

Visualization Tool for Ad Hoc Networks — ViTAN v1.1

F. Fitzek P. Seeling M. Reisslein* M. Zorzi

acticom GmbH – mobile networks
R & D Group
Germany
fitzek@acticom.de

Arizona State University
Department of Electrical Engineering
USA
{patrick.seeling,
reisslein}@asu.edu

Dipartimento di Ingegneria
Università di Ferrara
Italy
zorzi@ing.unife.it

February 2003

Technical Report acticom-03-001

The Visualization Tool for Ad Hoc Networks — ViTAN is a tool for visualizing the connectivities and link qualities (capacities) between the terminals in wireless ad hoc networks. The tool takes the location of the terminals (specified by (x, y) coordinates) and the link qualities between the terminals (specified by positive integers) as input. The tool produces a visualization of the graph of the terminals' connectivities in the **fig** format, which in turn can be converted to any common graphic format. ViTAN does not evaluate the connectivities and link qualities in ad hoc networks. Instead, ViTAN takes the link qualities obtained from other tools, simulations, or analytical evaluations as input and graphically visualizes these link qualities and the resulting connectivities in the network. ViTAN facilitates the visual study of complex ad hoc networks by depicting higher link qualities with thicker edges and in darker grey shades. In addition, ViTAN draws the edges at different depth levels of the **fig** format depending on the corresponding link quality. This feature enables the selective display and visual study of the connectivities provided by links with a specific quality range.

*The work of M. Reisslein is supported in part by the National Science Foundation through Grant No. Career ANI-0133252 and Grant No. ANI-0136774. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

Contents

1. Motivation	4
2. Installation	4
3. Usage of the ViTAN tool	6
4. Syntax of the Input File	7
5. ViTAN Example	8
6. Export to Other Graphic Formats	12
7. Examples	12
8. Future Work	16
9. Acknowledgement	16
A. Download Information	17
B. ViTAN Source Code	18
C. export Script	24

List of Figures

1. A simple chain configuration of six wireless terminals with graphical representation of the terminals.	9
2. A simple chain configuration of six wireless terminals.	10
3. A screenshot of the XFIG tool using the depth levels for the <i>chain</i> scenario.	11
4. A simple chain configuration of six wireless terminals with higher quality levels.	12
5. The <i>bridge</i> topology with 10 terminals.	13
6. The <i>circle</i> topology with 9 terminals.	13
7. The <i>Manhattan</i> topology.	14
8. 22 randomly distributed wireless terminals.	14
9. 44 randomly distributed wireless terminals.	15

List of Tables

1. Depth Level of the ViTAN XFIG output.	10
--	----

Listings

<code>./ViTANv11/directory</code>	4
<code>./ViTANv11/chain.input</code>	7
<code>./ViTANv11/chain.output</code>	8



./ViTANv11/vitan.pl	18
./ViTANv11/export	24

1. Motivation

In the study of ad hoc wireless networks researchers often face the problem of analyzing complex networks formed by nodes (terminals) that are placed in a wide variety of topologies. The placement of the nodes and their respective wireless transmission and reception capabilities typically rise to a complex network graph consisting of nodes interconnected by wireless links (edges) of heterogeneous quality (capacity). A visual representation of this network graph is oftentimes very helpful in evaluating the performance of a network protocol or mechanism for ad hoc networks. We were looking for a simple yet effective visualization tool for the network graph and came to realize that there was no such tool available.

We were thus motivated to develop a tool that takes the terminal (node) locations and the qualities (capacities) of the links (edges) interconnecting the nodes as input and generates a visual representation of the network graph. We chose the `fig` (often referred to as `xfig`) graphics format for our ViTAN tool, since this format is widely used.

The source code of our ViTAN tool is available for free from a number of web sites listed in the appendix of this document. We hope that others find ViTAN helpful and welcome your feedback. Also, we welcome additions and refinements to the tool.

2. Installation

After having downloaded the ViTAN tool (see Section A for details), the file has to be untared and unzipped:

```
tar zxvf ViTANv11.tar.gz
```

This process generated a subdirectory called ViTAN. Be sure that you have the following files in the subdirectory (dates and user names are different, file names are the same):

```

-rw-rw-r-- 1 pseeling pseeling 178653 Feb 19 18:14 100-.fig
-rw-rw-r-- 1 pseeling pseeling 177427 Feb 19 18:14 100.fig
-rw-rw-r-- 1 pseeling pseeling 22142 Feb 17 08:57 100-.input
-rw-rw-r-- 1 pseeling pseeling 22542 Feb 17 08:57 100.input
-rw-rw-r-- 1 pseeling pseeling 22948 Feb 17 08:57 100+.input
-rw-rw-r-- 1 pseeling pseeling 188527 Feb 19 18:14 100-.jpg
-rw-rw-r-- 1 pseeling pseeling 149096 Feb 19 18:14 100.jpg
-rw-rw-r-- 1 pseeling pseeling 144617 Feb 19 18:14 100-.pdf
-rw-rw-r-- 1 pseeling pseeling 144113 Feb 19 18:14 100.pdf
10 -rw-rw-r-- 1 pseeling pseeling 102705 Feb 19 18:14 100-.png
-rw-rw-r-- 1 pseeling pseeling 79881 Feb 19 18:14 100.png
-rw-rw-r-- 1 pseeling pseeling 39428 Feb 17 18:02 1.fig
-rw-rw-r-- 1 pseeling pseeling 187242 Feb 17 08:57 300.input
-rw-rw-r-- 1 pseeling pseeling 114376 Feb 19 18:14 30.fig
-rw-rw-r-- 1 pseeling pseeling 35007 Feb 19 18:14 30+.fig
-rw-rw-r-- 1 pseeling pseeling 2675 Feb 17 08:59 30.input
-rw-rw-r-- 1 pseeling pseeling 2498 Feb 17 08:59 30+.input
-rw-rw-r-- 1 pseeling pseeling 172180 Feb 19 18:14 30.jpg
-rw-rw-r-- 1 pseeling pseeling 100066 Feb 19 18:14 30+.jpg
20 -rw-rw-r-- 1 pseeling pseeling 99970 Feb 19 18:14 30.pdf
-rw-rw-r-- 1 pseeling pseeling 52700 Feb 19 18:14 30+.pdf

```

```

-rw-rw-r-- 1 pseeling pseeling 110093 Feb 19 18:14 30.png
-rw-rw-r-- 1 pseeling pseeling 41405 Feb 19 18:14 30+.png
-rw-rw-r-- 1 pseeling pseeling 75 Feb 17 08:56 3.input
-rw-rw-r-- 1 pseeling pseeling 15567 Feb 19 18:14 bridge.fig
-rw-rw-r-- 1 pseeling pseeling 482 Feb 17 16:10 bridge.input
-rw-rw-r-- 1 pseeling pseeling 96689 Feb 19 18:14 bridge.jpg
-rw-rw-r-- 1 pseeling pseeling 15440 Feb 19 18:14 bridge.pdf
-rw-rw-r-- 1 pseeling pseeling 29206 Feb 19 18:14 bridge.png
30 -rw-rw-r-- 1 pseeling pseeling 12007 Feb 19 18:13 chain2.fig
-rw-rw-r-- 1 pseeling pseeling 240 Feb 17 08:50 chain2.input
-rw-rw-r-- 1 pseeling pseeling 98195 Feb 19 18:13 chain2.jpg
-rw-rw-r-- 1 pseeling pseeling 12481 Feb 19 18:13 chain2.pdf
-rw-rw-r-- 1 pseeling pseeling 29874 Feb 19 18:13 chain2.png
-rw-rw-r-- 1 pseeling pseeling 11079 Feb 19 18:13 chain3.fig
-rw-rw-r-- 1 pseeling pseeling 251 Feb 17 08:54 chain3.input
-rw-rw-r-- 1 pseeling pseeling 109991 Feb 19 18:13 chain3.jpg
-rw-rw-r-- 1 pseeling pseeling 36400 Feb 19 18:13 chain3.pdf
-rw-rw-r-- 1 pseeling pseeling 30007 Feb 19 18:13 chain3.png
40 -rw-rw-r-- 1 pseeling pseeling 283 Feb 17 08:55 chain4.input
-rw-rw-r-- 1 pseeling pseeling 10849 Feb 19 18:13 chain5.fig
-rw-rw-r-- 1 pseeling pseeling 212 Feb 17 08:57 chain5.input
-rw-rw-r-- 1 pseeling pseeling 57201 Feb 19 18:13 chain5.jpg
-rw-rw-r-- 1 pseeling pseeling 36444 Feb 19 18:13 chain5.pdf
-rw-rw-r-- 1 pseeling pseeling 16486 Feb 19 18:13 chain5.png
-rw-rw-r-- 1 pseeling pseeling 11008 Feb 19 18:14 chain.fig
-rw-rw-r-- 1 pseeling pseeling 242 Feb 17 08:50 chain.input
-rw-rw-r-- 1 pseeling pseeling 97943 Feb 19 18:13 chain.jpg
-rw-rw-r-- 1 pseeling pseeling 1430 Feb 19 18:14 chain.output
50 -rw-rw-r-- 1 pseeling pseeling 12384 Feb 19 18:13 chain.pdf
-rw-rw-r-- 1 pseeling pseeling 29639 Feb 19 18:13 chain.png
-rw-rw-r-- 1 pseeling pseeling 15884 Feb 19 18:13 circle.fig
-rw-rw-r-- 1 pseeling pseeling 406 Feb 17 08:53 circle.input
-rw-rw-r-- 1 pseeling pseeling 104755 Feb 19 18:14 circle.jpg
-rw-rw-r-- 1 pseeling pseeling 14919 Feb 19 18:14 circle.pdf
-rw-rw-r-- 1 pseeling pseeling 31924 Feb 19 18:14 circle.png
-rw-rw-r-- 1 pseeling pseeling 0 Feb 19 18:14 directory
-rwxrwxr-x 1 pseeling pseeling 1998 Feb 17 08:50 export
-rw-rw-r-- 1 pseeling pseeling 32469 Feb 17 08:54 gif87a.txt
60 -rw-rw-r-- 1 pseeling pseeling 38136 Feb 19 18:14 manhattan.fig
-rw-rw-r-- 1 pseeling pseeling 1788 Feb 17 08:52 manhattan.input
-rw-rw-r-- 1 pseeling pseeling 183364 Feb 19 18:14 manhattan.jpg
-rw-rw-r-- 1 pseeling pseeling 26753 Feb 19 18:14 manhattan.pdf
-rw-rw-r-- 1 pseeling pseeling 55741 Feb 19 18:14 manhattan.png
-rw-rw-r-- 1 pseeling pseeling 47499 Feb 19 18:14 random2.fig
-rw-rw-r-- 1 pseeling pseeling 5019 Feb 17 08:52 random2.input
-rw-rw-r-- 1 pseeling pseeling 192656 Feb 19 18:14 random2.jpg
-rw-rw-r-- 1 pseeling pseeling 64460 Feb 19 18:14 random2.pdf
-rw-rw-r-- 1 pseeling pseeling 72075 Feb 19 18:14 random2.png
70 -rw-rw-r-- 1 pseeling pseeling 27146 Feb 19 18:14 random.fig
-rw-rw-r-- 1 pseeling pseeling 1540 Feb 17 08:50 random.input
-rw-rw-r-- 1 pseeling pseeling 151274 Feb 19 18:14 random.jpg
-rw-rw-r-- 1 pseeling pseeling 25704 Feb 19 18:14 random.pdf
-rw-rw-r-- 1 pseeling pseeling 51477 Feb 19 18:14 random.png

```

```
drwxrwxrwx    3 pseeling pseeling    4096 Feb 19 18:09 terminalpics
-rwxrwxr-x    1 pseeling pseeling    9303 Feb 19 12:55 vitan.pl
-rwxrwxr-x    1 pseeling pseeling     290 Feb 17 08:55 vitanview.pl
```

3. Usage of the ViTAN tool

The ViTAN tool is invoked from the command line. Several switches are used to enable or disable the features. In the following we explain the usage of the parameters and switches in detail.

Program Call:

```
vitan.pl inputfile outputfile [flag1] [flag2] [minX minY maxX maxY]
```

Parameters:

inputfile

The inputfile for the ViTAN tool specifying the ad hoc network as explained in Section 4.

outputfile

The outputfile is in the fig format [3] which can displayed with the XFIG [2] tool. Importantly, the suffix .fig is automatically added to the specified outputfile name to give outputfile.fig.

Optional Switches:

flag1

Flag1 has two valid values [0,1]. In case the flag is set to 0 the terminals are represented by a graphic. The graphic is selected with the `TERMINALREPRESENTATION` variable, see Section 4. If flag1 is set to 1 the terminals are represented by a red point, and the `TERMINALREPRESENTATION` variable is ignored. The default setting is 0.

flag2

Flag2 has two valid values [0,1]. In case the flag is set to 0 the figure is in the landscape format, otherwise the portrait format is used. The default setting is 0.

minX

minX specifies the minimum value of the x coordinate.

minY

minY specifies the minimum value of the y coordinate.

`maxX`

`maxX` specifies the maximum value of the x coordinate.

`maxY`

`maxY` specifies the maximum value of the y coordinate.

Note that `flag1` and `flag2` must be explicitly set when specifying the size of the coordinate system.

4. Syntax of the Input File

The input file for ViTAN contains all important information to generate and illustrate the ad hoc network. Assuming that we have J wireless terminals, each wireless terminal j , $j = 1, \dots, J$, is identified by its unique Cartesian coordinates (x_j, y_j) . The Cartesian coordinates might be positive or negative. Furthermore each wireless terminal has a name, a transmission range, and a representation. In addition, the link quality LQ to each of the other wireless terminals has to be specified. A valid row entry of the inputfile looks as follows:

```
XCOOR YCOOR NAME RANGE TERMINALREPRESENTATION LQ1 LQ2 ... LQJ
```

The unit of the length for `XCOOR`, `YCOOR`, and `RANGE` is one centimeter [cm]. In case `minX`, `minY`, `maxX` and `maxY` are not specified, ViTAN sets these values to the minimum/maximum x_j and y_j values plus some offset value. To give visually appealing outputs we introduced automatic scaling. The automatic scaling can be disabled, by setting the `$scale` value in the perl script to zero.

A valid entry for a name is any ASCII word. The `TERMINALREPRESENTATION` gives the name of the gif figure without the gif suffix, which represents the node. Note, the ViTAN package comes along with a set of gif figures. The figures are stored in the sub-directory named *terminalpics*. Any picture can be stored in the sub-directory and used with the ViTAN tool. An open problem is the automatic scaling of the gif figures. Because the `IMAGE` package does not run on each platform, the size is set statically. This might result in a disproportionate representation of the terminal. This issue will be fixed in one of the next versions. If `flag1` is set to 1 in the program call, then the `TERMINALREPRESENTATION` is ignored. However, some dummy file name must be specified.

For the link quality LQ any integer value between 1 and 255 is valid.

As an example we look at the `chain.input` file. The inputfile `chain.input` describes six terminals through their coordinates (first two columns), their names (Enno, Patrick, Gaby, Rolf, Gimpel, Stefan), the transmission range (same unit as the coordinates), and the related link quality to each other. A valid inputfile as given below has always J rows and $J + 5$ columns.

```
1000 1000 Enno 2237 soldier 0 8 0 0 0 0
3000 2000 Patrick 2237 soldier 2 0 8 0 0 0
5000 3000 Gaby 2237 smart 0 4 0 5 0 0
7000 4000 Rolf 2237 smart 0 0 5 0 4 0
9000 5000 Gimpel 2237 pda 0 0 0 8 0 2
11000 6000 Stefan 2237 pda 0 0 0 0 8 0
```

5. ViTAN Example

After installing the source simply evoke

```
./vitan.pl chain.input chain
```

The tool should generate the following output.

```
#####
# ViTAN - Visualisation tool for ad hoc networks #
#
# verison 1.1      -      acticom mobile networks #
#####
1000 1000 Enno 2237 soldier 0 8 0 0 0 0
3000 2000 Patrick 2237 soldier 2 0 8 0 0 0
5000 3000 Gaby 2237 smart 0 4 0 5 0 0
7000 4000 Rolf 2237 smart 0 0 5 0 4 0
10 9000 5000 Gimpel 2237 pda 0 0 0 8 0 2
11000 6000 Stefan 2237 pda 0 0 0 0 8 0
6 WTs are in the list and 6 WTs are depicted
Scaling up!
picture format: 1000 1500 23000 12500 (landscape)
no pictures used to represent wireless nodes
generating extra colors for XFIG tool
40 #FFFFFF
41 #EEEEEE
42 #CCCCCC
20 43 #AAAAAA
44 #888888
45 #666666
46 #454545
47 #232323
48 #010101
Enno is in the range of Patrick receiving with quality level 8
Patrick is in the range of Enno receiving with quality level 2
Patrick is in the range of Gaby receiving with quality level 8
Gaby is in the range of Patrick receiving with quality level 4
30 Gaby is in the range of Rolf receiving with quality level 5
Rolf is in the range of Gaby receiving with quality level 5
Rolf is in the range of Gimpel receiving with quality level 4
Gimpel is in the range of Rolf receiving with quality level 8
Gimpel is in the range of Stefan receiving with quality level 2
Stefan is in the range of Gimpel receiving with quality level 8
ViTAN has finished.
```

In the following we explain the output line by line:

line 1–5 Information Field

line 6–11 Values read from the inputfile

line 12 Information about the number of terminals read from the inputfile and the number of terminals that will be depicted in the `fig` file. Note, in case XMAX and YMAX are specified, terminals with a location outside of the bounding box are not depicted. In case XMAX and YMAX are not specified, all terminals are depicted.

line 13 Figure will be scaled

line 14 Information about the picture format

line 15 Indicates whether a graphic is used to represent the mobile. In case graphics are used, the file size of the figures will increase (This is the reason why we chose not to use the graphic version in this document.).

line 16-25 Grey scaled colors are defined. In this example eight different colors are generated. In case more than eight quality levels are used, more colors are generated.

line 26-35 Information about the connectivity and the link quality of all nodes.

line 36 Message that ViTAN has finished.

ViTAN will use the inputfile `chain.input` file and create a file in the XFIG format with the name `chain.fig`¹. The XFIG file should look exactly like Figure 1.

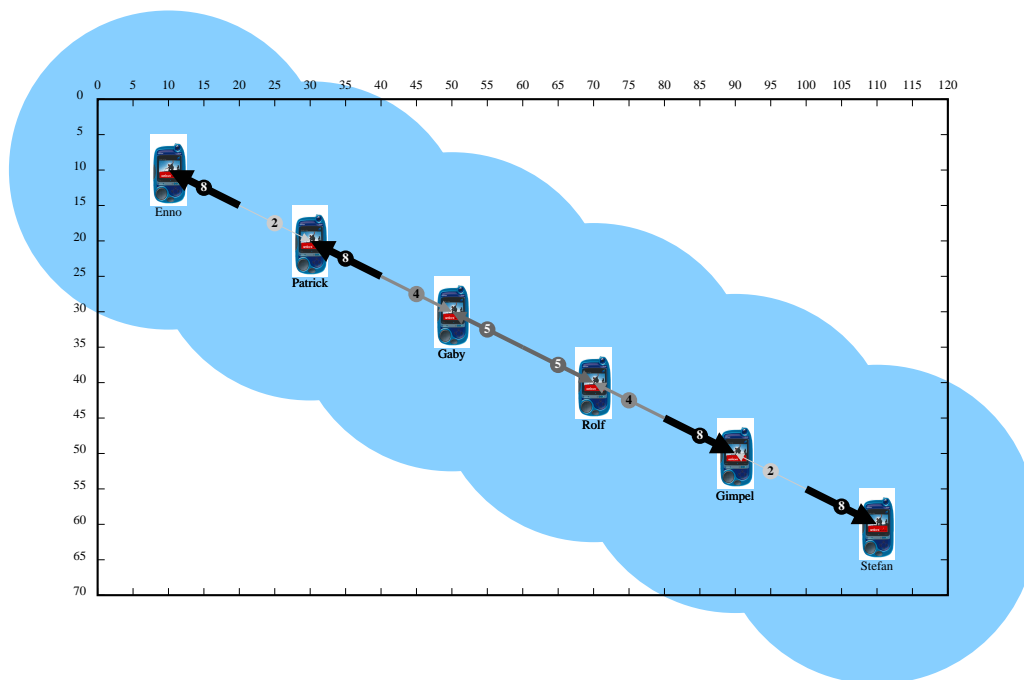


Figure 1: A simple chain configuration of six wireless terminals with graphical representation of the terminals.

Figure 2 shows the same configuration without graphical representation of the terminals.

¹The suffix `fig` is automatically generated

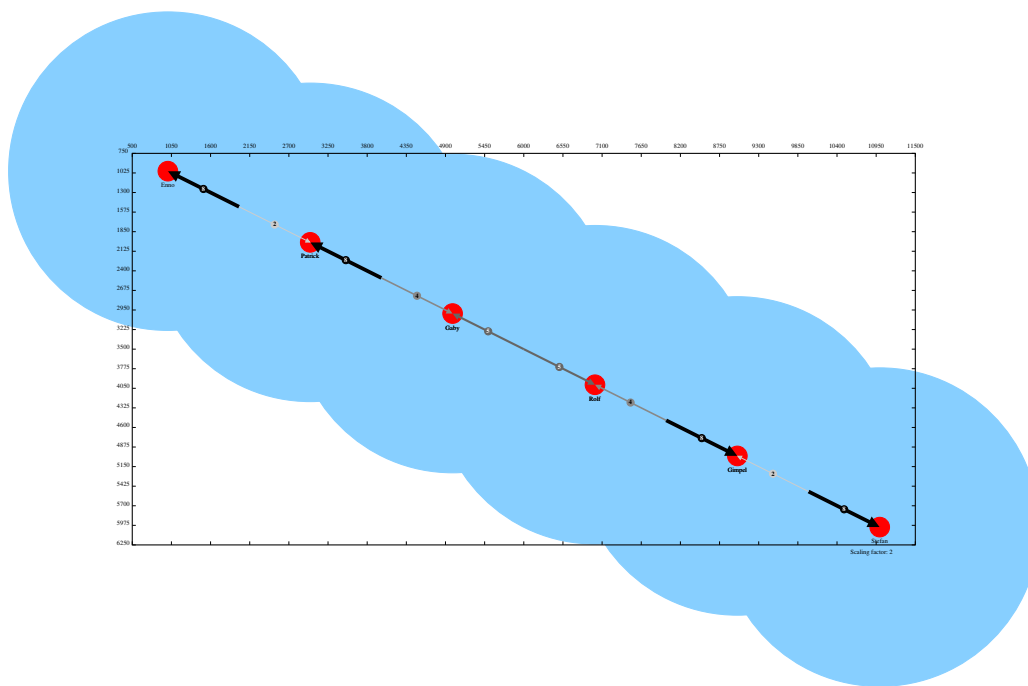


Figure 2: A simple chain configuration of six wireless terminals.

The XFIG tool version 3.2 can display elements of a given figure as a function of their depth level. This is very helpful to illustrate specific parts of the figure. By switching off (i.e., hiding) certain depth levels we can focus on the remaining parts of the figure. Table 1 gives the depth levels of the XFIG output. The terminal (node) representation (red dot or graphic), the terminal name, the bounding box, and the coverage area have static depth levels.

In Figure 2 six wireless terminals form the chain topology. The link quality is non-symmetrical, e.g., terminal Enno hears terminal Patrick with better quality than vice versa. This example is taken from [1], where a CDMA system is considered. For all terminals the transmission range is depicted. The area is 70 m by 120 m. On the x axis and the y axis the range in meters is given.

Table 1: Depth Level of the ViTAN XFIG output.

XFIG element	depth level
quality label on edge	quality level (i.e., 1, 2, ..., 8)
edge	600 + quality level
terminal (node)	700
names/ID	701
bounding box with numbers	800
coverage area	900

The color of the edges is a function of the link quality. Higher link quality given by larger values in the connectivity matrix is depicted by darker grey colors, while lower quality levels are depicted with lighter grey colors. The colors are generated automatically by ViTAN because these grey scaled colors are not part of the XFIG tool which offers only a limited set of colors.

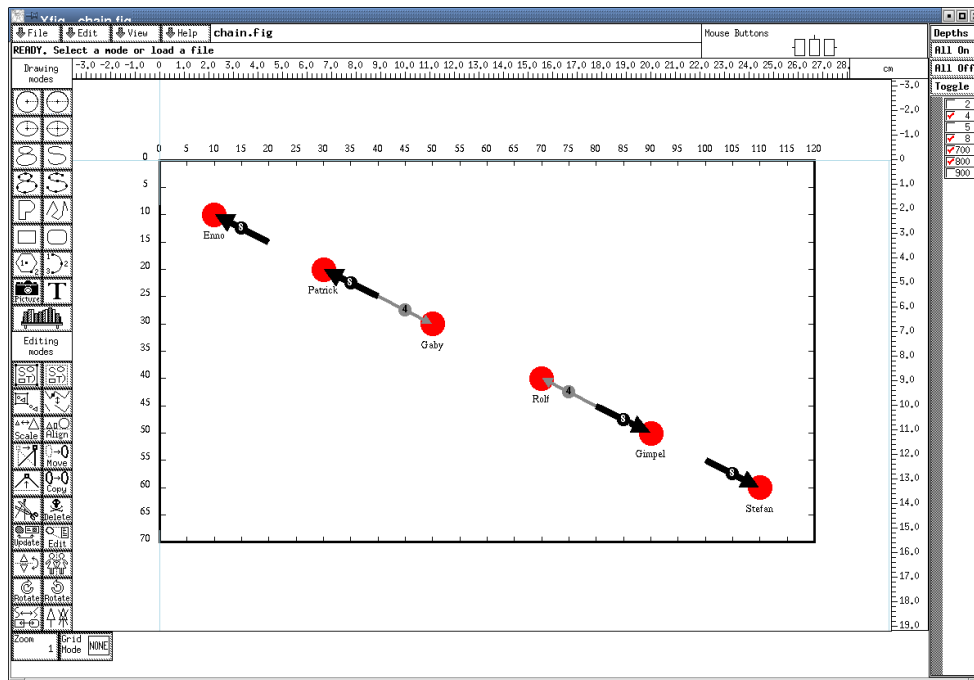


Figure 3: A screenshot of the XFIG tool using the depth levels for the *chain* scenario.

In addition, the width of the edges is a function of the link quality, with thicker edges representing higher link quality. For better illustration only width values between 1 and 8 are valid. In case higher quality levels are chosen in the connectivity matrix the values for the width are adjusted proportionally.

In Figure 3 the XFIG tool with the chain output is depicted without the depth levels 2, 5, and 900. Therefore the coverage and the links with quality levels 2 and 5 are missing.

6. Export to Other Graphic Formats

After generating the fig file any popular graphic format can be obtained by using the `fig2dev` program².

```
fig2dev -L GRAPHICFORMAT file.fig output
```

All common graphic formats such as `gif`, `jpeg`, `png`, `ppm`, `ps`, `eps`, and `pdf` are supported. For a full list of all supported graphic formats please refer to the `fig2dev` man pages.

The ViTAN tool comes with a short shell script to generate `pdf` files for all example topologies. By simply evoking the script `export` the files in the graphic formats `fig`, `pdf`, and `png` are generated. The script is given in Appendix C. The `pdf` files are displayed in the following section.

7. Examples

In the following we show some illustrative examples. Figure 4 gives again the chain scenario but this time with higher quality levels. In Figure 5 a bridge topology is depicted. Figure 6 shows 9 wireless terminals in a circle topology. Figure 7 gives the Manhattan scenario with different link qualities. Figure 8 and Figure 9 depict randomly generated topologies with 22 and 44 wireless terminals, respectively. The link qualities in all the examples are generated with the CDMA simulator introduced in [1].

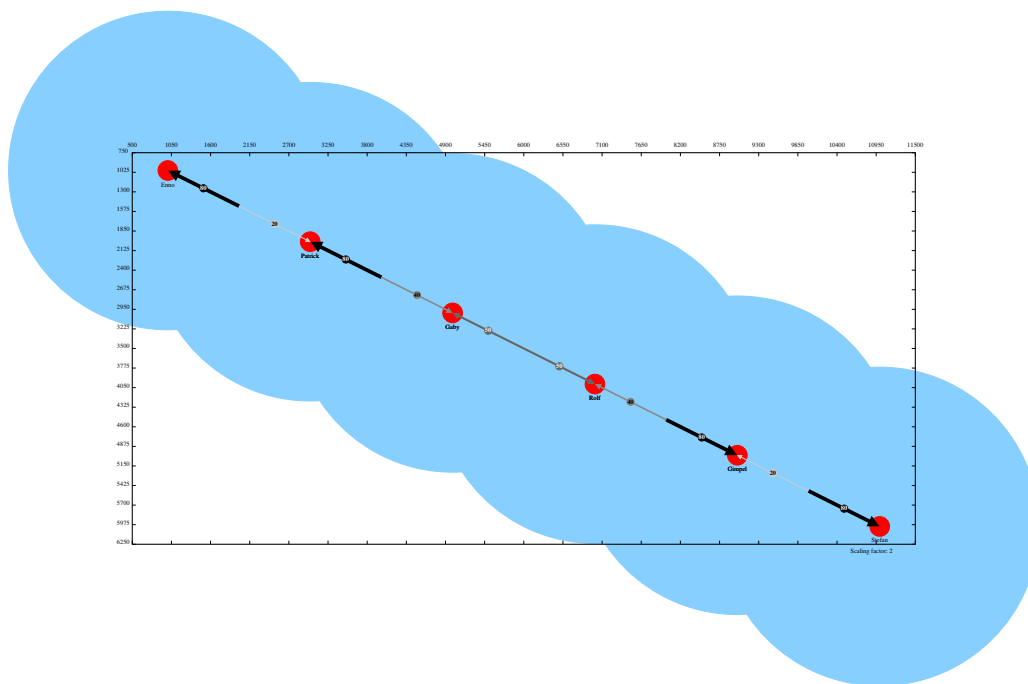


Figure 4: A simple chain configuration of six wireless terminals with higher quality levels.

²The `fig2dev` program is part of nearly every linux distribution and sometimes comes along with the `xfig` tool. [2]

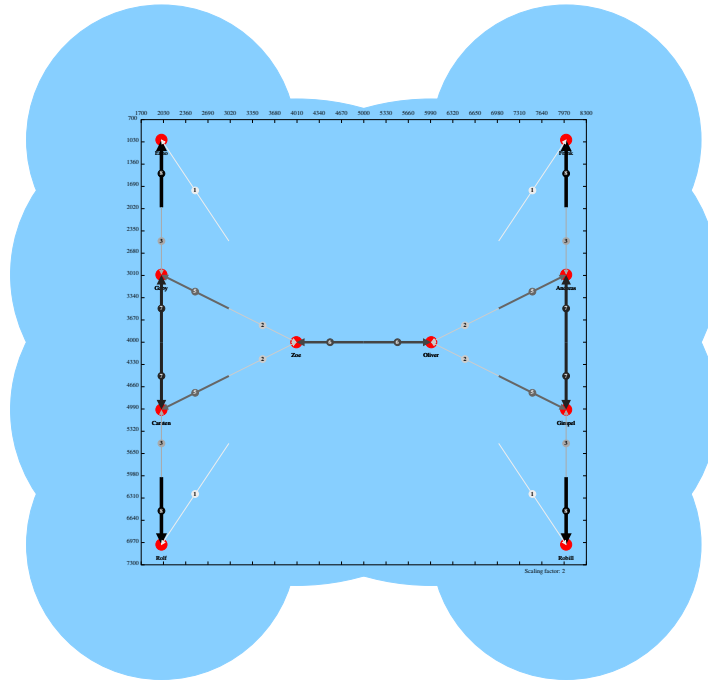


Figure 5: The *bridge* topology with 10 terminals.

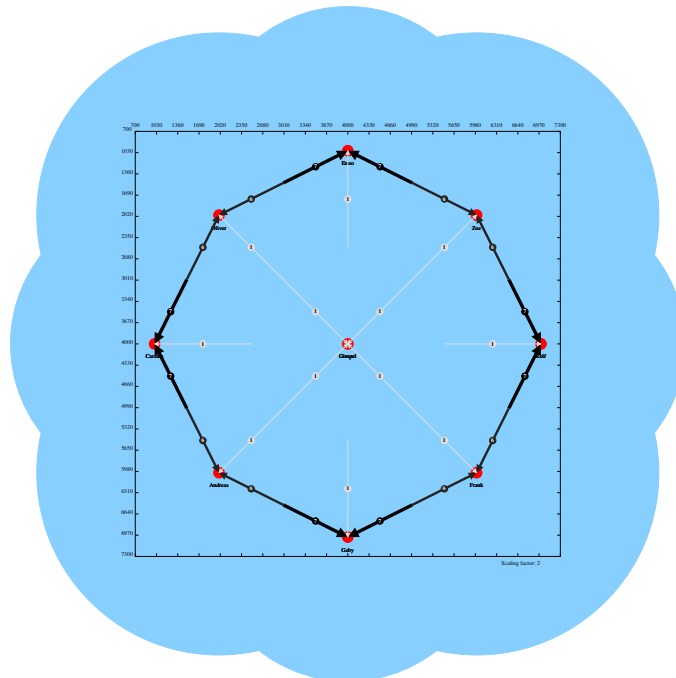


Figure 6: The *circle* topology with 9 terminals.

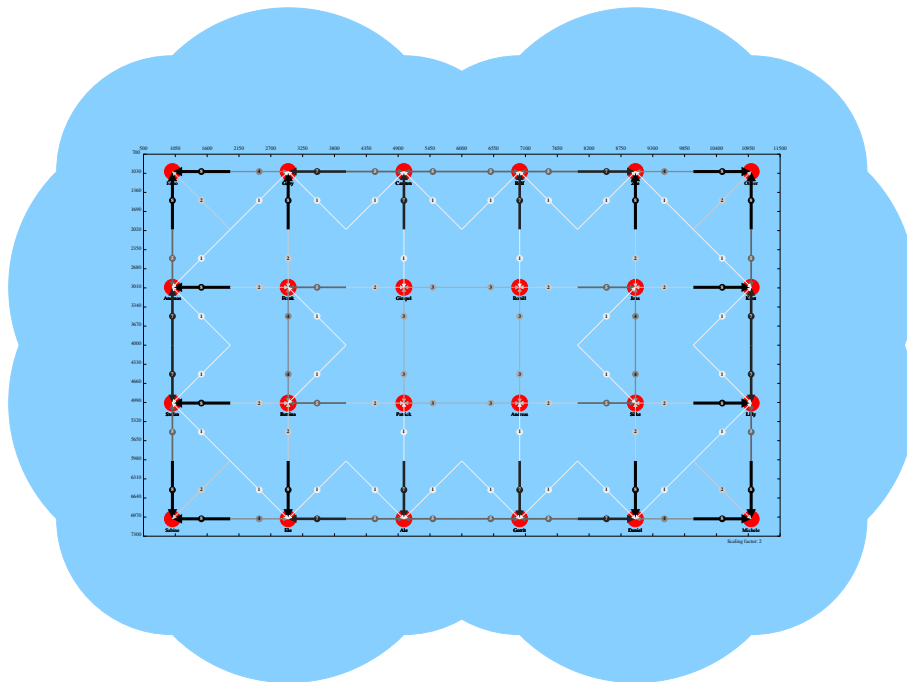


Figure 7: The *Manhattan* topology.

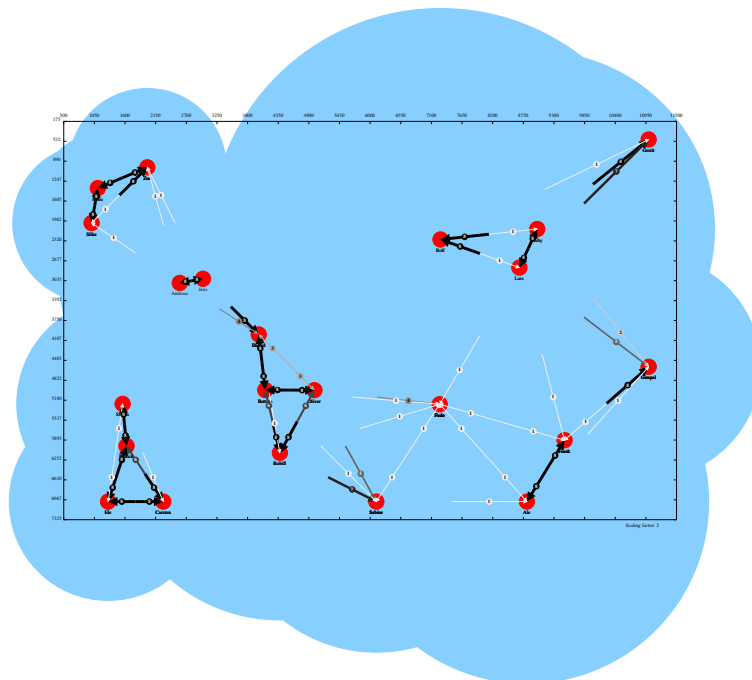


Figure 8: 22 randomly distributed wireless terminals.

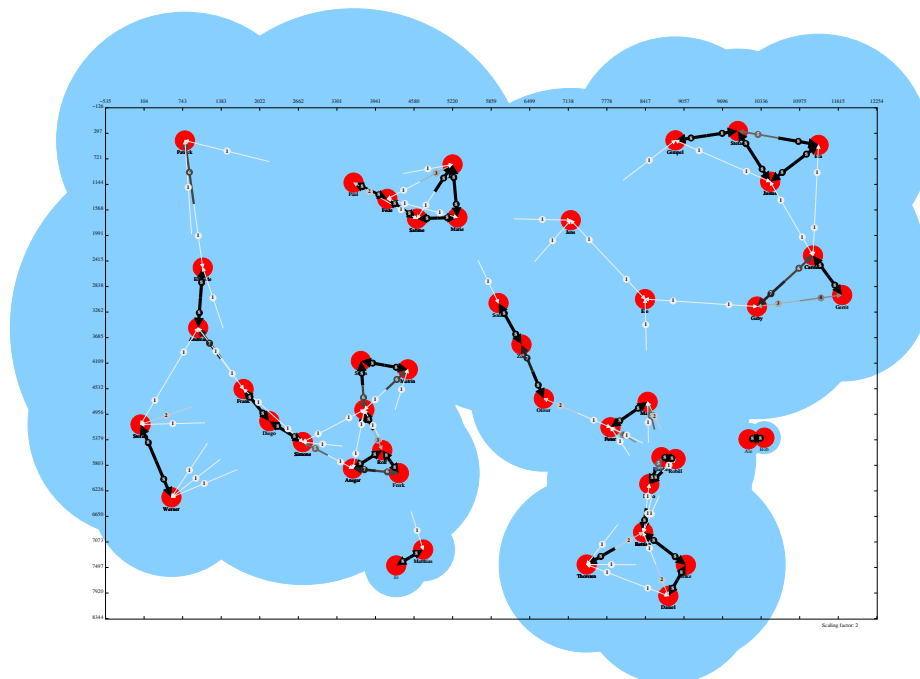


Figure 9: 44 randomly distributed wireless terminals.

8. Future Work

We encourage other researchers to work on the tool and to add new functionalities. Every contribution will be added to the tool by naming the author at the following reports. Some features that would be nice additions to the ViTAN tool are:

- User Interface for
 - specifying the input/output file
 - displaying the figure
 - manipulating the data
- Refinements of the visual presentation, e.g., restricting the coverage area to the inside of the bounding box
- Introducing float values for the quality
- automatic recognition of the size of a gif figure

We invite everyone to participate in this work.

9. Acknowledgement

We would like to thank Enno Ewers for his helpful advice on Perl programming.

References

- [1] F. H. P. Fitzek, P. Seeling, and M. Reisslein. Link Level Design Issues for IP based Multi-Hop Communication Systems. In *7th WWRP Meeting*. World Wireless Research Forum, December 2002. Eindhoven. 10, 12
- [2] Ian MacPhedran. Xfig tool and related software. <http://duke.usask.ca/~macphed/soft/fig/>, 1985-1995. 6, 12
- [3] S. Sutanthavibul, B. V. Smith, P. King, C. Blanc, and C. Schlick. Xfig 3.2 format. <http://duke.usask.ca/~macphed/soft/fig/FORMAT3.2.txt>. 6

A. Download Information

The ViTAN tool and related examples can be downloaded from the following web pages:

acticom GmbH

F.Fitzek

Research & Development

<http://www.acticom.de/vitan.html>

fitzek@acticom.de

Arizona State University

P. Seeling and M. Reisslein

Department of Electrical Engineering

<http://www.eas.asu.edu/~mre/vitan>

patrick.seeling@asu.edu and reisslein@asu.edu

Universita di Ferrara

M. Zorzi

Dipartimento di Ingegneria

http://www-tlc.ing.unife.it/new_cite/tools/vitan

zorzi@ing.unife.it

B. ViTAN Source Code

```
#!/usr/bin/perl

my %connection = ();

#input parameters
$file=$ARGV[0];

if ($ARGV[1]) {$XFIGname=$ARGV[1];} else {$XFIGname=SAVE}
if ($ARGV[2]) {$nographic=$ARGV[2];} else {$nographic=0}
10 if ($ARGV[3]) {$portrait=$ARGV[3];} else {$portrait=0}

$scale=1;
$entry=0;
$run=0;
$maxvalconnection=0;
$minvalconnection=0;
$maximumset=0;
$minstep=1;
open (FILE, $file) || die "cannot open file $file";
20 #here the coordinates can be specified default landscape
$XMAX=0;
$YMAX=0;
$XMIN=10000;
$YMIN=10000;

if ($ARGV[6] && $ARGV[7] && $ARGV[4] && $ARGV[5]) {
    $XMAX=$ARGV[6];$YMAX=$ARGV[7];$XMIN=$ARGV[4];$YMIN=$ARGV[5];$maximumset=1;
}

30 print "#####\n";
print "# ViTAN - Visualisation tool for ad hoc networks #\n";
print "# #\n";
print "# verison 1.1 - acticom mobile networks #\n";
print "#####\n";

while (<FILE>) {
    chomp;
    @elements = split;
40 push @table, [ @elements ];
    if ($portrait) {
        $xkor{$entry}=int($table[$entry][1]);
        $ykor{$entry}=int($table[$entry][0]);
    }
    else {
        $xkor{$entry}=int($table[$entry][0]);
        $ykor{$entry}=int($table[$entry][1]);
    }
    if ($maximumset==0) {
50 if (int($xkor{$entry}) > $XMAX) {$XMAX=int($xkor{$entry});}
    if (int($ykor{$entry}) > $YMAX) {$YMAX=int($ykor{$entry});}
    }
```

```

    if ( int($xkor{$entry}) < $XMIN) {$XMIN=int($xkor{$entry});}
    if ( int($ykor{$entry}) < $YMIN) {$YMIN=int($ykor{$entry});}
}
print "$xkor{$entry} $ykor{$entry} ";
$name{$entry}=$table[$entry][2];
if ($xkor{$entry}<=$XMAX && $ykor{$entry}<=$YMAX && $xkor{$entry}>=$XMIN
&& $ykor{$entry}>=$YMIN) {
    $radius{$entry}=$table[$entry][3];
    $terminalpresentation{$entry}=$table[$entry][4];
60    print "$name{$entry} $radius{$entry} $terminalpresentation{$entry} ";
    for ($terminals=0;$terminals<=#{$table[0]}-5;$terminals+=1) {
        $help=$terminals+5;
        $connection{$entry}{$terminals}=int($table[$entry][$help]);
        print "$connection{$entry}{$terminals} ";
        if ($connection{$entry}{$terminals}>$maxvalconnection) {
            $maxvalconnection=$connection{$entry}{$terminals};}
        if ($connection{$entry}{$terminals}<$minvalconnection) {
            $minvalconnection=$connection{$entry}{$terminals};}
    }
    print "\n";
    $inboundingbox{$entry}=1;
70 }
    else {
        print "$name{$entry} is not in the bounding box!\n";
        $inboundingbox{$entry}=0;
        $run++;
    }
    $entry++;
}
$display=$entry-$run;
print "$entry WTs are in the list and $display WTs are depicted\n";
80 close(FILE);

#scaling
$scaling=1;
if ($scale) {
    if (($XMAX-$XMIN)<12000 && ($YMAX-$YMIN)<8000) {
        while (($XMAX-$XMIN)<12000 && ($YMAX-$YMIN)<8000) {
            print "Scaling up!\n";
            for ($WT=0;$WT<$entry;$WT++) {
90         if ($inboundingbox{$WT}) {
                $xkor{$WT}=$xkor{$WT}*2;
                $ykor{$WT}=$ykor{$WT}*2;
                $radius{$WT}=$radius{$WT}*2;
            }
        }
        $scaling=$scaling*2;
        $XMAX=$XMAX*2;$XMIN=$XMIN*2;$YMAX=$YMAX*2;$YMIN=$YMIN*2;
    }
}
100 if (($XMAX-$XMIN)>24000 && ($YMAX-$YMIN)>16000) {
    while (($XMAX-$XMIN)>24000 && ($YMAX-$YMIN)>16000) {

```

```

    print "Scaling down!\n";
    for ($WT=0;$WT<$entry;$WT++) {
        if ($inboundingbox{$WT}) {
            $xkor{$WT}=$xkor{$WT}/2;
            $ykor{$WT}=$ykor{$WT}/2;
            $radius{$WT}=$radius{$WT}/2;
        }
    }
110   $scaling=$scaling/2;
      $XMAX=$XMAX/2;$XMIN=$XMIN/2;$YMAX=$YMAX/2;$YMIN=$YMIN/2;
    }
  }
}

if ($maximumset==0) {
    $yoffset=int(($YMAX-$YMIN)/20);;
    $xoffset=int(($XMAX-$XMIN)/20);;
120   $XMAX= int($XMAX+$xoffset);
      $YMAX= int($YMAX+$yoffset);
      $XMIN= int($XMIN-$xoffset);
      $YMIN= int($YMIN-$yoffset);
}

if ($portrait) {print "picture format: $XMAX $YMAX (portrait)\n";}
else {print "picture format: $XMIN $YMIN $XMAX $YMAX (landscape)\n";}
130 if ($nographic) {print "no pictures used to represent wireless nodes\n";}
    else { print "using picture to represent wireless nodes\n";}

#deleting connection of non displayed WTs
for ($WT=0;$WT<$entry;$WT++) {
    for ($actWT=0;$actWT<$entry;$actWT++) {
        if ($inboundingbox{$actWT}==0) {
            $connection{$WT}{$actWT}=0;
            $connection{$actWT}{$WT}=0;
        }
    }
140 }

#generating the XFIG file
open (XFIGFILE, ">".$XFIGname.".fig") || die "cannot open result file";
print XFIGFILE"#FIG 3.2\n";
if ($portrait) {
    print XFIGFILE"Portrait\n";
}
else {
150   print XFIGFILE"Landscape\n";
}
print XFIGFILE"Center\n";
print XFIGFILE"Metric\n";
print XFIGFILE"A4\n";
print XFIGFILE"100.00\n";

```

```

print XFIGFILE" Single\n";
print XFIGFILE"-2\n";
print XFIGFILE" 1200 2\n";

print "generating extra colors for XFIG tool\n";
160 @hexdigit = ('F', 'E', 'D', 'C', 'B', 'A', '9', '8', '7', '6', '5', '4', '3', '2', '1', '0'
    );
for ($colorcounter=0;$colorcounter<=$maxvalconnection;$colorcounter+=
    $minstep) {
    $value=40+$colorcounter;
    $factor=int(255/$maxvalconnection);
    $first=int($colorcounter*$factor/16);
    $second=int(( $colorcounter*$factor-$first)/16);
    $color{$colorcounter}=$value;
    print "$value # $hexdigit[$first]$hexdigit[$second]$hexdigit[$first]
        $hexdigit[$second]$hexdigit[$first]$hexdigit[$second]\n";
    print XFIGFILE" 0 $value # $hexdigit[$first]$hexdigit[$second]$hexdigit[
        $first]$hexdigit[$second]$hexdigit[$first]$hexdigit[$second]\n";
    }
170 #bounding box
print XFIGFILE" 2 2 0 3 0 7 800 0 -1 0.000 0 0 -1 0 0 5\n";
print XFIGFILE"$XMIN $YMIN $XMAX $YMIN $XMAX $YMAX $XMIN $YMAX $XMIN $YMIN\n
    ";
    $step=int(( $YMAX-$YMIN)/20);
    for ($h=$YMIN;$h<=$YMAX;$h+=$step) {
        print XFIGFILE" 2 1 0 1 0 7 800 0 -1 0.000 0 0 -1 0 0 2\n";
        $help=$XMIN+100;
        print XFIGFILE"$XMIN $h $help $h\n";
        $meter=int($h/($scaling));
180 if ($scaling>=1) {$help=$XMIN-250;} else {$help=$XMIN-500}
        print XFIGFILE" 4 1 0 800 0 0 12 0.0000 4 1350 0 $help $h $meter\\001\n";
        print XFIGFILE" 2 1 0 1 0 7 800 0 -1 0.000 0 0 -1 0 0 2\n";
        $help=$XMAX-100;
        print XFIGFILE"$help $h $XMAX $h\n";
    }
    $step=int(( $XMAX-$XMIN)/20);
    for ($h=$XMIN;$h<=$XMAX;$h+=$step) {
        print XFIGFILE" 2 1 0 1 0 7 800 0 -1 0.000 0 0 -1 0 0 2\n";
        $help=$YMIN+100;
190 print XFIGFILE"$h $YMIN $h $help\n";
        $meter=int($h/($scaling));
        $help=$YMIN-150;
        print XFIGFILE" 4 1 0 800 0 0 12 0.0000 4 1350 0 $h $help $meter\\001\n";
        print XFIGFILE" 2 1 0 1 0 7 800 0 -1 0.000 0 0 -1 0 0 2\n";
        $help=$YMAX-100;
        print XFIGFILE"$h $help $h $YMAX\n";
    }
print XFIGFILE " 4 1 0 800 0 0 12 0.0000 4 1350 0 " .($XMAX-1250)." " .($YMAX
    +250)." Scaling factor: $scaling\\001\n";

200 #placing the nodes and coverage
for ($h=0;$h<$entry;$h++) {

```

```

if ($inboundingbox{$h}) {
  if ($nographic) {
    $xlength=int(($XMAX-$XMIN)/80);
    $ylength=int(($XMAX-$XMIN)/80);
    print XFIGFILE"1 4 0 3 4 4 700 0 20 0.000 1 0.0000 $xkor{$h} $ykor{$h
      } $xlength $ylength 0 0 0 0\n";
  }
  else {
210    $xlength=int(($XMAX-$XMIN)/20);
    $ylength=int(($YMAX-$YMIN)/15);
    print XFIGFILE"2 5 0 1 0 -1 700 0 -1 0.000 0 0 -1 0 0 5\n";
    print XFIGFILE" 0 terminalpics/$terminalpresentation{$h}.gif\n";
    #some more quick hack
    $x1=$xkor{$h}-int($xlength/2);
    $y1=$ykor{$h}-int($ylength/2);
    $x2=$x1+$xlength;
    $y2=$y1;
    $x3=$x2;
    $y3=$y2+$ylength;
220    $x4=$x1;
    $y4=$y3;
    $x5=$x1;
    $y5=$y1;
    print XFIGFILE" $x1 $y1 $x2 $y2 $x3 $y3 $x4 $y4 $x5 $y5\n";
  }
  print XFIGFILE"1 4 0 3 11 11 900 0 20 0.000 1 0.0000 $xkor{$h} $ykor{$h
    } ".int($radius{$h})." ".int($radius{$h})." 0 0 0 0\n";
}
}

230 #drawing the connections
for ($WT=0;$WT<$entry;$WT++) {
  if ($nographic) {
    $help=$ykor{$WT}+450;
  }
  else {
    $help=$ykor{$WT}+650;
  }
  for ($actWT=0;$actWT<$entry;$actWT++) {
    if ($actWT!=$WT) {
240    $width=int(8*$connection{$WT}{$actWT}/$maxvalconnection);
    if ($width<1) {$width=1};
    if ($width<1) {$width=1};
    if ($width>8) {$width=8};
    if ($connection{$WT}{$actWT}) {
      print XFIGFILE"4 1 0 701 0 0 12 0.0000 4 1350 0 $xkor{$WT} $help
        $name{$WT}\\001\n";
      print "$name{$WT} is in the range of $name{$actWT} receiving with
        quality level $connection{$WT}{$actWT}\n";
      $depth=$connection{$WT}{$actWT};
      $arrowdepth=600+$depth;
      $arrow1=20*$width;
250    if ($arrow1<30) {$arrow1=30;}
    }
  }
}

```

```

    if ($arrow1<100) {$arrow1=100;}
    $arrow2=$arrow1;
    print XFIGFILE" 2 1 0 $width $color{$connection{$WT}{$actWT}} 7
        $arrowdepth 0 -1 0.000 0 0 -1 0 1 2\n1 1 $width.00 $arrow1.00
        $arrow2.00\n";
    $helpendpointx= int($xkor{$actWT} + 0.5*($xkor{$WT}-$xkor{$actWT}));
    $helpendpointy= int($ykor{$actWT} + 0.5*($ykor{$WT}-$ykor{$actWT}));
    $helpendpointx2= int($xkor{$actWT} + 0.95*($xkor{$WT}-$xkor{$actWT}
        ));
    $helpendpointy2= int($ykor{$actWT} + 0.95*($ykor{$WT}-$ykor{$actWT}
        ));
    print XFIGFILE"$xkor{$WT} $ykor{$WT} $helpendpointx $helpendpointy \
        n";
    $helpendpointx= int($xkor{$WT} + 0.25*($xkor{$actWT}-$xkor{$WT}));
    $helpendpointy= int($ykor{$WT} + 0.25*($ykor{$actWT}-$ykor{$WT}));
260  print XFIGFILE" 1 3 0 3 $color{$connection{$WT}{$actWT}} $color{
        $connection{$WT}{$actWT}} $depth 0 20 0.000 1 0.0000
        $helpendpointx $helpendpointy 100 100 0 0 0 0\n";
    $helpendpointx= int($xkor{$WT} + 0.25*($xkor{$actWT}-$xkor{$WT}));
    $helpendpointy= int($ykor{$WT} + 0.25*($ykor{$actWT}-$ykor{$WT}
        ))+50;
    $simplecolor=0;
    if ($connection{$WT}{$actWT}>int($maxvalconnection/2)) {$simplecolor
        =7;}
    print XFIGFILE" 4 1 $simplecolor $depth 0 2 12 0.0000 4 1350 0
        $helpendpointx $helpendpointy $connection{$WT}{$actWT}\\001\n";
        }
    }
270 }
close(FrameFILE);
print"ViTAN has finished.\n";

```

C. export Script

In the following the export script is given. Taking the ViTAN output in the `fig` format, this script generates pictures of the given examples in the `pdf`, `png`, and `jpg` formats. By evoking the script without any argument it will generate pictures without graphical representation of the wireless terminals. By evoking the script with argument 0 (`./export 0`) terminals are represented by the graphic specified in `terminals.gif`.

```
#!/bin/sh -x

rm *~
rm *.pdf
rm *.png
rm *.jpg

10 ./vitan.pl chain.input chain $1 0
   fig2dev -L pdf chain.fig chain.pdf
   fig2dev -L png chain.fig chain.png
   fig2dev -L jpeg chain.fig chain.jpg

   ./vitan.pl chain2.input chain2 $1 0
   fig2dev -L pdf chain2.fig chain2.pdf
   fig2dev -L png chain2.fig chain2.png
   fig2dev -L jpeg chain2.fig chain2.jpg

   ./vitan.pl chain3.input chain3 $1 0
20  fig2dev -L pdf chain3.fig chain3.pdf
   fig2dev -L png chain3.fig chain3.png
   fig2dev -L jpeg chain3.fig chain3.jpg

   ./vitan.pl chain5.input chain5 $1 0
   fig2dev -L pdf chain5.fig chain5.pdf
   fig2dev -L png chain5.fig chain5.png
   fig2dev -L jpeg chain5.fig chain5.jpg

   ./vitan.pl circle.input circle $1 0
30  fig2dev -L pdf circle.fig circle.pdf
   fig2dev -L png circle.fig circle.png
   fig2dev -L jpeg circle.fig circle.jpg

   ./vitan.pl bridge.input bridge $1 0
   fig2dev -L pdf bridge.fig bridge.pdf
   fig2dev -L png bridge.fig bridge.png
   fig2dev -L jpeg bridge.fig bridge.jpg

   ./vitan.pl manhattan.input manhattan $1 0
40  fig2dev -L pdf manhattan.fig manhattan.pdf
   fig2dev -L png manhattan.fig manhattan.png
   fig2dev -L jpeg manhattan.fig manhattan.jpg

   ./vitan.pl random.input random $1 0
   fig2dev -L pdf random.fig random.pdf
```

```
fig2dev -L png random.fig random.png
fig2dev -L jpeg random.fig random.jpg

./vitan.pl random2.input random2 $1 0
fig2dev -L pdf random2.fig random2.pdf
50 fig2dev -L png random2.fig random2.png
fig2dev -L jpeg random2.fig random2.jpg

./vitan.pl 30.input 30 $1 0
fig2dev -L pdf 30.fig 30.pdf
fig2dev -L png 30.fig 30.png
fig2dev -L jpeg 30.fig 30.jpg

./vitan.pl 30+.input 30+ $1 0
fig2dev -L pdf 30+.fig 30+.pdf
60 fig2dev -L png 30+.fig 30+.png
fig2dev -L jpeg 30+.fig 30+.jpg

./vitan.pl 100.input 100 $1 0
fig2dev -L pdf 100.fig 100.pdf
fig2dev -L png 100.fig 100.png
fig2dev -L jpeg 100.fig 100.jpg

./vitan.pl 100-.input 100- $1 0
fig2dev -L pdf 100-.fig 100-.pdf
70 fig2dev -L png 100-.fig 100-.png
fig2dev -L jpeg 100-.fig 100-.jpg

./vitan.pl chain.input chain 1 0 > chain.output

ls -l | grep -v total > directory
```