WIREGRID: A NEC2 pre-processor

C. F. du Toit, D.B.Davidson
Dept. of Electrical and Electronic Engineering
University of Stellenbosch
Stellenbosch 7600
South Africa
e-mail: davidson@firga.sun.ac.za

Abstract

WIREGRID is a pre-processor for the widely used public domain method of moments computer code NEC2. WIREGRID was developed for modelling metallic structures using a mesh of wires. In particular, it has powerful facilities for controlling the density of the mesh, permitting the same basic model to be used for different frequencies, with the code automatically generating a suitable mesh for a particular frequency. WIREGRID is not limited to generating wire meshes: it can also be usefully applied to true thin-wire modelling, for instance for Yagi-Uda and log-periodic antennas. In this paper, the basic philosophy of the code is described, and the code capabilities are detailed. Sufficient detail is presented so that the paper can serve as a reference for code users. The meshing algorithm is briefly described. An example is shown of the automatic mesh generation capabilities applied to an automobile. WIREGRID is available in the public domain via anonymous ftp; details of the ftp sites and hardware requirements are given in the appendices.

1 Introduction

NEC2 has become the *de-facto* standard thin-wire method of moments code that other codes are evaluated against, due not least to:

- Its powerful facilities for general thin-wire modelling. These include not only the conventional free-space treatment but also a ground treatment (both an approximate reflection coefficient approach and an "exact" — in a numerical sense — Sommerfeld treatment). NEC2 can also exploit symmetry, and handle non-radiating networks and lumped loads. This brief review highlights only some of its capabilities.
- The extensive documentation available.
- The widespread availability of the code.

However, anyone who has used NEC2 is well aware that the code, powerful as it is, can be frustrating and time-consuming to use, since the model input is described by a list of coordinate points. It is extremely easy to make an error in such an input file, and very difficult to pick it up without some graphical aid. Furthermore, when such an error is present, the code will often run and produce results that are plausible but incorrect. Hence a number of pre-processors have been written specifically for NEC2 to simplify model generation, one of which is the subject of this paper, namely WIREGRID, which we are making available in the public domain. We assume a basic familiarity with NEC2 in this paper, since WIREGRID was written specifically for NEC2 model generation.

DIDEC was one of the first pre-processors for NEC2 [1]. However, IGUANA (available from ACES as part of the NEEDS package) was one of the first *widely available* such pre-processors; it has been generally acknowledged as a useful tool but with a number of deficiencies. Work has also been reported by other researchers on graphical tools for preparing NEC2 data files; this journal has published at least two such papers [2, 3] and the 1993 ACES conference saw a number of codes of varying quality and cost being demonstrated. As this paper went to print, another South Africa group working at the University of the Witwatersrand announced a commercial software package called SIG (Structure Interpolation and Gridding), which offers facilities similar to or exceeding those in WIREGRID [4].

Our pre-processor WIREGRID was originally developed for a specific application of NEC2, namely modelling metallic structures using a mesh of wires, from whence the name. In particular, it has very powerful facilities for controlling the *density* of the mesh, permitting the same basic model to be used for different frequencies, with the code automatically generating a suitable mesh for a particular frequency (according to the $\lambda/10$ rule, for instance). Anyone who has had to generate a wire grid manually or even using a pre-processor such as IGUANA will know that this is an arduous task:

specifying the end point coordinates of each wire segment of the mesh (of even relatively simple structures) in the format required by NEC2 is an error-prone and exhausting task when done once; and if total number of segments has to be changed to produce a finer or coarser discretization, the structure must almost be re-modelled from scratch for each change. WIREGRID was developed to simplify these tasks considerably. The initial number of coordinates that must be typed in by hand is typically reduced by at least one order of magnitude for coarse meshes, and two or even three orders for fine meshes. Since many structures can be modelled by a number of flat or almost flat polygonal surfaces, it was decided that a pre-processor was needed which would take as input just the vertices of these polygons, and which would then generate the wiregrid discretization automatically. Changing the segment density then only requires a few keystrokes. Once the user is satisfied with the model, the code automatically generates the NEC2 input file with the geometry cards and a limited number of control cards.

WIREGRID is not limited to generating wire meshes: it can be usefully applied to true thin-wire modelling, for instance for Yagi-Uda and log-periodic antennas, and was most recently used for a project involving frequency selective surface modelling. However, most of its features were developed with wiregrid modelling in mind, since at the time the first version was developed (1990) we were involved with the electromagnetic characterization of HF and VHF whip antennas mounted on ground vehicles and we were encountering serious problems preparing the large data sets that this application required. Closely related research on developing a parallel version of NEC2 was also in progress at that time [5, 6]; due to the increased problem size that this parallel version of NEC2 could handle, WIREGRID also proved very useful with this work.

The code was originally developed by the first author in close liaison with the second author, and has subsequently been extensively used for both vehicle EMC modelling under contract (as outlined above) and teaching (at undergraduate and postgraduate level, as well as for short courses on NEC2). The idea underlying the code is similar to that reported by Najm [3], but the implementation is rather different. WIREGRID is highly graphically orientated, whereas Najm's code is not. It also automates a number of functions that Najm's code requires manual intervention to implement; see for example the discussion regarding the motor vehicle [3, Fig. 5]. Despite the conceptual similarities of the two codes, WIREGRID was developed independently and at about the same time [7].

As with most complex computational electromagnetic codes, while a pre-processor such as WIREGRID greatly simplifies the task of the user, a knowledge of the input requirements, and the basic functioning, of the code is still highly advisable. The user may need to append additional control statements to the input file and may even want to change some of the geometry data afterwards.

2 Using the program

2.1 Structure Modelling

The tasks required by the user are the following: The structure must first be divided into fairly flat polygonal surfaces, which are called major elements (see Fig. 1). WIREGRID can handle up to 160 such major elements. Each major element does not need to lie *completely* in a 2D plane, but severely "twisted" elements are really beyond the capability of the code. The frequent visual feedback available from the program will permit the user to identify unsuitable elements and rectify them. As mentioned in the Introduction, the code was developed for structures consisting mainly of flat surfaces; it is not really suited modelling curved surfaces, such as an aircraft fuselage. The coordinates of each major element's vertices, called major nodes, must be determined next (see Fig. 1). The simplest major element is defined by two major nodes, which would model a single straight wire. The most complex major element may be defined by a maximum of 10 major nodes.

When generating a mesh, it is important to ensure that meshes in adjacent elements are properly electrically connected across the boundary. (NEC2 users will be aware of a particularly insidious fault caused by two wires whose end-points that are closely located and should be physically connected, but are treated by the code as unconnected.) We will use the term conformality in this paper to describe a properly connected mesh. WIREGRID will ensure conformality provided that major nodes of adjacent major elements coincide on the boundary that the elements share. These requirements mean that when vertices on the boundary between two adjacent polygons are not coincident, extra major nodes for one or both of them must be defined. These extra nodes must coincide with the unmatched vertices (see Fig. 2). The same technique can be used for ensuring that a whip antenna is properly grounded to the vehicle frame. Superfluous nodes (for example, a node in the middle of an element side when there are no conformality problems) do not cause any problem within the program (providing of course that all elements sharing that side contain the extra node). However, the wire mesh generated may be distorted by their presence, and it is recommended that such unnecessary major nodes be avoided.

An additional geometric requirement for the major elements is that they must be convex, i.e. a straight line

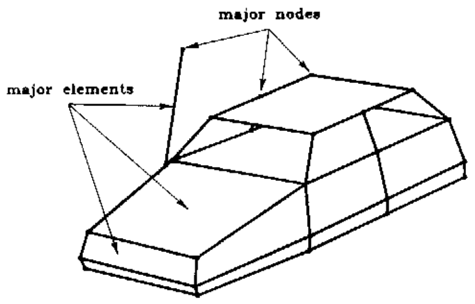


Figure 1: Arbitrary structures can be built up with flat polygonal surfaces. The example shows a simplified model of a generic "hatchback" motor car.

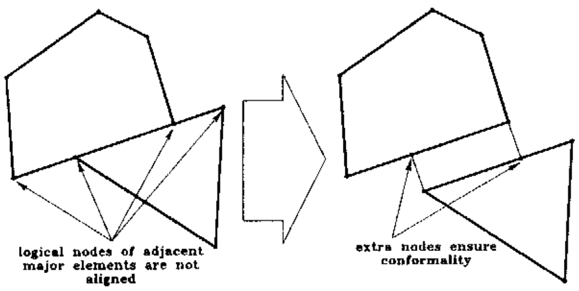

logical nodes of adjacent major elements are not aligned
extra nodes ensure conformality

Figure 2: To ensure that the mesh within each major element is electrically continuous across the common side, extra major nodes must sometimes be added as shown here.

joining any two points in the polygon must still be enclosed by the polygon boundaries. The reason for this requirement is that it greatly simplifies the task of the mesh generator in WIREGRID. It can be adhered to easily, since any non-convex polygon can be divided into two or more convex parts.

A number tag starting from 0 is assigned by WIREGRID to each major element. It is often helpful to keep a rough sketch of the structure, indicating all the major elements with their number tags. In the NEC2 input file that WIREGRID generates, all wires belonging to a certain major element will be tagged with this number. When certain wires are to be loaded, excited or even altered by hand, this will help to locate them.

2.2 Discretization

The *nominal length* ℓ_n of the wire segments in the mesh may be changed at any time during the structure modelling process. The new discretization is performed immediately. The nominal segment length is a key concept in the code: each straight wire is divided into p segments, defined by

$$p = \text{int}\left(\frac{L}{\ell_n} + 0.99999\right), \tag{1}$$

where L is the length of the wire section and $\text{int}(x)$ is the integer operator, taking the integer part of x.

All straight wires belonging to a major element defined by more than two major nodes are divided into segments of equal length. Following the "same surface rule" [8], these wires are modelled with a radius of $\frac{\ell_n}{2\pi}$. [1]

When the wire is a major element defined by two nodes, the "same surface rule" is irrelevant, and the user must specify radii for the two end points, which is independent of the discretization. (The code still offers a radius of $\frac{\ell_n}{2\pi}$ as the default, although this will of course almost certainly be incorrect in this case). The number of segment divisions on the wire is still defined by (1). In addition to simple wires of uniform radius (corresponding to a `GW` NEC2 card), WIREGRID can also handle tapered wires, generating the necessary `GW` and `GC` cards.

2.3 A brief overview of the meshing algorithm

A detailed description of the meshing algorithm is beyond the scope of this paper. The following description

[1] One reviewer expressed surprise at this radius, but Ludwig's paper does indeed verify this rule. This rule actually refers to each polarization, so as Ludwig comments, "twice surface area rule" might be a better name. Trueman and Kubina give the same formula for wire radius [2, p.21]. Wires of this radius are on the verge of being too thick for the thin-wire treatment in NEC2 (the extended kernel option does not help with a mesh, since it cannot be used for segments with a complex junction), but experience has produced generally acceptable results following this rule.

is intended to give only a flavour of the meshing algorithm. Figure 3 shows the significant features of the meshing process, and will be referred to during this discussion. Five major nodes are shown by •; only four are actually required for this major element, but the fifth would be required if another major element were to have a major node at this point. (See for instance Figure 2 for such an example.) At the risk of tedium, we reiterate that the coordinates of these major nodes are all that the user need enter.

- The sides of the polygonal are divided up, as close to the specified nominal segment length as possible. These generate minor nodes along the sides. The minor nodes are represented by ∘ in Figure 3. These minor nodes are then used to the basis for the mesh as follows:
- The vertex with included angle closest to 90° is then found.
- The sides of the polygon are then numbered counter-clockwise. Assume that the polygon is close to rectangular; this will then generate sides 1 to 4.
- Starting at the first point on side 1 where a minor node should be located, the code constructs vertical cross-wires from side 1 to side 3, endeavouring to keep the cross-wires as close as possible to parallel to each other. (These wires are coloured red on the screen).
- Once these cross-wires have been constructed, the cross-wires connecting sides 2 and 4 are constructed in a similar fashion. (These are coloured blue). The mesh thus created is shown in Figure 3. The mesh is represented by the dashed lines in the figure. In general, the horizontal cross-wires are individual wires running between each vertical cross-wire, with potentially slightly different lengths and directions. However, due to the geometrical simplicity of this example, all cross-wires in Figure 3 are uniform.
- The code has a number of features to try to generate a "good" mesh for irregularly shaped polygons; for instance, a point on a side may sometimes be skipped if a "better" mesh can be constructed by using the next point on a side as the end-point for a cross-wire. ("Good" and "better" refer to the angles that the cross-wires make with the element sides.) The intersections of the "blue" and "red" wires may also be perturbed slightly with the aim of making the included angles between the wires as close as possible to 90°. (Neither of these mesh improvement stages are required in the mesh of Figure 3.)
- The code tries to avoid creating long, thin triangular meshes. (The triangular examples shown by Najm [3] are not good meshes for NEC2: with a significant wire radius they are very likely to generate match points inside another wire's volume; the NEC2 user guide specifically cautions against this [9, p.3–6]). Users should try to avoid entering major elements with such geometries. Automatically detecting such violations of the NEC2 rules and guidelines is a feature we would like to add but have not yet; we will comment on this later.

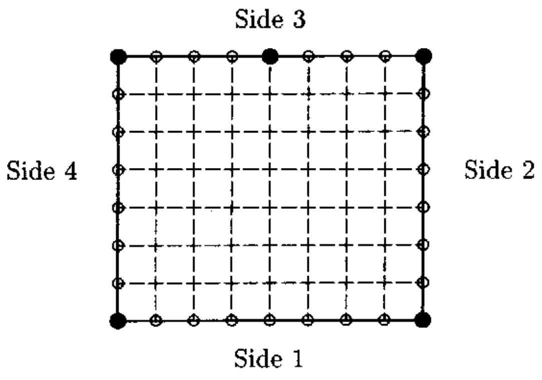


Figure 3: A rectangular major element illustrating the meshing algorithm outlined in Section 2.3. See text for discussion.

2.4 Generation of the NEC2 input file

WIREGRID supports two data formats for the NEC2 input file:

- Fixed length data fields. All numbers are separated by spaces, where all integers occupy fields three to five characters long, and floating point numbers occupy 10 character long fields. This is the format documented in the user guide [9].
- Variable length data fields. All numbers are separated by commas. A number of newer versions of NEC2 also support this format.

When satisfied with the model, the first part of the NEC2 input file containing all the geometric data may then be generated on command.

WIREGRID also provides for the generation of control statements which allow maximum coupling computation between voltage sources, and radiation pattern generation at specified frequencies. Only wire segments belonging to major elements defined by *two* nodes may be excited for this purpose. (Of course, the experienced user can edit the NEC2 data file created by WIREGRID to define more complex antennas). Furthermore, the radiation pattern cards generated are only for cross sections defined by angles of constant azimuth or elevation angles of the full three dimensional patterns. Since it is fairly easy to enter the control statements into the NEC2 input file by hand, only these very common type of requests are catered for by WIREGRID.

Any additional command lines or alterations to the geometry data may be added afterwards, with the user's editor of choice. For this purpose, the following information on the structure of the geometry data section is important:

- Each tag number in the file refers to the number of the major element wherein the wire segment is located. This means that wire segments belonging to major elements defined by more than two major nodes do not have unique tag numbers, and cannot be referred to by the control statements at the end of the file, unless the tag number is changed by the user afterwards. This is also the reason why WIREGRID only allows excitation of segments in two-node major elements in its control statement generation options.

- Geometry data for the polygonal boundaries (yellow[2]) between the major elements are entered first into the file, starting with the boundaries of major element number 0, and then the two node elements are added.

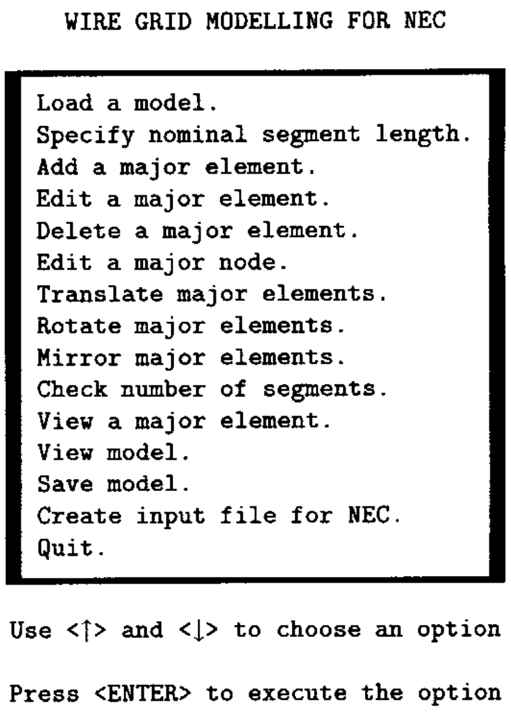
- This is followed by data statements describing the internal meshes (where relevant). These statements are divided into sets for each polygonal major element, starting with the element with the lowest tag number. In addition, the internal mesh data statements for each polygonal major element are subdivided into a subset of statements describing long, straight, segmented wire sections (red), and a subset describing short, single segment wire sections (blue).

2.5 The user interface

Fig. 4 shows the main menu screen of WIREGRID. An option is selected by moving the cursor to the required one and hitting return. Should the wrong option be selected accidentally, hitting escape at any time while WIREGRID is prompting for input information returns the user to the main menu, without any action being taken[3].

[2]The colours mentioned are used in the graphical display of the model (see Section 2.3).

[3]Editing major element coordinates, or selecting wire segments for excitation, are exceptions to this rule. In these cases hitting escape ends the editing or selecting session. An element thus incorrectly created is removed by deleting the major element.



```
          WIRE GRID MODELLING FOR NEC

  Load a model.
  Specify nominal segment length.
  Add a major element.
  Edit a major element.
  Delete a major element.
  Edit a major node.
  Translate major elements.
  Rotate major elements.
  Mirror major elements.
  Check number of segments.
  View a major element.
  View model.
  Save model.
  Create input file for NEC.
  Quit.

Use <↑> and <↓> to choose an option

Press <ENTER> to execute the option
```

Figure 4: WIREGRID's main menu screen

2.5.1 Creating a new model

When a new model is started, the nominal segment length must be specified before any major elements can be entered. Experience has taught many new users have problems with the code at this initial stage, not realizing that a nominal segment length is *required.* Any reasonable value suffices; as already discussed the nominal segment length can be changes at any time during the modelling process. The first major element can then be entered. If a model is already loaded, and the user wants to start a new model, it is easiest to quit and run the program again.

2.5.2 Deleting major elements

A major element is deleted by selecting the "Delete a major element" option. It is important to note that all elements with tag numbers higher than the one deleted will be assigned a new number, one less than before. When more than one element must be deleted, it is therefore best to start with the highest-numbered element and work downwards. This eliminates the need to continuously recalculate the tag number of the elements still to be deleted.

2.5.3 Adding, editing and saving models

An existing model may be loaded from disk, together with the last discretization used. The user may therefore start to edit major elements or add new ones immediately after loading.

When a new major element is defined, WIREGRID prompts for the element's number tag and the number of nodes defining the element. The default value of the latter is two, and replacing it by larger numbers (up to 10), will result in more complex elements. WIREGRID subsequently presents the user with a table of major node coordinates, with default values corresponding to the respective nodes of the previous major element[4]. The cursor is moved around in the table with the left and right tab keys, and with the up and down arrow keys, to edit the various coordinates. Coordinate editing is followed by specification of the end point radii for two-node major elements. (The default radius of $\ell_n/2\pi$ has already been discussed in Section 2.2).

The default values for the latter is $\ell_n/2\pi$, the value used for the polygonal major element wire sections at that stage. (This is of course probably wrong, since two-node element frequently represent wire antennas, but it is trivial to enter the correct radius).

Editing an existing major element is the same as defining a new one, except that the default values for the number of nodes, coordinates and end point radii, are the actual values specified previously.

Models can be combined by loading an existing model from disc at any time and merging it with the current one in memory, helping the user to model the structure in an organized and structured fashion. An example would be the hull and turret of a ground vehicle, which are conveniently modelled separately and then combined. When the models are merged, WIREGRID ensures that elements from the different models with common edges are correctly picked up, ensuring that the mesh is properly conformal.

A range of subsequent major elements may be spatially translated by adding a vector to all coordinates involved. Alternatively, they can be rotated around an axis (specified by the user as a line through two three-dimensional Cartesian points) by a certain angle, or mirrored in a plane (specified by the user as the spherical coordinate vector from the origin to the nearest point on the plane surface). In the process the elements can be effectively moved to these new positions, or new elements can be created in this way. This helps the user to build up structures with symmetrical properties fairly quickly. Note that these "Translate", "Mirror" and "Rotate" options do not generate associated structure geometry lines (`GM`, `GX`, `GR` - "cards" respectively) [9, p. 16] in the NEC2 input file, but physically generate new or altered major elements.

The model can be saved at any stage by specifying an appropriate file name. It is recommended that this be done often, since a major element may be accidentally deleted, or an unforeseen run-time error may occur. The latter is unfortunately not infrequent, since it is very difficult to foresee all errors that may occur, in particular with incorrect user input. Provided that the model has been recently saved, it is simple to reload WIREGRID, load the model and then continue building the model.

2.5.4 Entering numbers

When typing in a number, or editing one, the `Backspace`, `Delete` and the `Left` and `Right` arrow keys can be used in the usual manner. The default editing mode is always the "Insert" mode, but it can be toggled to the "Overwrite" mode with the Insert key. `Ctrl + Left` or `Right` arrows will move the cursor to the end or beginning of the number being edited.

2.5.5 Inspecting the model

The entire model can be inspected at any time by selecting the "View model"-option, or a single major element may be inspected by selecting the "View a major element"-option instead.

The boundaries of the major elements are displayed in yellow, while wires generated by the discretizer are displayed in red and blue. These have been discussed in §2.3.

The view of the model can be changed by zooming in or out with the + and − keys, by rotating it with the arrow keys, and by translating it sideways or up and down with the `L`, `R`, `U`, and `D` keys.

Any major element can be highlighted by hitting `S` and specifying the major element number. When viewing a single major element, this can be used to add other major elements to the picture. This feature is the only one that does not work with a monochrome graphics adapter, since the highlighted (white) colour cannot be distinguished from the other colours.

2.5.6 Excitation of wire segments

In the final generation of the NEC2 input file, the user may specify the excitation of some wire segments for maximum coupling computation and radiation pattern generation (see Section 2.4).

For maximum coupling calculations, up to five segments may be excited, and for an antenna array pattern calculation, up to ten segments may be excited.

[4]This eliminates to a certain extent the need to retype node coordinates common to the previous element.

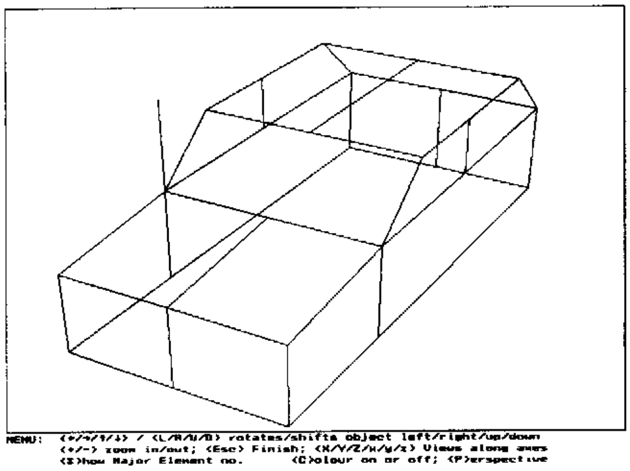

MENU: <←/→/↑/↓> / <L/R/U/D> rotates/shifts object left/right/up/down
<+/-> zoom in/out; <Esc> Finish; <X/Y/Z/x/y/z> Views along axes
<S>how Major Element no. <C>olour on or off; <P>erspective

Figure 5: A passenger motor car showing only the major elements. There are 41 segments in this model. The model was generated using symmetry, hence the wires on the centreline.

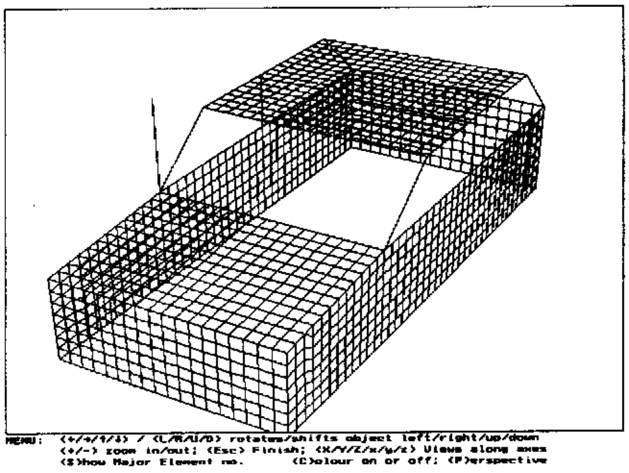

MENU: <←/→/↑/↓> / <L/R/U/D> rotates/shifts object left/right/up/down
<+/-> zoom in/out; <Esc> Finish; <X/Y/Z/x/y/z> Views along axes
<S>how Major Element no. <C>olour on or off; <P>erspective

Figure 6: The same passenger motor car with a nominal segment length of $0.1m$. (Using the $\lambda/10$ rule, this model would be valid up to around 300 MHz). The number of segments is now 3008. It must be emphasized that only one key stroke was required to generate this mesh from the model shown in Figure 5. Note that the code does not do hidden line removal. (A very powerful computer would be required to *run* this model: the discretization shown here is to emphasize the capabilities of WIREGRID).

A list of selected segments (specified by their major element tag number, and the segment number[5]) is shown near the top of the computer screen. Below that, information on all two-node major elements is displayed for one element at a time, while the user is "paging through" the list with the left and right arrow keys. In order to select a wire segment, the user "pages" to the particular two-node major element, and increments or decrements the segment number with the up and down arrow keys. The insert key is finally used to select a segment for excitation, whereupon WIREGRID will prompt for the excitation voltage, and the segment will be added to the list of excited segments. The delete key is used to remove segments from this list.

We have found from experience that users often have trouble using this particular feature of WIREGRID, partly because their is no visual feedback to confirm the location of the sources. Since it is relatively easy to manually append the necessary source commands to the NEC2 input deck created by WIREGRID, many users elect to do this.

2.6 An example

In Figures 5 and 6 we show the WIREGRID mesh for a motor vehicle, very similar, although not identical, to the generic hatchback shown in Figure 1.

[5]Segment numbers start at 1, denoting the segment joined to node 0.

3 Conclusions

We have found that WIREGRID has greatly boosted our productivity for generating NEC2 input decks for simulations involving wire mesh modelling. WIREGRID has a number of limitations, many of which we have outlined in this paper, but there are some other *fundamental* limitations that potential users should be aware of. WIREGRID does *not* check the validity of the structures that it generates with respect to all the NEC2 model generation rules — admirably summarized by Trueman and Kubina [2]. Particularly when the structure involves thin triangular elements, it is quite possible that the WIREGRID mesh may violate some of the validity conditions. Of course, its output could easily be run through such a validity checker, but at the time of writing, Trueman and Kubina's code, CHECK, is still not generally available due to export restriction problems. We would like to extend WIREGRID to incorporate at least some of these validity checks. We would also like to add a facility to read in and display NEC2 data decks in WIREGRID; it does not presently have such a facility. However, funding for this work is very restricted at present and neither facility is likely to implemented in the near future. Finally, while we have found the user interface to be very functional, it does not have the latest Windows-style icons, dialogue boxes, hot-keys etc.

4 Acknowledgements

The authors would like to thank the many users at Stellenbosch and in South Africa who have helped to test the code for their comments and suggestions. D.H.Malan implemented the upgrades described in the appendix "Recent Improvements".

References

[1] S. J. Kubina and C. Larose, "A NEC topside antenna case study with Didec and Spectrum: model generation and current display codes," *ACES Newsletter*, vol. 1, no. 2, pp. 47–52, 1986.

[2] C. W. Trueman and S. J. Kubina, "Verifying wire-grid model integrity with program 'CHECK'," *ACES Journal*, vol. 5, pp. 17–42, Winter 1990.

[3] R. K. Najm, "Simplified 3-D mesh generator," *ACES Journal*, vol. 6, pp. 86–98, Winter 1991.

[4] A. P. C. Fourie, D. C. Nitch, and O. Givati, "A complex-body structure interpolation and gridding program (SIG) for NEC," *IEEE Antennas Propagat. Magazine*, vol. 36, pp. 85–89, June 1994.

[5] D. B. Davidson, "PARNEC," *IEEE Trans. Antennas Propagat. Magazine*, vol. 34, pp. 39–40, April 1992. In "PCs for AP and other EM Reflections" column, edited by E. K. Miller.

[6] D. B. Davidson, "Parallel processing revisited: a second tutorial," *IEEE Antennas Propagat. Magazine*, vol. 34, pp. 9–21, October 1992.

[7] D. B. Davidson and C. F. du Toit, "Recent progress with PARNEC and the pre-processor WIREGRID," in *Proceedings of the 1991 IEEE/SAIEE Joint AP-MTT Symposium*, pp. 242–249, August 1991. Held in Fourways, Johannesburg, South Africa.

[8] A. C. Ludwig, "Wire grid modelling of surfaces," *IEEE Trans. Antennas Propagat.*, vol. AP-35, pp. 1045–1048, September 1987.

[9] G. J. Burke and A. J. Poggio, "Numerical Electromagnetics Code (NEC) - Method of Moments; Part III: User's Guide." January 1981.

Appendices

A Obtaining and Installing the Code

The code is available on the NEC Archives ftp site. The ftp site name is `ftp.netcom.com`, and the code is in `pub/rander/NEC` directory as `wiregrid.zip`. The code was compressed using the widely used DOS shareware utility `PKZIP`; use `PKUNZIP` to decompress. Details on downloading codes from anonymous ftp sites may be found in any of the many guides to the Internet now appearing. To summarize, proceed as follows:

- Enter `ftp ftp.netcom.com`
- The system will respond with `Name:` (or similar message); reply with `anonymous`.
- The system will then prompt for a password; it is customary to enter your Internet (e-mail) address.
- Enter `cd /pub/rander/NEC`.
- then `binary` (this *is* actually the default for this site).
- and finally `get wiregrid.zip`.
- To exit the ftp session, `quit`.

There may be an appreciable delay (possibly a number of seconds) between entering a command and obtaining a response, especially if you are accessing the site from outside the USA. South African users should note that `wiregrid.zip` is mirrored on our local Stellenbosch ftp site, viz. `ftp.sun.ac.za`, and should download from this site in preference to `ftp.netcom.com`.

If `PKUNZIP` cannot unzip the file, check that you have a sufficiently recent version (the index entry for WIREGRID details the version used); if this is not the problem, you may have done a ftp transfer in ASCII mode en route to your PC. Your local Internet guru will be able to advise!

For installation, all files on the distribution disk should be copied to a directory of the user's choice. When `wiregrid.exe` is run, it will prompt for the directories where the wire grid models and the NEC2 input files are to be stored respectively. Several example files are included with the `zip` file. The user may change these directories during a session, and these changes then become the defaults for the next session. Whenever a directory which does not exist is specified, it will be created on request.

B Hardware requirements

WIREGRID.EXE runs under DOS on an IBM compatible PC. The program is written in TurboPascal, and makes extensive use of the TurboPascal graphics calls. As a result, it will unfortunately *not* be a simple matter to port to another hardware platform using a different operating system (such as an Apple Macintosh). Although a 8087/80287/80387 mathematical co-processor would be advantageous, it is not a prerequisite. (The code automatically detects the presence of a co-processor). A colour graphics adapter and a colour display monitor (or a monochrome display which can differentiate between colours using various shades of grey) are required to fully exploit the graphical features of WIRE-GRID, but almost all of these will still work satisfactorily with a monochrome graphics adapter. WIREGRID supports the CGA, EGA, IBM8514, Hercules, ATT400, VGA and the PC3270 graphics adapters.

C Recent Improvements

This paper describes WIREGRID Version 3. The latest version, 4.00, incorporates some useful new features for viewing the model. These are :

- The model can now be plotted as if viewed along on of the three axes, X, Y or Z. (Actually, there are six, not three, options here, since x and X rotate the view by 180°.)
- The user can now toggle perspective on and off. The default is to show the model in perspective, but there are cases where it is more convenient to remove this facility, such as when the model is viewed along an axis.
- The display can now be switched to monochrome mode on colour screens. (There is no effect on monochrome screens). This is useful for doing screen-prints and screen-captures; for example, the screen-capture utility that was used to generate Figures 5 and 6 in this paper produces satisfactory images from a monochrome display, but not from a coloured one.

D Further Information

Users are welcome to contact the second author at the address given on this paper to report bugs or make suggestions, but as with most public domain software, we unfortunately cannot offer extensive user support. This paper serves as the user manual at present; no further documentation is available.

The standard software disclaimer applies: no warrantee is given or implied that all errors have been eliminated from the program WIREGRID, and the authors and the University of Stellenbosch accept no responsibility for any losses, damages etc. resulting from the use of the program.

The authors

Cornelis (Nelis) F. du Toit was born in South Africa on May 3, 1962. He received the B.Eng., M.Eng. and Ph.D. from the University of Stellenbosch in 1984, 1986 and 1992 respectively. His doctoral dissertation was entitled "The computation of electromagnetic fields from cylindrical near-field measurements at non-asymptotic distances from an antenna". He was a Research Assistant at the University from 1988–1992, and worked at Plessey-Tellumat in Cape Town during 1993 as a contractor. Nelis and his family moved to New Zealand in January 1994, and he is presently working for DelTec in Wellington. The work reported in this paper was performed whilst Nelis was at the University of Stellenbosch. His research interests include electromagnetic field theory, antenna design, near-field measurement techniques and numerical methods in antenna engineering. Nelis is a keen amateur astronomer and an accomplished pianist; he also enjoys listening to music. Nelis is married to Elize (a musician) and they have one child.

David B. Davidson (IEEE Member) was born in London, England on April 13, 1961. He grew up in South Africa. He received the B.Eng., B.Eng. (Honours) and M.Eng. degrees (all cum laude) from the University of Pretoria in 1982, 1983 and 1986 respectively, and in 1991 he received the Ph.D. degree from the University of Stellenbosch. His Ph.D. dissertation was entitled "Parallel algorithms for electromagnetic moment method formulations". David is presently an Associate Professor of Electrical and Electronic Engineering at the University of Stellenbosch, where he started teaching in 1988 following three years as a research engineer at the national research laboratories. His research interests centre around computational electromagnetics — theory, code development and applications — and include the application of parallel processing to this field; the work reported here was closely linked to his work on a parallel version of NEC2. He has worked with both transputer arrays for moment method codes and the CM-2 for FDTD codes. He recently spent six months as a visiting scholar at the University of Arizona, Tucson, AZ, working on applying computational electromagnetics methods to optical device modelling. In his spare time he climbs mountains, builds models, board-sails, plays tennis or squash or listens to music. David is unmarried.