GRASS 3D Workshop – 3D-vector data how-to

J. Čepický

1Mendel University of Agriculture and Forestry Brno
Faculty of Forestry and Wood Technology
Department of Geoinformation Technologies

FOSS4G2006 Workshop
Visualization of 3D vector data using NVIZ

How to make 3D vector data

GRASS 3D vector modules
1. Visualization of 3D vector data using NVIZ
   - NVIZ

2. Howto make 3D vector data
   - GRASS Vector features
   - GRASS ASCII format
   - How to create 3D objects
   - House - 3D Object

3. GRASS 3D vector modules
   - v.extrude
   - v.drape
   - v.trees3d
<table>
<thead>
<tr>
<th>multiple surfaces in a 3D space</th>
</tr>
</thead>
<tbody>
<tr>
<td>thematic coloring</td>
</tr>
<tr>
<td>draping GRASS vector files over the surfaces</td>
</tr>
<tr>
<td><a href="http://grass.itc.it/nviz">http://grass.itc.it/nviz</a></td>
</tr>
</tbody>
</table>
Visualization of 3D vector data using NVIZ

How to make 3D vector data

GRASS 3D vector modules

NVIZ

multiple surfaces in a 3D space
thematic coloring
draping GRASS vector files over the surfaces

http://grass.itc.it/nviz

$ nviz elev=elevation
Visualization of 3D vector data using NVIZ
How to make 3D vector data
GRASS 3D vector modules

NVIZ
Visualization of 3D vector data using NVIZ
How to make 3D vector data
GRASS 3D vector modules
Visualization of 3D vector data using NVIZ
How to make 3D vector data
GRASS 3D vector modules

NVIZ

ˇCepický

GRASS 3D Workshop – 3D-vector data how-to
Visualization of 3D vector data using NVIZ

How to make 3D vector data

GRASS 3D vector modules

NVIZ

Čepický

GRASS 3D Workshop – 3D-vector data how-to
Visualization of 3D vector data using NVIZ

How to make 3D vector data

GRASS 3D vector modules

GRASS Vector features

GRASS ASCII format

How to create 3D objects

House - 3D Object
GRASS Vector features

- **2D**
  - P: point
  - L: line
  - B: boundary
  - C: centroid

- **3D**
  - F: face
  - K: kernel
1. Visualization of 3D vector data using NVIZ
   - NVIZ

2. Howto make 3D vector data
   - GRASS Vector features
   - GRASS ASCII format
   - How to create 3D objects
   - House - 3D Object

3. GRASS 3D vector modules
   - v.extrude
   - v.drape
   - v.trees3d
GRASS-ASCII Vector format

```
$ cat grassdata/LausanneDemoData/PERMANENT/\n    /txt/house.txt
B 5 1
 100 100
 500 100
 500 500
 100 500
 100 100
 1 1
```
GRASS-ASCII Vector format

B  5  1

- B – Boundary
- 5 – Number of coordinates
- 1 – Number of layers
GRASS-ASCII Vector format

B  5  1
  
  . . .
  500  500
  
  . . .
  1  1
GRASS-ASCII Vector format

Import of ASCII vector file

```sh
$ cat grassdata/ ... /txt/house.txt | v.in.ascii -n format=standard \ out=house
$ d.vect house
```
Visualization of 3D vector data using NVIZ

How to make 3D vector data

GRASS 3D vector modules

GRASS Vector features
GRASS ASCII format
How to create 3D objects
House - 3D Object

TOC

1. Visualization of 3D vector data using NVIZ
   - NVIZ

2. How to make 3D vector data
   - GRASS Vector features
   - GRASS ASCII format
   - How to create 3D objects
   - House - 3D Object

3. GRASS 3D vector modules
   - v.extrude
   - v.drape
   - v.trees3d
The best module for creation of 3D-vector maps:

- `v.in.ascii`
Import

- The best module for creation of 3D-vector maps:
  
  \texttt{v.in.ascii}
Import

- The best module for creation of 3D-vector maps:
  `v.in.ascii`

  ```
  v.in.ascii -z in=file.txt out=vector3d
  ```
The best module for creation of 3D-vector maps:

```
v.in.ascii
```

```
v.in.ascii -z in=file.txt out=vector3d
```

or

```
cat file.txt | v.in.ascii -z out=vector3d
```
GRASS-ASCII Vector format

$ cat grassdata/LausanneDemoData/PERMANENT/txt/3Dascii-01.txt

B 5 1
100 100 100
500 100 100
500 500 100
100 500 100
100 100 100

1 1

$
Visualizing first 3D object

# creating of new "DEM"
$ r.mapcalc dem=0

# importing ASCII file
$ cat grassdata/LausanneDemoData/PERMANENT/\txt/3Dascii-01.txt | \v.in.ascii -zn out=bound3d format=standard #--o

# visualize
$ nviz elev=dem vect=bound3d
Visualizing first 3D object
1. Visualization of 3D vector data using NVIZ
   - NVIZ

2. How to make 3D vector data
   - GRASS Vector features
   - GRASS ASCII format
   - How to create 3D objects
   - House - 3D Object

3. GRASS 3D vector modules
   - v.extrude
   - v.drape
   - v.trees3d
House - 3D Object

$ v.out.ascii house | make_2D_to_3D | v.in.ascii out=house
House - 3D Object

$ v.out.ascii house format=standard

ORGANIZATION:
DIGIT DATE:   
DIGIT NAME:   
MAP NAME:     
MAP DATE:     
MAP SCALE:    1
OTHER INFO:
ZONE:        0
MAP THRESH:  0.000000
VERTI:
  B  5 1
    100 100
    500 100
    500 500
    100 500
    100 100
    1 1
House - 3D Object

```sh
$ v.out.ascii house format=standard | \
grep "^ ([0-9][0-9] \+ *[0-9] \( *[0-9] \) \{2,3\})" | \
sed -e "s/\(^ \)\|\|\( \(*\$\)\)///g" -e "s/ \+/\_/g"
100_100
500_100
500_500
100_500
100_100
$
```
GRASS-ASCII vector format – house

```bash
#!/bin/sh
last=""
roof="F 5\n"
for line in $(v.out.ascii format=standard in=$1 | \
    grep "^ \+[\(0-9][0-9]\+[\.-][0-9]* \+[0-9]* \+[0-9]\+[0-9]\+[0-9]\+[0-9]\{2,3\}" | \
    sed -e "s/\(^ \))\(( *$\)//g" -e "s/ \+/\_/g" ); do

    line=$( echo "$line" | sed "s/\(\)/ /g" )

    if [ -n "$last" ]; then
        echo "F 5";
        echo "$last 0";
        echo "$line 0";
        echo "$line 100";
        echo "$last 100";
        echo "$last 0";
    fi;

    roof="$roof $line 100\n"
    last=$line
done
```

- Cepicky
- GRASS 3D Workshop – 3D-vector data how-to
GRASS-ASCII vector format – house

$ house_3d.sh house
F 5
100 100 0
500 100 0
500 100 100
100 100 100
100 100 0
F 5
500 100 0
...
$ house_3d.sh house | v.in.asii -nz out=house3d format=standard
$ nviz elev=dem vect=house3d
GRASS-ASCII vector format – house
Visualization of 3D vector data using NVIZ

Howto make 3D vector data

GRASS 3D vector modules

v.extrude
v.drape
v.trees3d

1. Visualization of 3D vector data using NVIZ
   - NVIZ

2. Howto make 3D vector data
   - GRASS Vector features
   - GRASS ASCII format
   - How to create 3D objects
   - House - 3D Object

3. GRASS 3D vector modules
   - v.extrude
   - v.drape
   - v.trees3d
v.extrude

- Since 2006 in GRASS 6.1 cvs
- 2D-objects (lines and polylines) → 3D-objects
- start height can be set according to digital elevation model
- attribute of the object can be used for object height
Visualization of 3D vector data using NVIZ
How to make 3D vector data
GRASS 3D vector modules

v.extrude

2D-vector industry
v.extrude

- **2D-vector** industry
- `db.columns`
  - `table=industry`
  - `database=grassdata/LausanneDemoData/PERMANENT/dbf/CAT`
  - `height`
v.extrude

- **2D-vector industry**
- `db.columns table=industry`  
  `database=grassdata/LausanneDemoData/PERMANENT/dbf/`  
  `CAT height`
- `v.extrude`  
  `in=industry`  
  `out=industry3d`  
  `elev=elevation`  
  `hcol=height`
**v.extrude**

- **2D-vector** *industry*

- `db.columns`  
  `table=industry`  
  `database=grassdata/LausanneDemoData/PERMANENT/dbf/CAT`  
  `height`

- `v.extrude`  
  `in=industry`  
  `out=industry3d`  
  `elev=elevation`  
  `hcol=height`
v.extrude

- 2D-vector industry
- db.columns
table=industry
database=grassdata/LausanneDemoData/PERMANENT/dbf/CAT
height
- v.extrude
in=industry
out=industry3d
elev=elevation
hcol=height

(c) Markus Neteler 2006
1 Visualization of 3D vector data using NVIZ
   - NVIZ

2 How to make 3D vector data
   - GRASS Vector features
   - GRASS ASCII format
   - How to create 3D objects
   - House - 3D Object

3 GRASS 3D vector modules
   - v.extrude
   - v.drape
   - v.trees3d
v.drape

- Since 2005 in GRASS 6.1 cvs
- 2D-objects (lines and polylines) $\rightarrow$ 3D-objects according to digital elevation model
Visualization of 3D vector data using NVIZ
How to make 3D vector data
GRASS 3D vector modules

v.extrude
v.drape
v.trees3d

v.drape

```
  r.contour in=elevation
  out=contours step=2
```

| GRASS 6.1 CVS | Monitor: x0 | Location: LausanneDemoData |
Visualization of 3D vector data using NVIZ
How to make 3D vector data
GRASS 3D vector modules

v.drape

- `r.contour in=elevation`  
  `out=contours step=2`
- `v.to.points in=contours`  
  `out=points`

![Image of 3D vector data visualization using NVIZ](image-url)
v.drape

- `r.contour in=elevation`  
  `out=contours step=2`

- `v.to.points in=contours`  
  `out=points`

- `v.out.ascii`  
  `in=points|v.in.ascii`  
  `out=points2d`
Visualization of 3D vector data using NVIZ
How to make 3D vector data
GRASS 3D vector modules

v.drape

r.contour in=elevation
out=contours step=2
v.to.points in=contours
out=points
v.out.ascii
in=points|v.in.ascii
out=points2d
v.delaunay in=points2d
out=tin2d
v.drape

- `r.contour in=elevation`  
  `out=contours step=2`

- `v.to.points in=contours`  
  `out=points`

- `v.out.ascii`  
  `in=points|v.in.ascii`  
  `out=points2d`

- `v.delaunay in=points2d`  
  `out=tin2d`

- `v.info tin2d`
Visualization of 3D vector data using NVIZ

How to make 3D vector data

GRASS 3D vector modules

v.extrude
v.drape
v.trees3d

v.drape

- $r$.contour in=elevation
  out=contours step=2
- v.to.points in=contours
  out=points
- v.out.ascii
  in=points | v.in.ascii
  out=points2d
- v.delaunay in=points2d
  out=tin2d
- v.info tin2d
- v.drape in=tin2d
  rast=elevation out=tin
v.drape

- `r.contour in=elevation out=contours step=2`
- `v.to.points in=contours out=points`
- `v.out.ascii in=points|v.in.ascii out=points2d`
- `v.delaunay in=points2d out=tin2d`
- `v.info tin2d`
- `v.drape in=tin2d rast=elevation out=tin`
- `v.info tin`
Visualization of 3D vector data using NVIZ
How to make 3D vector data
GRASS 3D vector modules

**v.drape**

```
- r.contour in=elevation
  out=contours step=2
- v.to.points in=contours
  out=points
- v.out.ascii
  in=points|v.in.ascii
  out=points2d
- v delaunay in=points2d
  out=tin2d
- v.info tin2d
- v.drape in=tin2d
  rast=elevation out=tin
- v.info tin
- nviz elev=dem vect=tin
```
Visualization of 3D vector data using NVIZ

Howto make 3D vector data

GRASS 3D vector modules

1. Visualization of 3D vector data using NVIZ
   - NVIZ

2. Howto make 3D vector data
   - GRASS Vector features
   - GRASS ASCII format
   - How to create 3D objects
   - House - 3D Object

3. GRASS 3D vector modules
   - v.extrude
   - v.drape
   - v.trees3d
v.trees3d

- [http://les-ejk.cz/?cat=grass](http://les-ejk.cz/?cat=grass)
- Developed since 2004 (perl)
- Since 2006 GRASS C-Module
- Purpose: 2D-points → 3D-trees

---

Visualization of 3D vector data using NVIZ
How to make 3D vector data
GRASS 3D vector modules

v.trees3d

http://les-ejk.cz/?cat=grass
Developed since 2004 (perl)
Since 2006 GRASS C-Module
Purpose: 2D-points → 3D-trees

v.trees3d in=trees out=trees3d
elev=elevation diam=6 height=20 hvar=3
vvar=4 hvar=20 spruce=50 beech=25 fir=25

Visualization of 3D vector data using NVIZ
How to make 3D vector data
GRASS 3D vector modules

v.trees3d

- http://les-ejk.cz/?cat=grass
- Developed since 2004 (perl)
- Since 2006 GRASS C-Module
- Purpose: 2D-points → 3D-trees

```
v.trees3d in=trees out=trees3d
elev=elevation diam=6 height=20 hvar=3
vvar=4 hvar=20 spruce=50 beech=25 fir=25
v.trees3d in=trees out=trees3d
elev=elevation diam=6 hcol=height
specol=species hvar=3 clength=60
```

---

v.trees3d

v.trees3d in=trees out=trees3d elev=elevation
  hvar=10 vvar=4 spruce=50 beech=25 fir=25
v.trees3d

- "Forest" should be up to 350 m
  r.mapcalc
  forest="if(elevation > 350, 1, null())"

Trees stubs

```sh
r.mapcalc
forest="if(elevation > 350, 1, null())"
```

g.region res=10
r.to.vect in=forest out=stubs feature=point
tress
v.trees3d in=stubs out=forest3d spruce=50 beech=30 fir=20 hvar=5
elevation=elevation
```
v.trees3d

- "Forest" should be up to 350 m
  
  ```
  r.mapcalc
  forest="if(elevation > 350, 1, null())"
  ```

- Trees stubs
  
  ```
  g.region res=10
  r.to.vect in=forest
  out=stubs feature=point
  ```
v.trees3d

- "Forest" should be up to 350 m
  
  ```
  r.mapcalc
  forest="if(elevation > 350, 1, null())"
  ```

- Trees stubs
  
  ```
  g.region res=10
  r.to.vect in=forest
  out=stubs feature=point
  ```

- Tress
  
  ```
  v.trees3d in=stubs
  out=forest3d spruce=50 beech=30 fir=20 hvar=5 vvar=20 elevation=elevation
  ```
v.trees3d

- "Forest" should be up to 350 m
  
  ```
  r.mapcalc
  forest="if(elevation > 350, 1, null())"
  ```

- Trees stubs
  
  ```
  g.region res=10
  r.to.vect in=forest
  out=stubs feature=point
  ```

- Tress
  
  ```
  v.trees3d in=stubs
  out=forest3d spruce=50 beech=30 fir=20 hvar=5 vvar=20
  elevation=elevation
  ```
v.trees3d

- "Forest" should be up to 350 m
  
  ```
  r.mapcalc
  forest="if(elevation > 350, 1, null())"
  ```

- Trees stubs
  
  ```
  g.region res=10
  r.to.vect in=forest
  out=stubs feature=point
  ```

- Trees
  
  ```
  v.trees3d in=stubs
  out=forest3d spruce=50 beech=30 fir=20 hvar=5 vvar=20
  elevation=elevation
  ```
End

jachym.cepicky@centrum.cz
http://Les-ek.cz

The presentation was reported thanks to GA ČR project nr. 526/03/H036 „Current stage and trends of development of forests in cultural landscape“. 