11. Multilayer Ceramic Antenna for Bluetooth/Wlan IEEE 802.11b & WLAN IEEE 802.11a (2.45/5.2GHz) (Surface Mounted Ceramic Dual Band Antenna) Product Specification. Taiwan: Phycomp Taiwan LTD. September, 2001
12. http://www.murata.com/nproduct/6l04.html
13. http://www.centurion.com/antennaProd/whisper5ghz.asp
14. http://www.hyperlinktech.com/web/hg5308_11p.php