

WvW/RoadDesign
Version 1.5/ august 2020
Copyright © 2014-2020 by Carnetsoft BV

All rights reserved. No part of the contents of this document may be reproduced or transmitted in any form without the written permission of Carnetsoft.



RoadDesign

1. INTRODUCTION	2
2. DATABASES AND FILES.....	2
GENERAL WORKFLOW AND PITFALLS	2
1. <i>Open an existing database</i>	2
2. <i>Make changes</i>	2
3. <i>Important files created and where to put them</i>	2
2.1 RELATION BETWEEN PROGRAMS AND FILES.....	4
2.2 EGG, BAM AND REF FILES	5
2.2.1 <i>Structure of the egg files</i>	5
2.2.2 <i>Structure of the ref files</i>	9
2.3 FILE LOCATIONS FOR RUNNING A SIMULATION	10
3. USER INTERFACE.....	11
3.1 READING AND MERGING OF DATABASES	12
3.2 USER INTERFACE	14
3.3 START WITH BUILDING A NETWORK OF ROADS	16
3.4 THEN ADD GRAPHICAL OBJECTS	16
4 UNITS	17
5 GROUPS	17
6 SEGMENTS.....	17
6.1 CREATING SEGMENTS.....	17
6.2 LANES	21
6.3 SIGNS	21
7 INTERSECTIONS	31
8 CONNECTION NODES.....	35
9 GRAPHICAL ELEMENTS.....	40
9.1 FIELDS (LANDPOLY)	40
9.2 TREES (TREEPOLY)	43
9.3 FACADES (FACEPOLY)	46
9.4 EXTERNAL (3D) OBJECTS (EXTERNOBJECT).....	49
10 PATHS.....	51

1. Introduction

RoadDesign is a designer for creating road network files for the Carnetsoft driving simulation software. With RoadDesign you design the road network geometry and this results in both a 'logical' database and a graphical database. The graphical databases are in a format that is required for the Panda3d rendering engine. Panda3d is a realtime game engine that was developed by Disney Studios and Carnegie Mellon University. For more information about Panda2d, see: <http://www.panda3d.org/>

RoadDesign lets you design a 3D database in a 2D working environment, where you see the 2D representation in a top-down way. It lets you design a network of roads and road signs, plus underlying fields, vegetation, facades, and positions of 3D objects. The 3D objects (for example, buildings) are read into the designer and have been created by other software such as 3D studio. The 3D objects have been converted into egg and bam files, see next chapter.

2. Databases and files

General workflow and pitfalls

To make an entirely new graphical database requires some level of skill that not all users have from the start. Because of that it is highly recommended to use existing databases that come with the research simulator software installation or to slightly modify existing databases. Here are a few point to keep in mind when you do that.

The workflow is:

1. *Open an existing database.*

2. *Make changes.*

- If you make changes, make sure that you assign a texture when making a new landpoly (field) or facade. If you don't assign a texture you may get strange errors later, for example when you compile the *.egg file in a *.bam file of <TRef> { -1 }, which means that a texture can't be found.
- Please try not to create Trees (facepolys) because these are implemented as billboards that result in a state change in the simulator every frame, and because of that these trees cannot be optimized for rendering, probably resulting in lower frame rates and performance issues if you use a lot of them. So it is advised to use 3D External Tree objects instead.
- You can save changes in between your work and continue to make changes. However, sometimes texture references are not saved in the *.egg file resulting in the <TRef> { -1 } error. If you are finished and you are ready to compile the *.egg file into a *.bam, please just open the *.rdb file in RoadDesign and save the file again. Now all texture references are found if you have defined them.
- You can also merge a database with another one. See paragraph 3.1 for more details.

3. *Important files created and where to put them.*

A number of files are created in the RoadDesign\databases folder. The *.rdb file is the source file that must be read into RoadDesign, the *.sdb file contains the scenery elements in binary form. For the database to be used in the simulator, the following are important:

*.egg files.

This contains the roads, fields, facades and TreePolys (although it is recommended not to use these) plus a number of External Objects that occur a lot (for example poles alongside the road and streetlights) plus objects that contain a simulation (for example traffic lights). In the file **externalobjects.tex** (in the folder where RoadDesign is located), which defines the location of all external objects, the 2nd column specifies what to do with these objects. These are the values:

- 0 : add position and orientation of the object to the *.egg file
- 1: add position and orientation of the object to the *.ref file and in the simulation this is added to the 'close' tiles.
- 2: add position and orientation of the object to the *.ref file and in the simulation this is added to the 'far' tiles

When the *.bam database and the *.ref file are read in the simulation, there are a number of optimization steps. One of these steps is to organize all objects in square tiles that are visualized in total in order to minimize GPU data transfers, because this makes optimal use of the band width of the GPU. *Close tiles* are visualized when the driver is close near the center of that tile and should contain objects that are low, so you can't see them from a larger distance anyway. *Far tiles* are visualized already from a larger distance and should contain objects that are higher, for example a high building or a hill.

An *.egg file must be converted into a *.bam file for the simulator to read the file. This conversion is done with the program egg2bam.exe, which is located in the folder c:\DriveSim3\bin. So, if you want to convert an egg to a bam, make sure the folder c:\DriveSim3\bin is added to the environment PATH variable in your system configuration.

It is important to realize that the *.egg file contains references to a large number of textures and 3D objects, and all these things are located in folders that are located within the \models folder. This means that you have to copy the *.egg file from the \databases folder to the \models folder (c:\DriveSim3\Carnetsoft\models), and open a dos window in c:\DriveSim3\Carnetsoft\models and then apply egg2bam as: egg2bam -ps rel modelname.egg modelname.bam
This command is refused if the *.bam already exists, so make sure that you remove or rename an already existing modelname.bam before you apply egg2bam.

*.ref files.

This contains all external objects (with a 1 or 2 indicated in externalobjects.tex). If you want to change the type or position of existing models in the database it is usually much easier to just make these changes in the *.ref file with the TextPad 4 editor instead of making the changes in RoadDesign. If you just want to add a few new 3D objects to the database, then it's easier to add these in RoadDesign and simply copy the new lines of the new objects in the new *.ref file to the existing *.ref file in the \models folder. So, the *.ref file must be located in the \models folder too.

The *.ref file contains a list of 3D objects together with the position and orientation of the model. It is a text file that can easily be edited. However, the first line of the *.ref file contains information that is used by the rendering system (worldC.py and worldLR.py), for example:

```
1 1000 500 1 0 1 1
```

The first number (1) has a special meaning for specialized applications, simply leave that as it is. The 2nd number (1000) is the switch distance of the far tiles. So if you want to see high objects from a larger distance, then set it to 1000 and otherwise use a lower value. The 3rd number (500) is the switch distance of the close tiles. If you make this too high then much more tiles are rendered than required and this may result in poorer graphics performance (lower frame rate or hiccups). In the existing databases sensible values are chosen for these switch distances so it is advised to use the values as they were in the files that were shipped with

the research simulator software. So these switch distances are important optimisation parameters.

The 4th number (1) is used to indicate whether the database is used for *night driving*. It should be 1 if it is and 0 if it is not used for night driving. For example, rural.ref has a 1 for this field because it is used in a night driving scenario (P-Night2) as well as day driving scenarios (for example P-Traffic1)

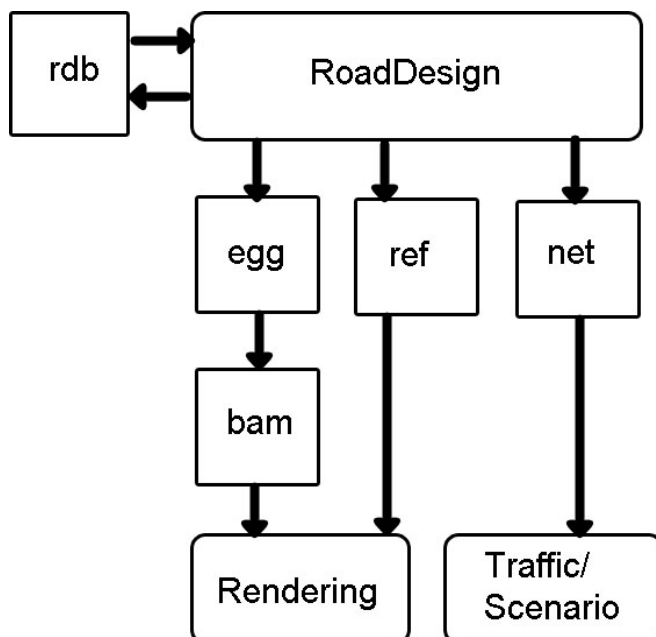
The 5th, 6th and 7th number are not used anymore because they are for older versions of the software.

*.net file.

The *.net file contains the logical database and represents the perspective of the world as seen by the autonomous traffic and the simulations (for example to measure the lateral position, the road you are driving on etc. This file must be copied to the \scenegraph folder because that is where the simulator will try to read it from. If the simulation starts and there is no modelname.net file in the \scenegraphs folder, to simulation will terminate with an error message from traffic.exe.

2.1 Relation between programs and files

The relation between files and programs is shown in the following figure.



The source file of a database is always the (binary) *.rdb file. This contains all the descriptions of roads, intersections, objects, etc. So, RoadDesign reads and writes a *.rdb file when you select a database.

The graphical database consists of two ascii files:

- an *.egg file
- a *.ref file

The logical database consists of an ascii file:

- a *.net file

Egg files are the database files used by Panda3d. Egg files are compiled into *.bam files because a bam file loads faster and requires less memory. The *.ref files contain references to 3D objects (buildings etc.) that are loaded in the graphics system together with the *.bam file. The rendering modules that read the *.bam and *.ref files are:

- c:\DriveSim3\python\ppython_center.exe
- c:\DriveSim3\python\ppython_left.exe
- c:\DriveSim3\python\ppython_right.exe

The *.net files contain definitions that are used by the traffic and scenario generation system:

- c:\DriveSim3\Carnetsoft\SimCarnet\traffic.exe
- in the script files a reference is made to a database and to paths and segments of the network of roads (as defined in the *.net file)

2.2 Egg, bam and ref files

Egg and **ref** files are stored when a database is saved and the Create 3D DB checkbox is ticked.

The egg files must be converted into bam files by running the egg2bam utility from the command line (via a DOS window), for example:

```
egg2bam -ps rel Highway.egg Highway.bam
```

To see all the options of the egg2bam utility, type egg2bam -h

2.2.1 Structure of the egg files

Sometimes you may want to modify things in the egg file because you convert it into a bam file. Egg files have the following structure:

1) List of materials used. Material numbers conform to the texture numbers, so for example <Material> 1 belongs to ><Texture> 1 etc.

2) List of all textures used. Textures for road signs, roadmarkings, TreePoly's and Facepoly's, are *.png (semi transparent). Textures for LandPoly's are *.jpg. For example,

```
<Texture> 42 {
  "textures/trees/S_Deciduous07.png"
  <Scalar> wrap { repeat }
  <Scalar> wrapu { clamp }
  <Scalar> wrapv { clamp }
  <Scalar> minfilter { linear_mipmap_linear }
  <Scalar> magfilter { linear }
  <Scalar> envtype { modulate }
}
```

This refers to a texture on a TreePoly (a rotating billboard, for trees and bushes). The folders referred to are below the \models folder where the *.bam file is located. So, the actual folder where S_Deciduous07.png is located is: c:\DriveSim3\Carnetsoft\models\ttextures\trees\

3) This is followed by a long list of vertices (vertex pool), for example:

```
<VertexPool> vpool {
  <Vertex> 0 {
    -101.781 -173.814 0
    <UV> { 0 49.8838 }
```

```

}
<Vertex> 1 {
  -101.781 -170.864 0
  <UV> { 1 49.8838 }
}
....

```

Vertices are 3D points (x, y, z). Each vertex has a unique id, referred to in polygons.

4) This is followed by a list of segments. These are the roadsegments with id's as defined in the *.net file. So, suppose the egg file is Highway.egg, then there's also a file named Highway.net, with segments that have id's that refer to the segment id's in the egg file. For example:

```

<Group> "Segment 0 subseg 0" {
  <SwitchCondition> {
    <Distance> { 500 0 <Vertex> { -40.00 85.00 0.00}}
  }
  <Group> "Segment 0 subseg 0 lod 0" {
    <SwitchCondition> {
      <Distance> { 500 0 <Vertex> { -40.00 85.00 0.00}}
    }
    <Group> "Seg 0 0 Lod 0 objs" {
      <Polygon> {
        <Normal> { 0.0000 0.0000 1.0000 }
        <RGBA> { 1 1 1 1 }
        <TRef> { 0 }
        <BFace> { 1 }
        <VertexRef> { 3000 3001 3002 3003 <Ref> { vpool } }
      }
      <Polygon> {
        <Normal> { 0.0000 0.0000 1.0000 }
        <RGBA> { 1 1 1 1 }
        <TRef> { 0 }
        <BFace> { 1 }
        <VertexRef> { 3003 3002 3004 3005 <Ref> { vpool } }
      }
    }
  }
}

```

This is a definition of Segment 0. It is switched into view when you are 500 meters from position x,y = Vertex> { -40.00 85.00 0.00}. It consists of a list of polygons (triangles or squares). Each polygon refers to a texture (for example <TRef> { 0 }, which is texture 0 in the list of textures), and consists of a number of vertices shown in <VertexRef>.

Segments consist of subsegments, and may have road signs or roadmarkings connected to them.

5) This is followed by a list of intersections. These have the same id's as the intersections in the *.net file. For example:

```

<Group> "Inter 0 Lod 0" {
  <SwitchCondition> {
    <Distance> { 500 0 <Vertex> { 277.50 -965.69 0.00}}
  }
  <Group> in0 {
    <Polygon> {
      <Normal> { 0.0000 0.0000 1.0000 }
      <RGBA> { 1 1 1 1 }
      <TRef> { 2 }
    }
  }
}

```

```

    <BFace> { 1 }
    <VertexRef> { 44588 44589 44590 <Ref> { vpool } }
  }
  <Polygon> {
    <Normal> { 0.0000 0.0000 1.0000 }
    <RGBA> { 1 1 1 1 }
    <TRef> { 2 }
    <BFace> { 1 }
    <VertexRef> { 44588 44590 44591 <Ref> { vpool } }
  }
}

```

This is similar to the segments. .

6) After the intersections, there's a list of LandPoly's. A LandPoly is a ground texture, for example a grass field. For example:

```

<Group> "LandPoly 1" {
  <SwitchCondition> {
    <Distance> { 1000 0 <Vertex> { 225.00 -1918.64 0.00 } }
  }
  <Group> lp1 {
    <Polygon> {
      <Normal> { 0 0 1 }
      <RGBA> { 0.918 0.918 0.918 1 }
      <TRef> { 13 }
      <BFace> { 1 }
      <VertexRef> { 4 5 6 7 8 4 <Ref> { vpool } }
    }
  }
}

```

If the ground is blue while you are driving in the simulator, then the SwitchDistance must be increased. In this example this is 1000 meter, meaning that if you are more than 1000 meters from x,y,z = 225.00 -1918.64 0.00, the LandPoly is not visible. A landpoly can have any number of vertices and must consist of a closed set of vertices (first and last point connect). It always MUST have a texture connected to it (in this case texture 13). If a texture has not been defined in RoadDesign, then <TRef> { -1 } will give an error when the egg file is converted into a bam file.

7) After the LandPoly's there's a list of TreePoly's. A Treepoly is a rotating billboard, that rotates in a way that is always is turned at the the viewer, giving it a 3D impression. TreePoly's are trees, bushes and other vegetation, with a transparent texture on it. It's simply a vertical polygon that rotates. For example:

```

<Group> "TreePoly 2" {
  <SwitchCondition> {
    <Distance> { 500 0 <Vertex> { 260.00 -1908.64 0.00 } }
  }
  <Instance> bb2 {
    <Transform> {
      <Translate> { 260.00 -1908.64 0.00 }
    }
  }
  <Group> bbg2 {
    <Billboard> { axis }
    <Polygon> {
      <Normal> { 0 -1 0 }
    }
  }
}

```

```

    <RGBA> { 0.918 0.918 0.918 1 }
    <TRef> { 31 }
    <BFace> { 1 }
    <VertexRef> { 256 257 258 259 <Ref> { vpool } }
  }
}
}
}

```

8) This is followed by a list of FacePoly's. A FacePoly is a vertical transparent series of polygons where a façade is displayed. This can be a wall, a forest of a hill, in fact anything that's further away and is aimed to block the horizon. For example:

```

<Group> "FacePoly 0" {
  <SwitchCondition> {
    <Distance> { 1000 0 <Vertex> { 610.00 -2196.14 0.00 } }
  }
  <Group> fg0 {
    <Polygon> {
      <Normal> { 1.0000 -0.0000 0.0000 }
      <RGBA> { 1 1 1 1 }
      <TRef> { 88 }
      <BFace> { 1 }
      <VertexRef> { 3636 3637 3638 3639 <Ref> { vpool } }
    }
    <Polygon> {
      <Normal> { 0.9487 0.3162 0.0000 }
      <RGBA> { 1 1 1 1 }
      <TRef> { 88 }
      <BFace> { 1 }
      <VertexRef> { 3639 3638 3640 3641 <Ref> { vpool } }
    }
  }
  .....
}

```

The switching distance must be large enough to prevent that it pops up when you approach it.

9) The FacePolys are followed by a list of 3D objects. They are referred to as externobject. For example:

```

<Group> externobject7 {
  <SwitchCondition> {
    <Distance> { 750 0 <Vertex> { -500.00 -1700.00 0.00 } }
  }
  <Group> fo7 {
    <Instance> {
      <Transform> {
        <Rotate> { 90.00 0 0 1 }
        <Translate> { -500.00 -1700.00 0.00 }
      }
      <File> {
        "objects/PowerPylon1/PowerPylon1.egg"
      }
    }
  }
}
}
}

```


All 3D objects that are listed in the egg file must have the .egg extension, as in PolwerPylon1.egg. However, it is advisable to put the larger (in the sense of containing more polygons) 3D models in the *.ref file. So, a number of 3D models will be referred to in the egg file, while the larger 3D models will be referred to in the ref file.

2.2.2 Structure of the ref files

A ref file contains the reference to the larger 3D models.

For a description of the meaning of values on the first line see page 3.

```
1 1000 550 0 0 1 1
0 models/objects/House27/House27.bam 0 500 180.00 0 0 1 459.95 3830.00 0.00
0 models/objects/TowerCrane1/TowerCrane1.bam 0 750 0.00 0 0 2 -720.00 3360.00 0.00
0 models/objects/TowerCrane1/TowerCrane1.bam 0 750 225.00 0 0 2 -773.10 3334.01 0.00
0 models/objects/sitedumper/sitedumper.bam 0 500 0.00 0 0 1 -758.00 3376.51 0.00
```

This is followed by a list of references to 3D objects:

- a number that indicates if the texture of the model can be changed from within the simulation, 1 = dynamic model, 0 = static model (default)
- name of the model (must have the *.bam extension)
- obsolete
- obsolete
- heading (orientation along the z axis)
- pitch (usually 0)
- roll (usually 0)
- value to indicate if it is in close field (1) or far field (2)
- x coordinate in the database (world)
- y coordinate in the database (world)
- height (in meters)

Whether or not a 3D object is referenced in the *.ref file instead of the *.egg file, is determined by the contents of the file named: externalobjects.tex (in the folder where RoadDesign is located). Externalobjects.tex is read by RoadDesign.exe and lists the thumbnails that are shown in relation to the 3D object. Here's an example line:

```
house007.gif house007 1 0.0 500.0
```

There's an object, house007 (actually house007.bam, but listed here without an extension).

- house007.gif is the thumbnail. This file MUST be in the thumbnails folder and is a 64x64 gif texture.
- the 1 signifies that the object will be referenced in the *.ref file and not in the *.egg file (if it is 0 then it will be in the *.egg file)
- the 0.0 is the default height in meters. If you use a 3D object as a roadmarking, it must have some height, for example 0,01, as in:

```
Stop.gif Stop 1 0.01 300.0
```

This is a 3D object that consists of a text saying 'STOP', that is put on the ground in front of a stop sign.

- the last number (500.0) is the default switch distance.

2.3 File locations for running a simulation

When the database has been created, the files must be stored in the appropriate folders.

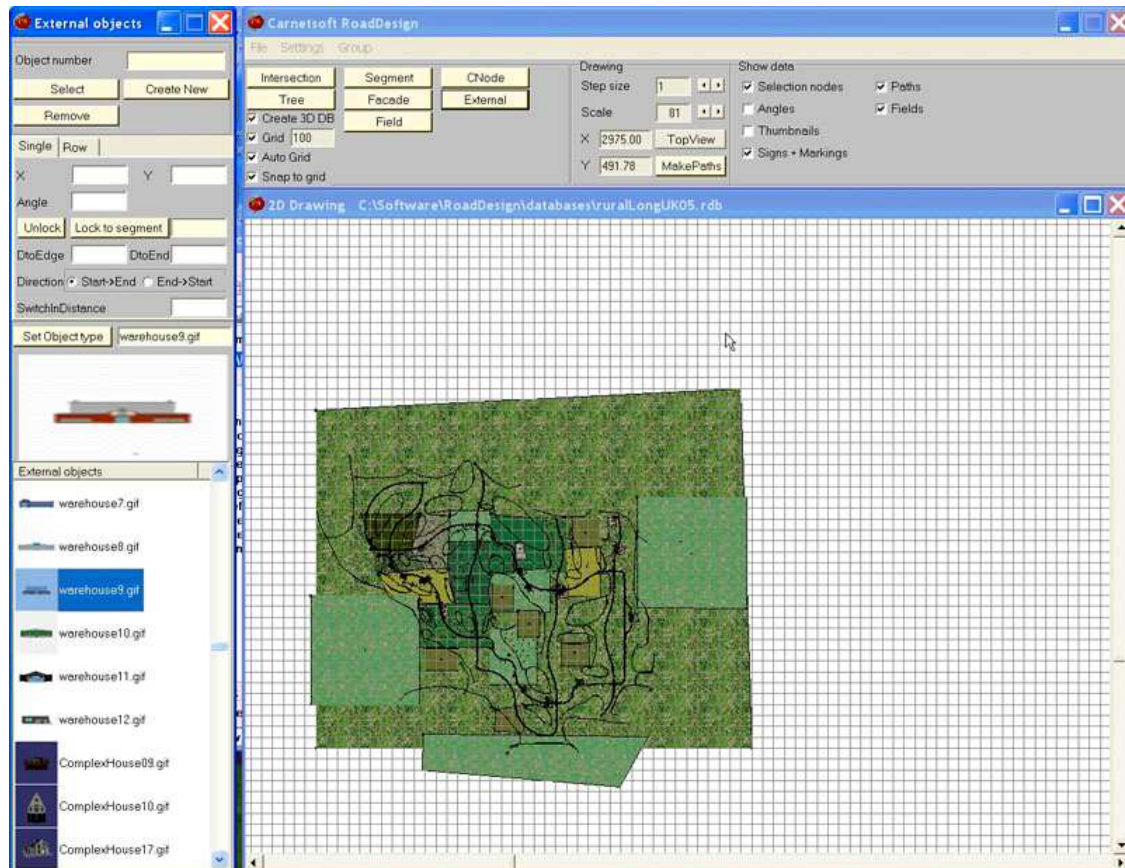
- *.bam file of the entire database: c:\DriveSim3\Carnetsoft\models\
- *.ref file of the database: c:\DriveSim3\Carnetsoft\models\
- *.net file c:\DriveSim3\Carnetsoft\SimCamet\scenegraphs

If the *.net file is not located in the \scenegraphs folder, traffic.exe will return an error when it tries to load the file.

If the *.bam file is not in the \models folder, the renderers will exit and return an error. If the *.ref file is not in the \models folder, it will simply not read and use the 3D models.

3. User interface

The following Figure shows an overview of a topview 2D view in the designer.



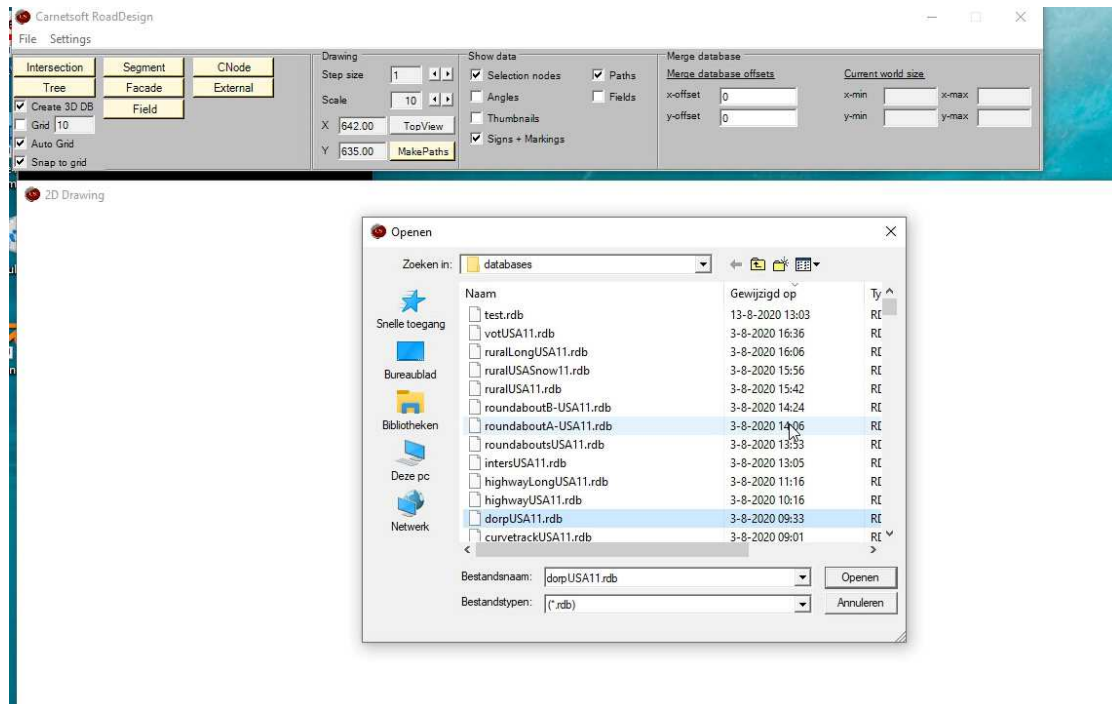
The design of databases takes place in a 2D view. Designing a road network geometry is much easier in a 2D view than in a 3D view.

A grid is drawn on the 2D drawing if the 'grid' checkbox is checked. The grid makes accurate positioning much easier and it gives you an idea of the distances between different objects. If the 'Auto Grid' box is checked, the dimensions of the grid are created automatically by the program. Depending on the scale of the 2D drawing, the distance between gridlines is either 1 meter, 10 meters, 100 meters or 1000 meters. The current value is always shown next to the draw grid checkbox. Sometimes it is useful if the user determines the grid size. In that case 'Auto grid' must be unchecked, and the user may then enter the required value in the field next to the 'Grid' checkbox. If the 'Snap to grid' checkbox is checked, all things that the user enters, such as positions of an intersection, a segment etc, snap to the nearest upper-left grid point. This allows precise positioning along the grid. If you want to position all things freely (not on the grid), then the 'Snap to grid' checkbox must be unchecked.

The general way the program works is as follows: Before you want something to be done, you indicate your intentions by pressing a button. For example, if you want to create an intersection, first press the 'Intersection' button'. Then press the 'Create New Inter' button. After that press the 'Position intersection' button. And only after that position your mouse somewhere in the 2D drawing window and press the left mouse button to position the intersection.

3.1 Reading and merging of databases

When databases have already been made, for example the databases that come with the research simulator installation, you can read an existing database (the *.rdb file) and process that, for example by adding intersections, segments etc. in the main menu, click on <File> and then <Open...>. The following selection box appears from you you can select an *.rdb file.

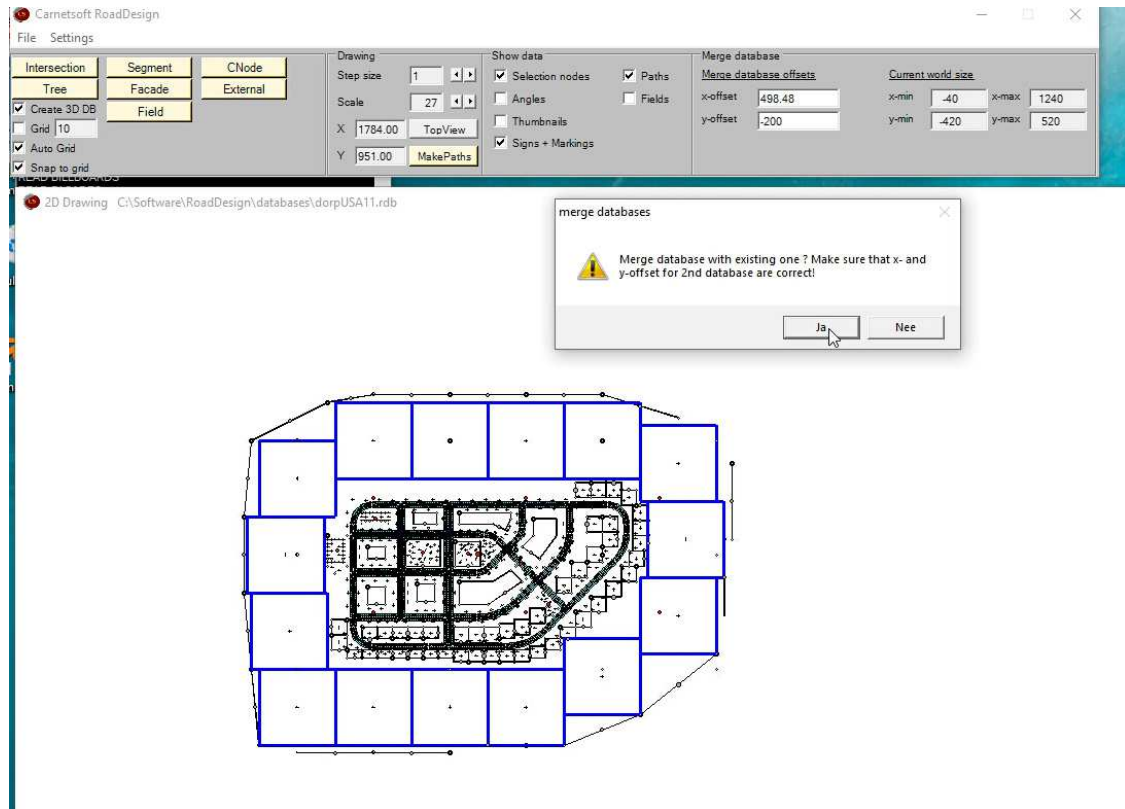


If the file has been opened and read, a top view of the database is shown. After that you can open an additional database and merge this with the one that is already opened, see the picture below. The dialog box says:

Merge database with existing one? Make sure that x- and y-offset for second database are correct!

In the <Merge databases> area of the user interface, you can see that an x-offset and an y-offset are filled in. This results in a displacement over the X and Y axis (in meters) such that the second database is aligned properly with respect to the first databases and intersections and roads do not overlap. In this case, the values are chosen such that an intersection in database 1 is exactly above an intersection in database 2, which makes it easier to connect the two intersections with a new segment. x-offset is 498.48 which means that the second database is shifted 498.48 meters to the right, and y-offset is -200 which means the 2nd database is shifted 200 meters down.

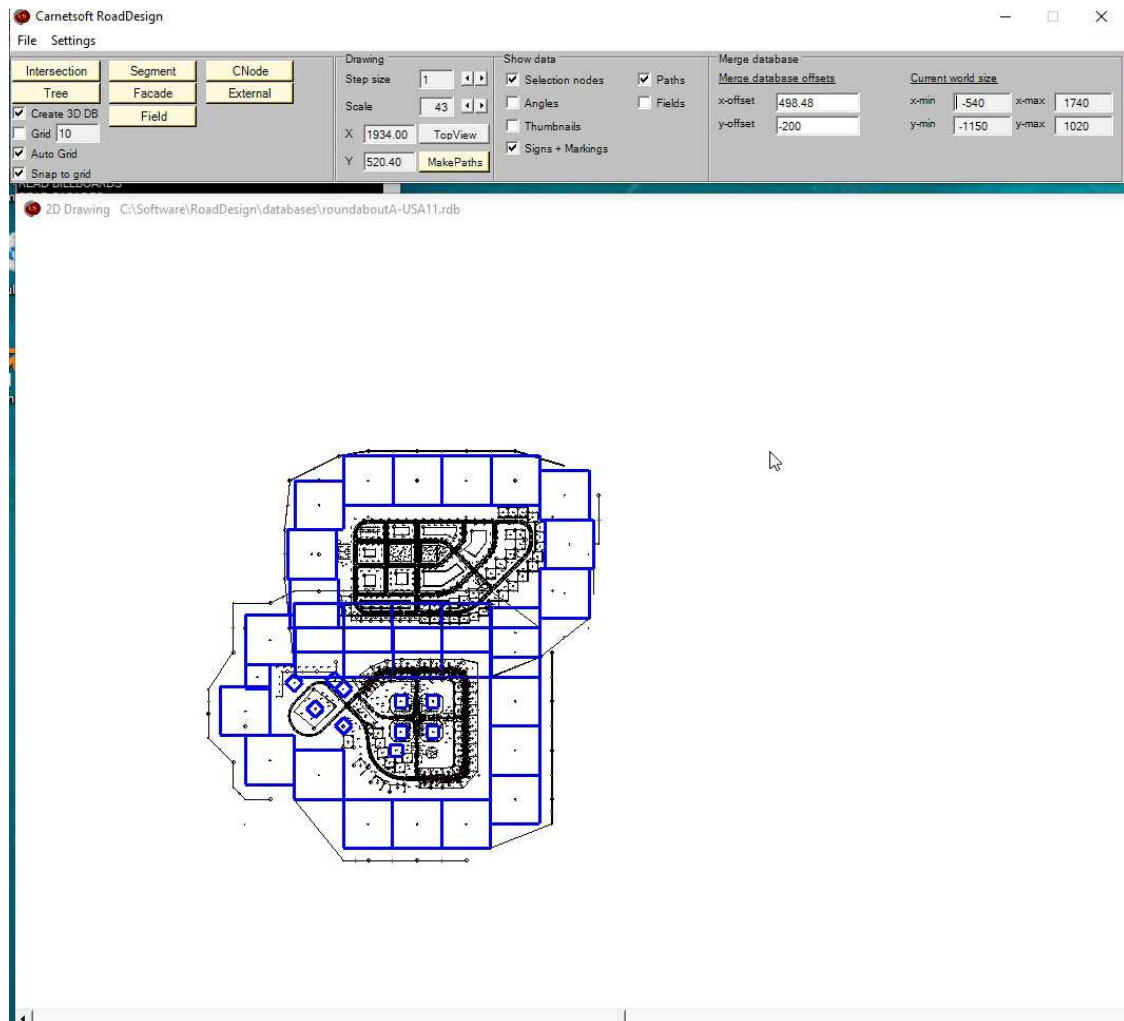
So, if you want to merge the already loaded database with another one, press Yes, otherwise press No. If you press No, the program asks "**The current database contains data, are you sure?**". If you click on Yes, then all existing work is cleared and you can read another database. If you do not want that, select No.



The result is the shown in the next picture.

The next steps typically are:

- remove objects that overlap, for example the landscape objects (blue blocks)
- add segments between the 2 databases so you can drive from one to the other
- dress the new database up with 3D objects
- save the new database with a new name.



If you want to read a new database you can't merge it with the existing ones because only 1 merging is permitted per session. In that case the program asks:
"The current database contains data, are you sure?". If you click on Yes, then all existing work is cleared and you can read another database. If you do not want that, select No.

3.2 User interface

The user interface (GUI) consists of three windows:

- 1) a main window
- 2) a 2D design window
- 3) an object window

1) The **main window** contains

- the main menu
- the buttons for selection of objects: intersections, segments, ConnectionNodes (Cnodes) and graphical objects (Fields, Trees, Facades and Externals).
- information on the cursor position and controls to view the 2D design

The <Show Data> section has a number of toggles that control what is drawn in the 2D Drawing window.

- Toggle Selection Nodes: toggles the drawing of selection nodes
- Toggle Angle Info : controls whether segment angle info is shown
- Toggle Thumbnails : if you add Files, Trees, Facades or Externals, a thumbnail on the 2D Drawing window shows the type of object. But these thumbnails may overlap other elements and may thus interfere with the design process. By toggling these thumbnails off, you can see the grid and the elements that you are designing.
- Toggle Signs and Markings : Toggles the redrawing of road signs and road markings
- Toggle Paths : after paths have been created (with the 'MakePaths' button), all path numbers are drawn on the 2D design window. This drawing is toggled on/off
- Toggle Fields : If you want to make screenshots with the path numbers, the fields sometimes make the path numbers hard to see. In that case the fields can be toggled on/off.

2) The **2D design window** contains a symbolic 2D drawing of the roadnet and associated graphical objects. The drawing can be scrolled using the horizontal and vertical scrollbars. In this window you can select parts of the drawing. The drawing can be scrolled up or down (over the Y-axis) with the vertical scrollbar. The drawing can be scrolled to left and right (over the X-axis) with the horizontal scrollbar. The stepsizes of horizontal and vertical scrolling can be adjusted by clicking on the stepsize buttons on the main window. By adjusting the scale on the main window the drawing is zoomed in or out. Clicking on the TopView button gives a topview of the entire roadnetwork database. The X and Y coordinate fields show the current coordinate position of the mouse in the 2D Drawing window. Moving the mouse to the right increases the X-coordinate. Coordinates are in meters. Moving the mouse up increases the Y-coordinate.

If you position the mouse in the 2D Drawing window, a selection of the drawing can be made by pressing the left mouse button and moving the mouse with the left mouse button pressed. The selection is made if you release the left mouse button.

3) The **object window** contains information concerning the respective object. Type of object is selected with the buttons on the main window.

The *Intersection* panel is the default startup panel. If you create a new database it's best to first create a new intersection. An intersection is a junction where one or more roads (segments) cross and are connected to. Although (the center of) intersections are represented as a point (X, Y) in space, their layout can be specified in the designer.

If you press the <Segment> button, the *segment* panel is displayed. Segments are stretches of road that can be either STRAIGHT or CURVE. A segment consists of at least one lane (a DLane or 'driving lane'). A segment has a direction (from START to END) and both the START and END are connected to other segments, intersections or ConnectionNodes.

If you press the <Cnode> button, the *Cnode* panel is displayed. A Cnode is an object where a segment splits in two different segments (a Split node) or where two segments join into one different segment (a Merge node). Cnodes then connect the lanes of incoming and outgoing segments.

If you press the <Field> button, the *field* panel is displayed. A field is used to dress up the database with areas of grass, water, soil, etc. A field is a polygon with as many vertices as you like. You can stick a texture on that polygon.

If you press the <Tree> button, the *Trees* panel is displayed. A Tree is a vertical board on which you can glue a texture with an alpha channel. Trees may rotate with the position of the driver, so that a 3D suggestion is created. Trees are used for trees and bushes.

If you press the <Façade> button, the *facade* panel is displayed. A facade is a sequence of vertical boards on which you can glue a texture with an alpha channel. Facades have a fixed orientation. They are used as a background for house fronts, fences, tree lines or hills.

If you press the <External> button, the *External objects* panel is displayed. External objects are 3D objects such as houses or street furniture (streetlights etc.).

3.3 Start with building a network of roads

The typical way to work with this designer is that you first build a network of roads. You build a road network by constructing segments, intersections and Cnodes. Segments are stretches of road with 1..n lanes. All segments connect either to another segment, to an intersection, or to a CNode. All connections have to be made explicitly by the user. An intersection is a point in space where 1..n segments connect to. A segment is either straight or curved. It has a fixed direction from Start to End. It may contain any number of road signs and road markings or other objects. When the database is saved and a graphical database is created by the system, segments are cut into sub-segments that are smaller for graphical optimization.

Each intersection (or junction) has a specific lay-out that can be specified by the user. An intersection may contain traffic lights and specific types of road markings.

A CNode connects a segment to two other segments by connecting the incoming lanes of a segment to outgoing lanes of other segments. A CNode can then be regarded as a switch board where lanes come in and lanes go out. The total number of incoming lanes must match the total number of outgoing lanes. A connectionNode has no specific geometry: it is just a logical switch board.

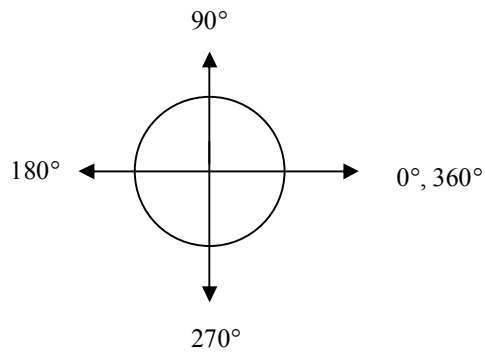
3.4 Then add graphical objects

After the road network geometry is defined, you know where the roads and intersections are. It is then time to add all kinds of graphical objects to the database. The following types of objects can be added to the database:

- Fields: these are landscape polygons containing a specific texture. The polygon is defined by the user and can contain any number of vertices. These landscape polygons can contain, for example, water, grass, different types of soil.
- Billboards: these are rectangular vertical boards on which a texture with alpha channel can be glued. Billboards turn with the position of the simulator driver, so that a 3D suggestion is generated. They typically contain trees and bushes.
- Facades: are rectangular boards with a fixed orientation on which a texture with alpha channel can be glued. Facades may contain house fronts, fences etc.
- External Objects: these are 3D objects such as houses and street furniture.

4 Units

All distances are in meters. All angles are specified in degrees in the user interface. Angles increase counterclockwise and are defined as follows:



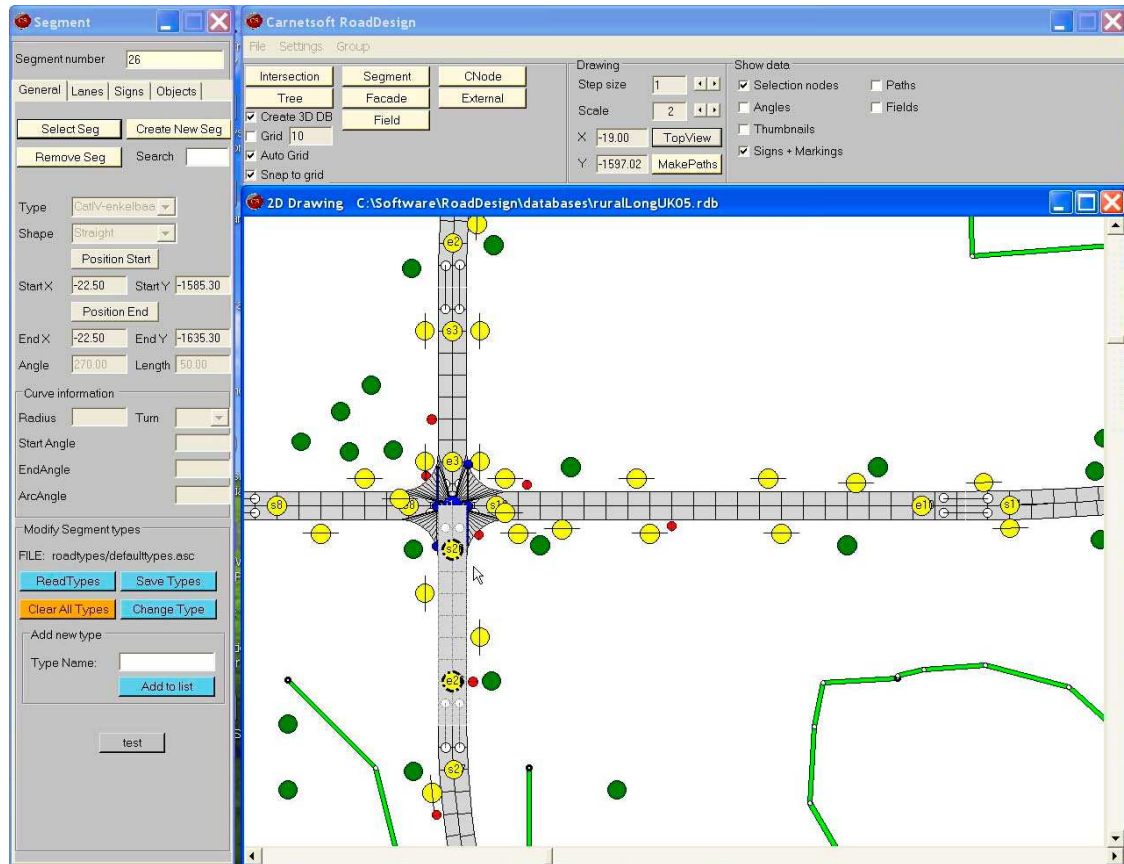
5 Groups

Groups have been disabled. All references to groups in this document must be ignored.

6 Segments

6.1 Creating segments

The following figure shows a segment that has been selected by clicking either on the START (yellow) node or the END (yellow) node.



To create a new segment do the following:

- In the central-top, click on 'Segments'. Then the 'Segment panel' shows up on the left side of the screen
- Now you can either create a new segment or select an existing segment. If you want to create a new segment, press the button 'Create New Seg'. Then the 'Segment number' field shows the value '-1'.
- Determine the 'Type' by selecting the appropriate segment type from the combobox.
- Determine the appropriate shape with the 'Shape' combobox. Shape is either 'Straight' or 'Curve'.
- Then click the button 'Position Start' or fill out the StartX and StartY coordinates. You can also first determine the end position if you like.
- Suppose you have clicked the button 'Position Start' In that case you must click the mouse pointer somewhere in the 2D Drawing form. The blue circle in the example represents an intersection hotspot that was created first. If you click on the intersection hotspot, the start of the segment is connected with the intersection center.
- After the start position of the segment has been determined, the remainder of the segment can be specified. In this case of a straight segment, you can specify the 'Angle' field and the 'Length' field. Alternative, the button 'Position End' can be clicked. After that you have to click the mouse pointer somewhere in the 2D Drawing form.

It is important to keep a few conventions in mind:

- Segments have a direction: they go from START (StartX, StartY) to END (EndX, EndY). When a database is saved (File, Save As..) a segment in the opposite direction is created by the program.
- The START of a new segment can be connected to the START of another segment or to the END of another segment. However, all sequential segments (between two intersections or

connection nodes) MUST have the same direction: a START must connect to an END and an END must connect to a START. If you specify a segment by starting it on the START point of another segment, then after the segment has been completed, the program turns the segment around.

- Never connect 2 straight segments with different angles. Always put a curved segment in between or connect 2 straight segments with the same angle.
- The angle subtended by a curved segment can never be larger than 180 degrees. If you want a curve that spans, for example, 270 degrees, then first make a curve with 180 degrees and then add a curve with a subtended angle of 90 degrees.
- The program tries to make a segment based on as little information as is possible. As soon as there is enough information the program creates a 2-lane segment. For example, if you create a curved segment, and specify the start angle, the radius and the turn, then the program still needs either the length of the curve or the ArcAngle (the subtended angle). As soon as this is specified, a curved segment is drawn. There are however situations in which the user makes specifications of a curve that are impossible to implement. A notable example is the following: the user positions the start point of the curve at an end point of another segment. In that case the StartAngle is fixed. After that, the user clicks the <Position End> button and then clicks on an intersection of another segment selection node to position the end node of the segment. This will almost always result in a warning by the program and a refusal to create such a segment. The reason being that this specification is almost always impossible. If the start angle is determined you need a radius, a turn and an ArcAngle or length or you need a radius and an EndAngle. But specifying a point ensures that the equations cannot be solved unless a sequence of segments is created simultaneously. So always make sure that you do not use this method !

There are several ways by which you can create a new segment:

METHOD 1

- first create an intersection
- select the Segments form and press <Create New Seg>
- Press <Position Start>: the shape will be Straight by default
- click on the blue intersection hotspot: if you have clicked right, it becomes yellow with a thick black circle around it
- now you see that StartX and StartY have gained a value. Because you have connected the start of the segment to an intersection, the angle of the segment is undefined: this must be defined explicitly:
- Fill out the Angle field (for example 0, meaning that it will be a horizontal segment)
- Fill out the Length field (for example 100, meaning 100 meters)
- press <tab>
- now you see that a segment is created and added with 2 lanes. If you press the lanes tab then the lanes can be changed. A segments MUST have at least 1 lane (#Dlanes >= 1)

METHOD 2

- first create an intersection
- select the Segments form and press <Create New Seg>
- Press <Position Start>: the shape will be Straight by default
- click on the blue intersection hotspot: if you have clicked right, it becomes yellow with a thick black circle around it
- now you see that StartX and StartY have gained a value. Because you have connected the start of the segment to an intersection, the angle of the segment is undefined: this must be defined explicitly
- Press <Position END>
- Position the mouse cursor in the 2D Drawing form and left click to indicate where the end point of the segment must be.

- You can see now that Angle and Length is determined by the program and filled out in the respective fields.
- If desired, you can change the Angle or Length, as long as the end point is not connected to another segment, intersection or connection node.

METHOD 3

- first create a segment
- press <Create New Seg>
- Press <Position Start>
- click on the yellow hotspot of an endpoint of a segment: the hotspots of the selected segment get a thick black circle
- The StartX and StartY fields are filled out by the program
- Press <Position End>
- Position the mouse cursor in the 2D Drawing form and left click to indicate where the end point of the segment must be.
- You will see that a curved segment has been created. Radius has been determined, as well as the turn direction (left or right), the end angle of the segment and the arc angle. Start Angle has been fixed to the end angle of the segment this one has been connected to. This cannot be changed anymore.
- If desired you can change the radius, turn direction, end angle or arc angle, as long as the end point of this segment has not been connected to another segment, intersection or connection node
- If by this method, there is no mathematical possibility to create a segment, the program refuses the operation and gives a warning.

METHOD 4

- first create a segment
- press <Create New Seg>
- Select Shape <Straight>
- Press <Position Start>
- click on the yellow hotspot of an endpoint of a segment: the hotspots of the selected segment get a thick black circle
- The StartX and StartY fields are filled out by the program
- The Angle is fixed as the end angle of the segment this new segment is connected to
- Fill out the Length field (for example 100, meaning 100 meters)
- press <tab>
- now you see that a segment is created and added with 2 lanes.

METHOD 5

- first create a segment
- press <Create New Seg>
- Select Shape <Curve>
- Press <Position Start>
- click on the yellow hotspot of an endpoint of a segment: the hotspots of the selected segment get a thick black circle
- The StartX and StartY fields are filled out by the program
- Fill out Radius, Turn direction and Arc Angle
- press <tab>
- now you see that a curved segment is created and added with 2 lanes

There are more methods, especially related to connection nodes. Connection Nodes are not covered in this documentation because the use is quite advanced and requires more attention than can be given in this sparse documentation.

6.2 Lanes

The lanes tab gives access to the lane information. The following types of lanes are available:

- **Dlanes:** these are normal driving lanes. There must be at least 1 of these. Dlanes are sequentially ordered from right to left as DLane 0 .. DLane n-1. The center line through DLane 0 is the reference track for driving (lateral position = 0.0).
- **EntryRight:** These are EntryLanesRight Lanes on the right side of the segment (from Start to End). If you change the value from 0 to 1 a new EntryRight is created. An EntryRight lane is a lane you use to enter the road from right, for example on a highway (an entry lane) in a system where you drive on the right side of the road (as in continental Europe). An EntryRight lane often has a Transition Length: this is the length of the lane during which the lane becomes narrower from the lane width to 0 meters. In that case End Transition Length is for example 50 meters.
- **EntryLeft:** These are EntryLanesLeft Lanes on the left side of the segment (from Start to End). If you change the value from 0 to 1 a new EntryLeft is created. An EntryLeft lane is a lane you use to enter the road from left, for example on a highway (an entry lane) in a left-hand driving system. An EntryLeft lane often has a Transition Length: this is the length of the lane during which the lane becomes narrower from the lane width to 0 meters. In that case End Transition Length is for example 50 meters.
- **ExitRight:** These are ExitLanesRight Lanes on the right side of the segment (from Start to End). If you change the value from 0 to 1 a new ExitRight is created. An ExitRight lane is a lane you use to exit the road to right, for example on a highway (an exit lane), or in front of an intersection (to choose the correct lane). An ExitRight lane often has a Transition Length: this is the length of the lane during which the lane becomes wider from 0 meter to the ultimate lane width. In that case Start Transition Length is for example 50 meters.
- **ExitLeft:** These are ExitLanesLeft Lanes on the left side of the segment (from Start to End). If you change the value from 0 to 1 a new ExitLeft is created. An ExitLeft lane is a lane you use to exit the road to left, for example on a highway (an exit lane) in a left-hand driving system, or before an intersection where the driver needs to turn left. An ExitLeft lane often has a Transition Length: this is the length of the lane during which the lane becomes wider from 0 meter to the ultimate lane width. In that case Start Transition Length is for example 50 meters.
- **HardShoulderRight:** the hardshoulder on a highway. Should be 0 or 1.
- **HardShoulderLeft:** the hardshoulder on a highway in a left-hand driving system. Should be 0 or 1.
- **BicycleLaneRight:** a special lane for bicyclists, to the right of the Dlanes. Should be 0 or 1.
- **BicycleLaneLeft:** a special lane for bicyclists, to the left of the Dlanes. Should be 0 or 1.
- **SideWalkRight:** a pavement for pedestrians, to the right of the road. Should be 0 or 1.
- **SideWalkLeft:** a pavement for pedestrians, to the left of the road. Should be 0 or 1.

A lane can be selected by clicking on the round radiobutton right of the respective.

For each selected lane, the lane width can be specified. If you want to specify the edge line types you should specify it both for the left and the right side of the lane. There are also a number of other things you can specify for each lane.

6.3 Signs

The Signs tab gives access to the road signs information. Signs are added to a segment.

The resourcefile 'roadsigns.tex' contains all sign definitions. This file can be configured by the user. Each line contains a definition of a specific sign:


















- **symbol value** : The logical number represents the 'meaning' of the sign in the traffic system. The number is responsible for the way traffic responds to this particular sign.
- **NameOfSignInGUI** : The name of the sign as it is used in the GUI
- **ThumbnailName** : the texture filename of the corresponding thumbnail in the GUI (in the folder "thumbnails")
- **ModelTextureName** : the texture filename of the corresponding texture used in the 3D model (in the folder "textures\roadsigns")
- **BackGroundId** : each sign has a background that determines the shape of the sign. Logical symbol value numbers ≥ 1000 are sign backgrounds. This BackGroundId must be present as a sign in roadsigns.tex.
- **Width** : the Width of the sign in meters. By this you can make signs smaller or larger if the 3D database
- **Height** : the Height of the sign in meters. By this you can make signs smaller or larger if the 3D database





















Lines in 'roadsigns.tex' then look like this:





















2	A2-50	a1.gif	a1_50_rgba	1001	0.80	0.80
.....						
1001	bck_sqr	sgn_bck_sqr.gif	sgn_bck_sqr.rgb	1001	0.80	0.80



















The symbol value is for use in the traffic system. In the example, the value on the first line is 2. In the table below it can be seen that this is a '*speed limit 50 km/h*' sign. The textures in the GUI (thumbnails) correspond to the ThumbNailName in the 3rd column. The descriptions in the GUI (for example, B5 or C4-right) are read from NameOfSignInGUI in the 2nd column. The columns 4-7 contain data for the generation of the 3D databases.






















The following table shows the symbol values in relation to their meaning. It is very important that users adhere to this system, because the symbol value in the roadsigns.tex file is also used by the traffic system (in Traffic.exe) for the traffic to interpret the meaning of these traffic signs. In the table sometimes the pictogram should be a variation of the one depicted. In that case this is mentioned in the text (under Meaning).

Symbol value	Meaning	Pictogram
1	Speed maximum 30 km/h Pictogram should say 30 instead of 50	
2	Speed maximum of 50 km/h	
3	Speed maximum 70 km/h Pictogram should say 70 instead of 50	
4	Speed maximum 100 km/h Pictogram should say 100 instead of 50	
5	Speed maximum 120 km/h Pictogram should say 120 instead of 50	
6	End speed limit 50 km/h	
7	End speed limit 70 km/h Pictogram should say 70	
8	End speed limit 100 km/h Pictogram should say 100	
9	End speed limit 120 km/h Pictogram should say 120	
10	Advised speed 50 km/h	
11	Advised speed 70 km/h Pictogram should say 70 instead of 50	
12	Advised speed 100 km/h Pictogram should say 100 instead of 50	
13	Advised speed 120 km/h Pictogram should say 120	
14	End of Advised speed 50 km/h	
15	End of Advised speed 70 km/h Pictogram should say 70 instead of 50	
16	End of Advised speed 100 km/h Pictogram should say 100 instead of 50	
17	End of Advised speed 120 km/h Pictogram should say 120	
















18	Major road	
19	End of major road	
20	Priority crossing (left and right road)	
21	Priority crossing (only left road)	
22	Priority crossing (only right road)	
23	Give ROW to traffic on crossing road	
24	Stop and give ROW	
25	Road closed in both directions	
26	Road closed in this direction (one-way road)	
27	One-way road (entrance allowed from this direction)	
28	One-way road (entrance allowed from this direction) Arrow points to right	
127	One-way road (entrance allowed from this direction) Arrow points to left	
29	driving in allowed	
30	Road closed for vehicles (no entrance allowed)	
31	Road closed for trucks	
32	Road closed for vehicles that are slower than 25 km/h	
33	Similar as 32, but also closed for bicyclists and mopeds	
34	Road closed for vehicles with trailers	
35	Closed for motorcycles	
36	Closed for all motorvehicles	

37	Closed for mopeds	
38	Closed for bicycles	
39	Closed for mopeds and bicycles	
40	Closed for pedestrians	
41	Closed for vehicles longer than ... meters	
42	Closed for vehicles wider than ... meters	
43	Closed for vehicles higher than ... meters	
44	Closed for vehicles heavier than ... tons	
45	Closed for vehicles with a total weight larger than ... tons	
46	Closed for vehicles with dangerous chemicals	
47	Roundabout: compulsory driving direction	
48	Pass the sign on the side indicated by the arrow	
49	Sign may be passed on both directions	
50	Following the direction is compulsory	
51	Follow driving direction to right	
129	Follow driving direction to left (Pictogram should show arrow to left)	
52	Follow driving direction to right or straight on	
128	Follow driving direction to left or straight on (Pictogram should show arrow to left)	
53	Follow driving direction to right or left	
54	Parking prohibition	

55	Stopping prohibition	
56	Prohibition to park bicycles or mopeds	
57	Parking	
58	Taxi parking	
59	Parking for disabled	
60	Facility for loading and unloading	
61	Parking, only for cars	
62	Parking for licence-holders	
63	Zone for parking disc	
64	End of parking disc zone	
65	Parking for transfer to public transportation	
66	Parking for carpooling	
67	Overtaking prohibition	
68	End of overtaking prohibition	
69	Overtaking prohibition for trucks	
70	End of overtaking prohibition for trucks	
71	Prohibition to drive on in case of traffic from oncoming direction	
72	Traffic from oncoming direction should give precedence to traffic from this direction	

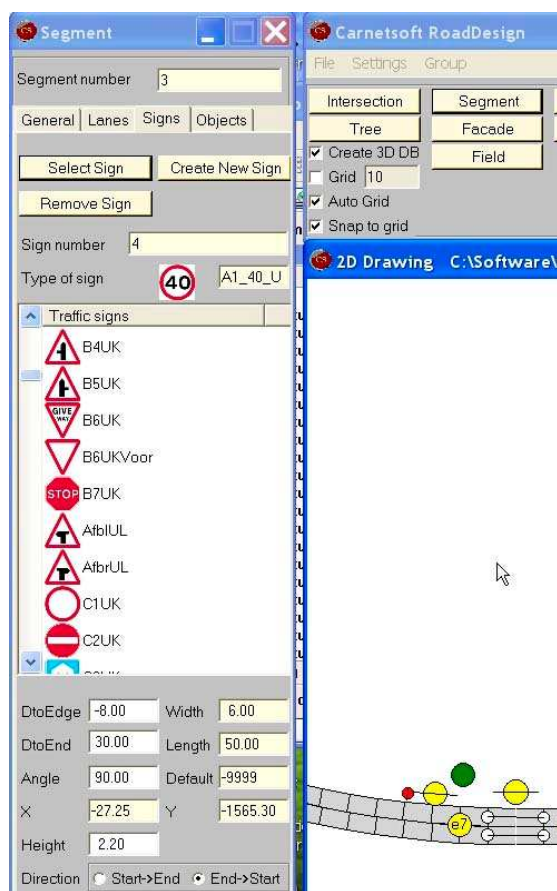
73	Turning prohibition	
74	End of all prohibitions (indicated by traffic signs)	
75	Stop	
76	Highway	
77	End of highway	
78	Motorway	
79	End of motorway	
80	Residential area (slow down traffic)	
81	End of residential area	
82	Pedestrian path	
83	End of pedestrian path	
84	Bridle path	
85	End of bridle path	
86	Obligatory bicycle path	
87	End of obligatory bicycle path	
88	Combined bicycle/moped path	
89	End of combined bicycle/moped path	
90	Non-obligatory bicycle path	
91	End of non-obligatory bicycle path	
92	Built-up area	
93	End of built-up area	

94	Bad road surface	
95	Curve to right	
96	Curve to left	
97	S-curve(s), first to right	
98	S-curve(s), first to left	
99	Dangerous crossing	
100	Roundabout	
101	Work in progress	
102	Narrowing of road	
103	Narrowing of road	
104	Narrowing of road	
105	Slippery road	
106	Children	
107	Zebra crossing	
108	Pedestrians	
109	Bicycles and mopeds	
110	Oncoming traffic	
111	Traffic lights	
112	Traffic jam	
113	Danger	 KOFFIE

114	Pedestrian crossing		
115	Bus stop		
116	Get in lane (LeftStraight-Straight-Right)		
130	Get in lane (LeftStraight-Right)		
117	End of driving lane		
118	Branching of road (fork)		
119	Number of ongoing lanes		
120	Deadend road		
121	Advance notice deadend road		
123	Roundabout: compulsory driving direction		
124	Undersign : except bicycles		
125	Undersign: Branching off major road		
126	Undersign: Branching off major road		
131	Undersign: Branching off major road		
132	Undersign: Branching off major road		

To add a sign, you first select a segment.

- Then you press <Create New Sign>.
- Select a Traffic sign from the list by clicking on it.
- change DtoEdge: this is the distance from the right road edge: a positive value shifts the sign to the right, and a negative value shifts the sign to the left. The field <Width> gives the road width as a reference.
- change DtoEnd: this is the distance from the end of the segment. The field <Length> gives the segment Length as a reference.
- The angle is calculated by the program so that the face of the sign is perpendicular to the road at the current position. But this can be changed manually.
- Height sets the height of the pole of the sign.
- The sign is directed at the direction from START to END. If you want it to be seen from the other way, Direction must be set to End->Start.

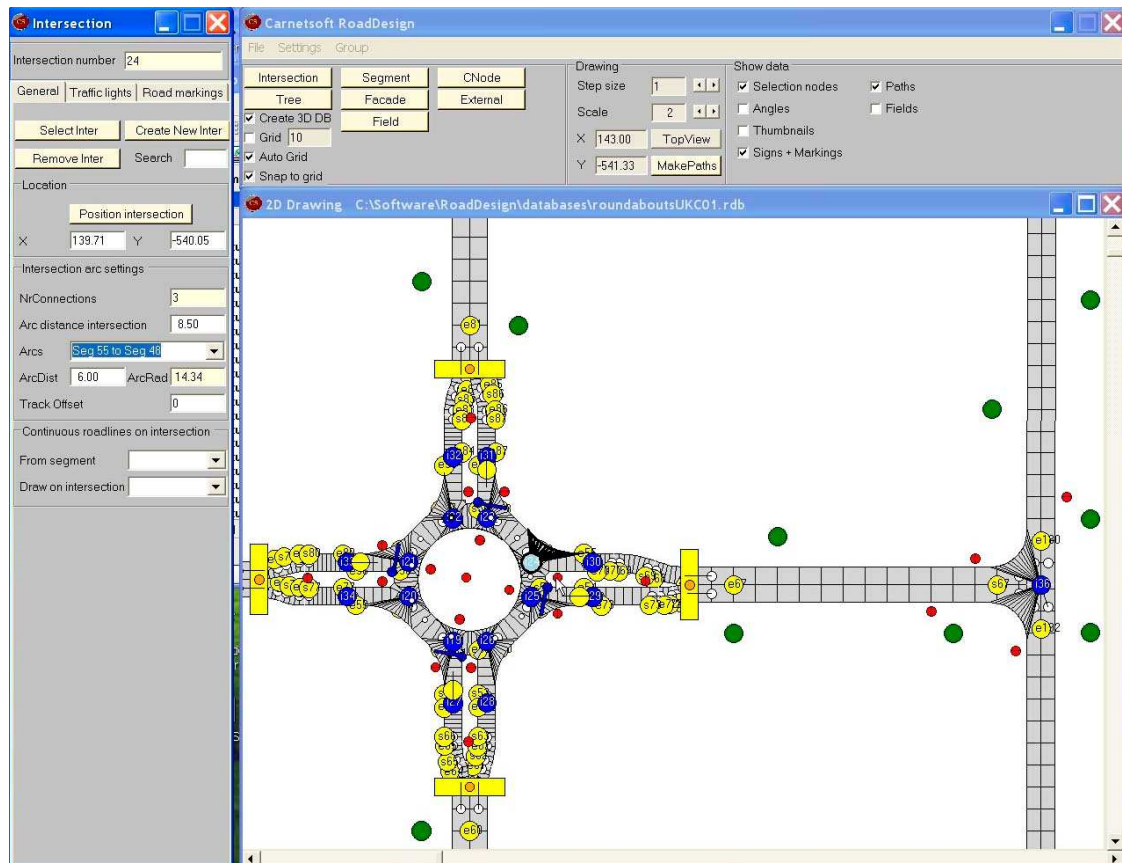


7 Intersections

To create a new intersection do the following:

- In the central-top, click on 'Intersections'. Then the 'Intersection panel' shows up on the left side of the screen
- Now you can either create a new intersection or select an existing intersection. If you want to create a new intersection, press the button 'Create New Inter'. Then the 'Intersection number' field displays the value '-1'.
- Press <Position Intersection>
- click with the left mouse button in the 2D Drawing form. A blue intersection hotspot appears, and the intersection is created.
- If the intersection is not connected to another object, you can change the X and Y fields by entering a number and pressing the <tab> key.

The following figure shows an intersection (black circle) that has been selected.



It is important to keep a few conventions in mind:

- You can connect any number of segments to an intersection, but 3 or 4 segments is more standard. If you want to reverse the order of segments (for example, connect a segment START->END to segment END->START), you can put an intersection in between. This is not a real intersection, but a 'virtual' intersection, with only 2 connected segments.
- You can connect straight segments and curved segments to an intersection. However, it is important to keep in mind that you should always connect at least 1 straight segment to an intersection.
- Make sure that segments connected to an intersection have different incoming angles.

There are several ways by which you can create a new intersection:

METHOD 1

It is possible to generate intersections from two overlapping segments. If you create a segment (curve of straight) it is tested whether the segment crosses any other segments. All points where the new segment crosses another segment are then indicated in the 2D design window as a white square. The user can then create an intersection on that point by :

- Create New Inter
- Position Intersection
- Click with the mouse button on the white square

An intersection is then created on that point and all connecting segments are 'added' to the intersection.

This works with the following combinations of crossing segments:

- STRAIGHT vs STRAIGHT

So this method is not allowed for crossings with a curved segment.

METHOD 2

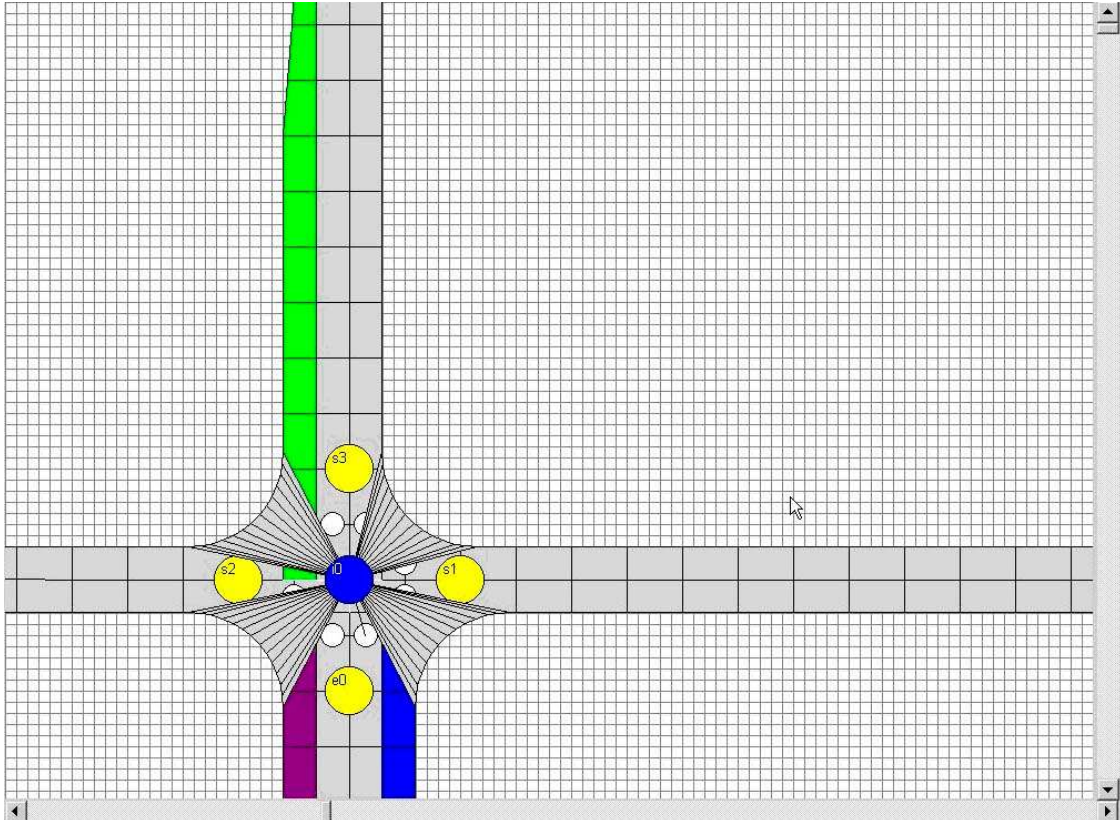
- Press <Create New Inter>
- Fill out X and Y (for example X 100 Y 100) and enter the <tab> key
- If you press now the TopView button, you see that a new intersection has been created

METHOD 3

- Press <Create New Inter>
- Press <Position intersection>
- position the mouse over the 2D Drawing area and press the left mouse button.
- An intersection is now created. If you have clicked on a yellow hotspot of a segment, the intersection is connected to that segment, and you see that the intersection is placed at the end or start of the selected segment.

To connect segments to intersections, you must click on the intersection while you create the segment. For example, if you create a segment you can <Position Start> on the intersection. Then the start of the segment is connected to the intersection. According to the same principle you can connect the end of a segment to an intersection (<Position End> on an intersection).

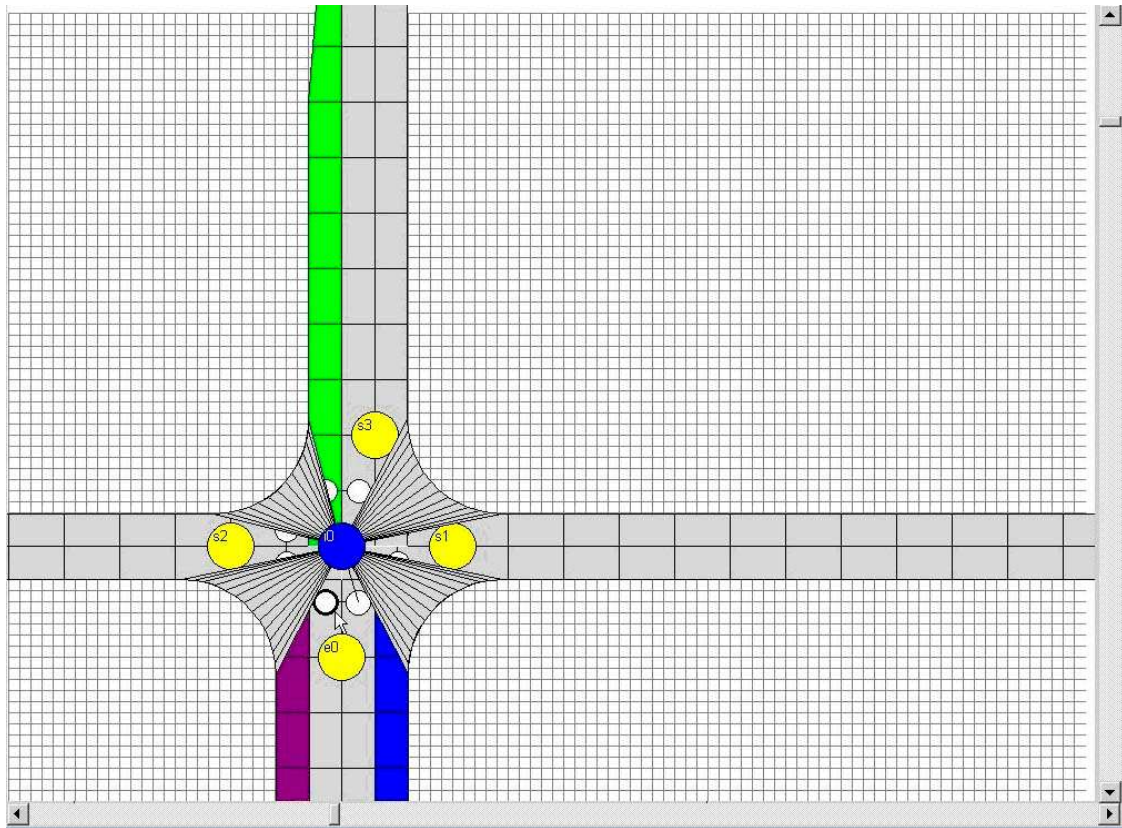
The way the segments connect to an intersection can be controlled by the user. Normally, if you connect a segment to an intersection, the leftpoint of Lane 0 is attached to the intersection. Although this often is precisely what you want, in some cases you may want a different intersection geometry. Changing the geometry of an intersection in this way can be accomplished by connecting the start or end of a lane that connects to the intersection to a different lane of another segment that connects to that intersection. This is explained in the following example. Suppose you create a road with an ExitLaneLeft and an ExitLaneRight. This will ofcourse be a one-way road in which traffic can go either straight on, turn to the left via a separate lane or turn to right via a separate lane at the intersection. This is the road from south to north in the example. Then an intersection is created. After that the segments 1 and 2 are created. Nothing special so far. But now you want to construct a road (from south to north) with an EntryLaneLeft. Ofcourse this will also be a oneway road. If you create this from intersection 0 up, then the intersection looks like the following figure.



But you want traffic from segment 0 on the right DLane (DLane 0) to proceed on DLane 1 of Segment 3 if traffic drives straight on at intersection 0. So the lanes on segment 3 have to shift to right. To do this, follow the next steps:

- Go to the <Segments> panel and select Segment 3
- Go to <Lanes>
- Click on EntryLeft 0 and enable the round selection button next to it (in the upper part of the Lanes panel)
- Then click on the <Position Start (Select other lane end)> button
- Click on the DLane 1 selection node (the white round lane selection node in the 2D Drawing field) of Segment 0 (that is, the lower south to north segment with the purple and blue lane)
- Now Segment 3 will shift one lane to the right and the intersection geometry is changed, see the next figure.

In order for a segment to shift it **MUST** be unconnected at the other side. SO, this only works if the end of Segment 3 is not connected to another segment, intersection of Cnode.



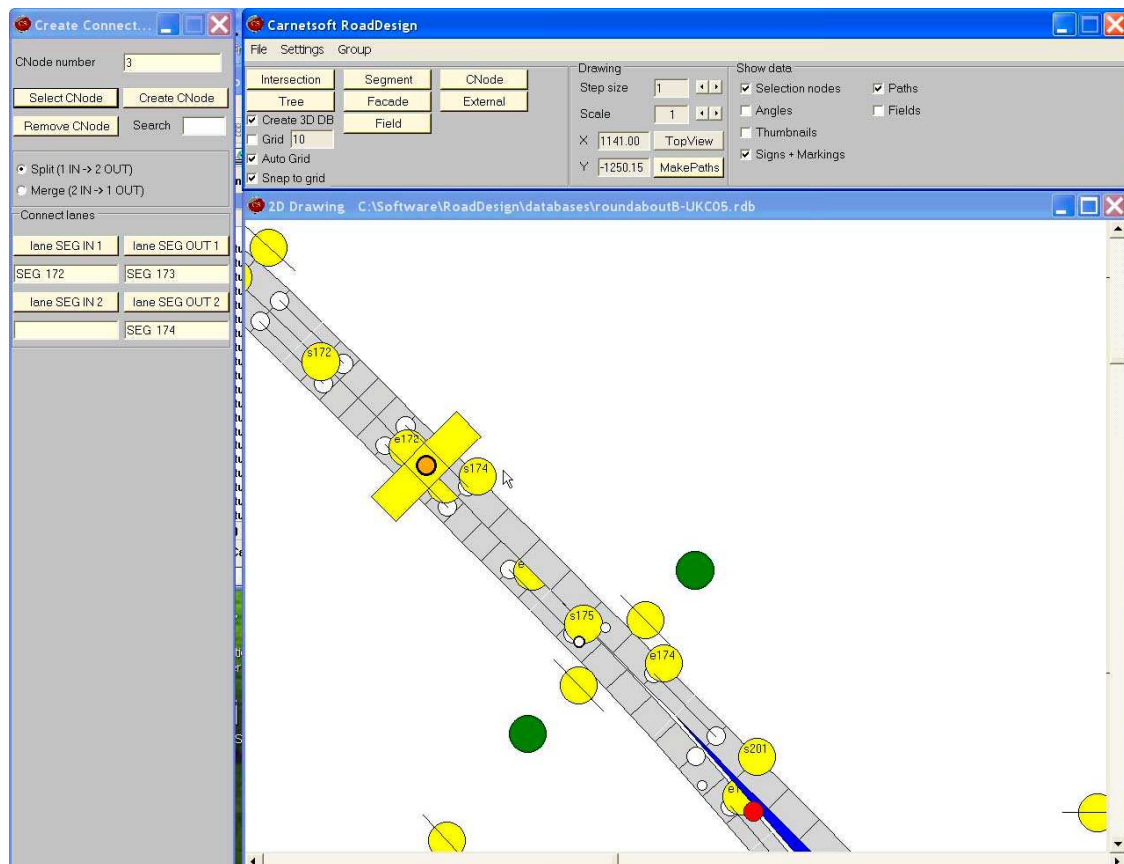
A second way of altering the geometry of an intersection is as follows. Between each incoming and outgoing segment there is an arc on the intersection. If an intersection has been created and if there are more than 2 connecting segments, the system determines the arcs (indicated as gray radiating lines in the 2D drawing window). In the <Intersection> <General> panel, if you click on the Arcs selection box, you can select a specific arc (the arc that goes from Segment A to segment B). If you have selected an arc, this will be drawn in black with thick lines. Now you can change the ArcDist by entering another value followed by a <tab>. ArcDist is the distance from the intersection of the right edges of the two segments where the arc will start. By default this value is 8.5 meters. If you change it to, for example, 4 then the selected arc will be smaller.

Traffic lights and Roadmarkings will be explained in a later version of the documentation.

8 Connection Nodes

Cnodes connect the lanes of incoming and outgoing segments. The total number of incoming lanes MUST match the total number of outgoing lanes.

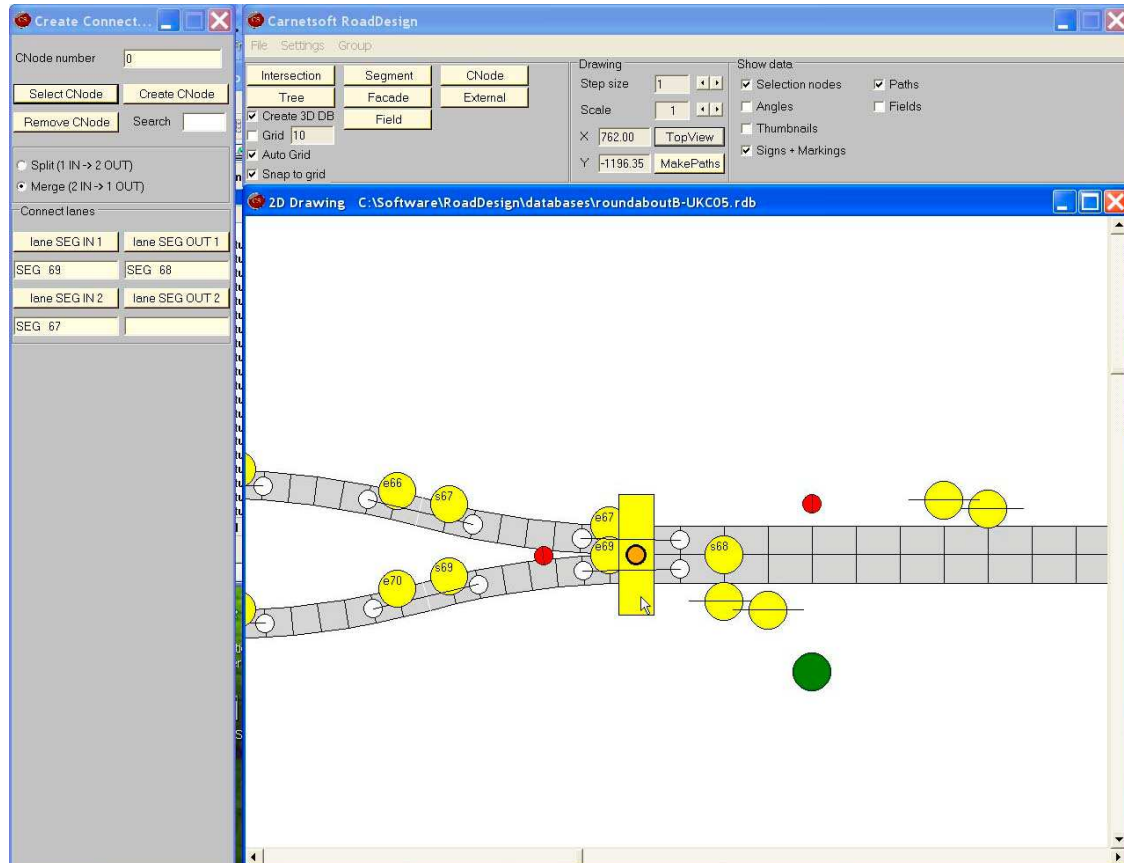
Cnodes come in two versions: Split nodes and Merge nodes. A split node splits one IN segment in two different OUT segments. An example of a split node is an exit on a highway. The End angle of the IN segment must be equal to the Start angles of the OUT Segments. The following figure gives an example of a split node.



As can be seen, a connection node is graphically indicated as a yellow square box. Black lines indicate how lanes are connected. In this example Segment 172 has two DLanes. Segment 172 is the IN segment that is split into two different OUT segments: Segment 173 and Segment 174. Both OUT segments have 1 DLane each. DLane 0 of segment 172 is connected to DLane 0 of Segment 173. DLane 1 of segment 172 is connected to DLane 0 of Segment 174. Note that there are no overlapping lanes: Overlapping of lanes is not allowed in Cnodes. Also note that the END node of an IN segment is connected to the Cnode, while the START node of the OUT segments must be connected to the Cnode. So, in summary:

- a split node has 1 IN and 2 OUT segments
- the number of lanes of the IN segment must equal the sum of the number of lanes of the 2 OUT segments
- each IN lane connects to a different OUT lane: an IN lane connects to precisely 1 OUT lane and vice versa (lanes must not overlap)
- the IN segment connects with its END point to the Cnode
- the OUT segments connect with the START point to the Cnode

A merge node merges two IN segments into one OUT segment. An example of a merge node is an entry on a highway. The End angle of the IN segments must be equal to the Start angle of the OUT Segment. The following figure gives an example of a merge node.



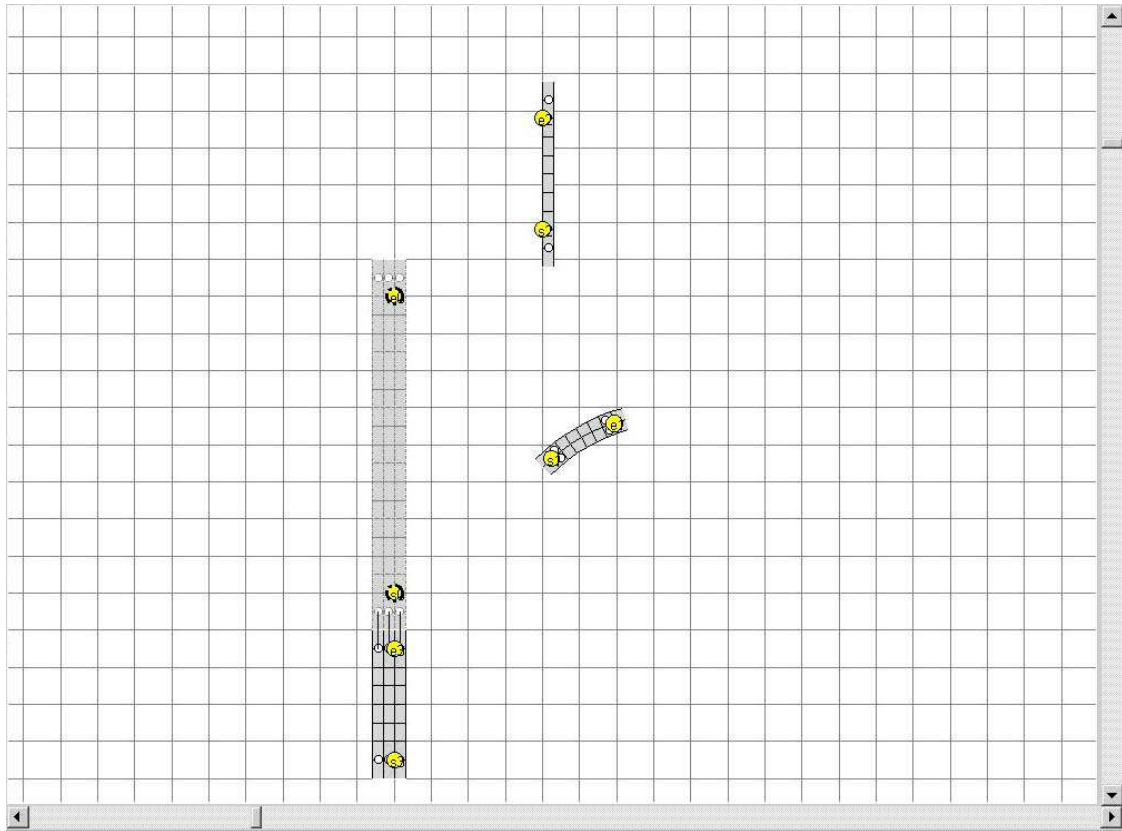
In this example Segment 67 has one DLane. Segment 67 is the IN segment that is merges together with Segment 69 into Segment 68. Both In segments have 1 DLane each. DLane 0 of segment 67 is connected to DLane 1 of Segment 68. DLane 0 of segment 69 is connected to DLane 0 of Segment 68. So, in summary:

- a merge node has 2 IN segments and 1 OUT segment
- the sum of the number of lanes of the two IN segment must equal the number of lanes of the OUT segment
- each IN lane connects to a different OUT lane: an IN lane connects to precisely 1 OUT lane and vice versa (lanes must not overlap)
- the IN segments connects with its END point to the Cnode
- the OUT segment connects with the START point to the Cnode

To create a new Cnode, do the following:

- a) In the central-top buttons, click on 'CNode'. Then the 'Connection node panel' shows up on the left side of the screen
- b) Now you can either create a new CNode or select an existing one. If you want to create a new CNode, press the button 'Create CNode'. Then the 'CNode number' field displays the value '-1'.
- c) First decide whether you want to create a Split of a Merge node, and click the appropriate selection button

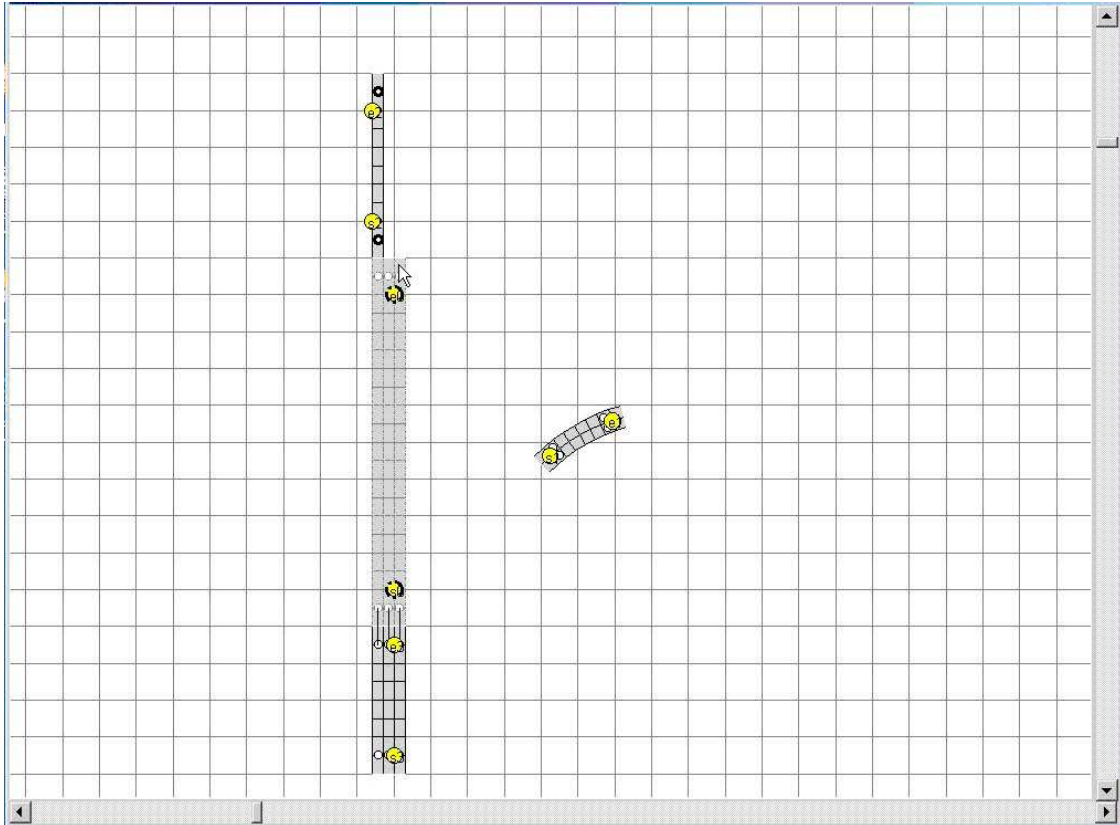
- d) Make sure that all segments have already been created and that all lanes have been determined. You cannot change the number and types of lanes of segments AFTER you have created a Cnode. So, there must be three segments ready for constructing the Cnode. If they have not yet been created, create the segments now.
- e) Press the <Create Cnode> button. The situation is now as in the next figure, in which a Split will be created.



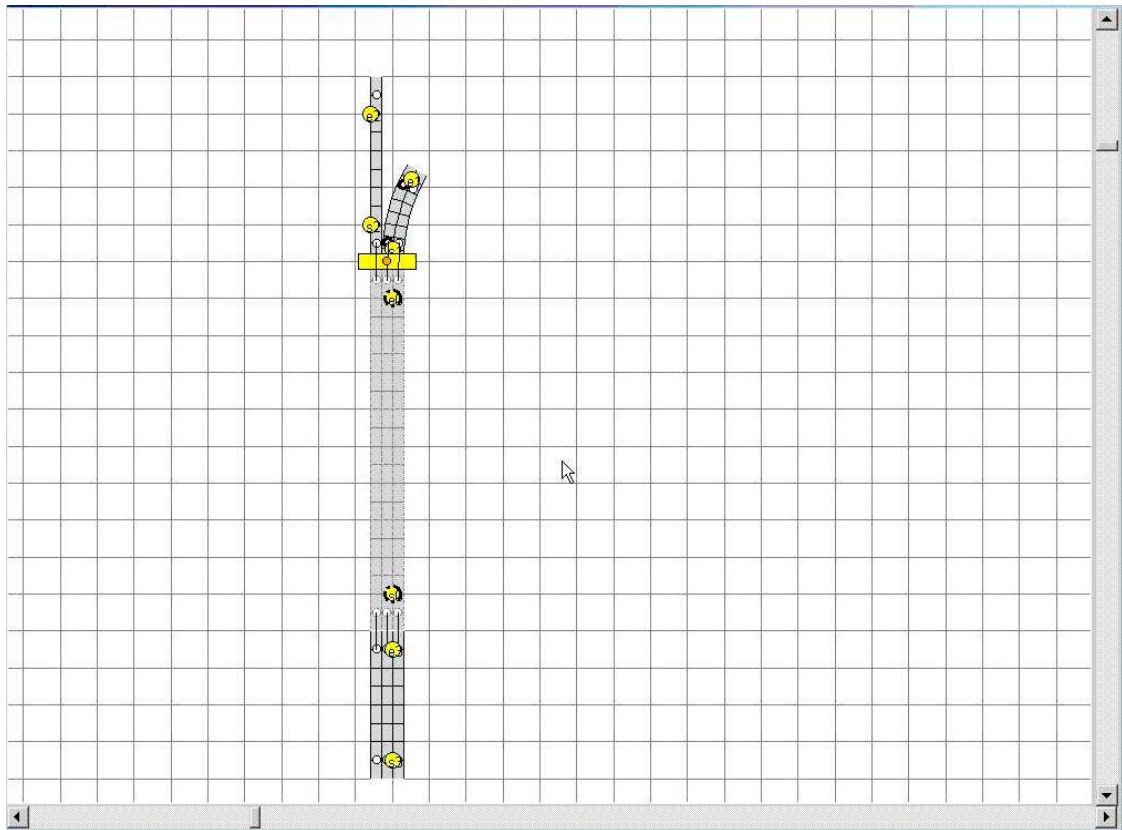
We want to create a Cnode with the three-lane Segment 0 as the IN segment and the straight one-lane segment 2 and the curved two-lane segment 1 as the OUT segments. If you connect the lanes, then one of the segments will be moved and rotated by the program so that it fits to the other segment. A segment can only be moved and rotated if it is not already connected to another segment, intersection or Cnode. So, one of the segments must be fixed. In the present example Segment 0 is fixed, because it has been connected to Segment 3. So, Segment 0 can not move. However, Segment 2 and three are free segments and they can be moved and rotated by the program.

Now, we first want to connect segment 2 to the leftmost lane of segment 0. Then do the following steps:

- f) Click the <lane SEG IN 1 > button
- g) click the leftmost lane selection node (white round node) at the end of Segment 0
- h) Then click the <lane SEG OUT 1> button
- i) Then click the lane selection node (white round node) at the start of Segment 2. Now Segment 2 is moved so that the start of the selected lane of Segment 2 connects to the end of the selected lane of Segment 0, see the following figure.



- j) Click the <lane SEG IN 2 > button
- k) click the middle lane selection node (white round node) at the end of Segment 0
- l) Then click the <lane SEG OUT 2> button
- m) Then click the leftmost lane selection node (white round node) at the start of Segment 1. Now Segment 1 is moved and rotated so that the start of the selected lane of Segment 1 connects to the end of the selected lane of Segment 0. Now the program has all information to create a Cnode and the Cnode is created, see the following figure.



Merge nodes are created in a similar way.

9 Graphical elements

The following types of graphical elements can be added:

Fields	RED selection nodes
Trees	GREEN selection nodes
Facades	GRAY selection nodes
External objects	YELLOW selection nodes

The user may add specific textures to the appropriate resource files in order to create the required elements in the database. The resource files have the extension '.tex', and must be located in the same folder as RoadDesign. All thumbnail textures for the GUI are loaded at start of the program. If any thumbnail file, to which a reference is made in the corresponding .tex file, is missing from the 'thumbnails\' folder, an error message box signals this error. The box can then be clicked away by the user, and the file 'RDStartup.log' contains messages of all missing textures.

9.1 Fields (LandPoly)

Fields are used to texture the environment with grass, sand or other ground-related textures. In the *.egg databases, they are referred to as "LandPoly". A Field is a polygon that contains any number of vertices. A texture is added to the polygon. The user can extend the designer with as many textures as he likes. These textures have to be put in the appropriate folders:

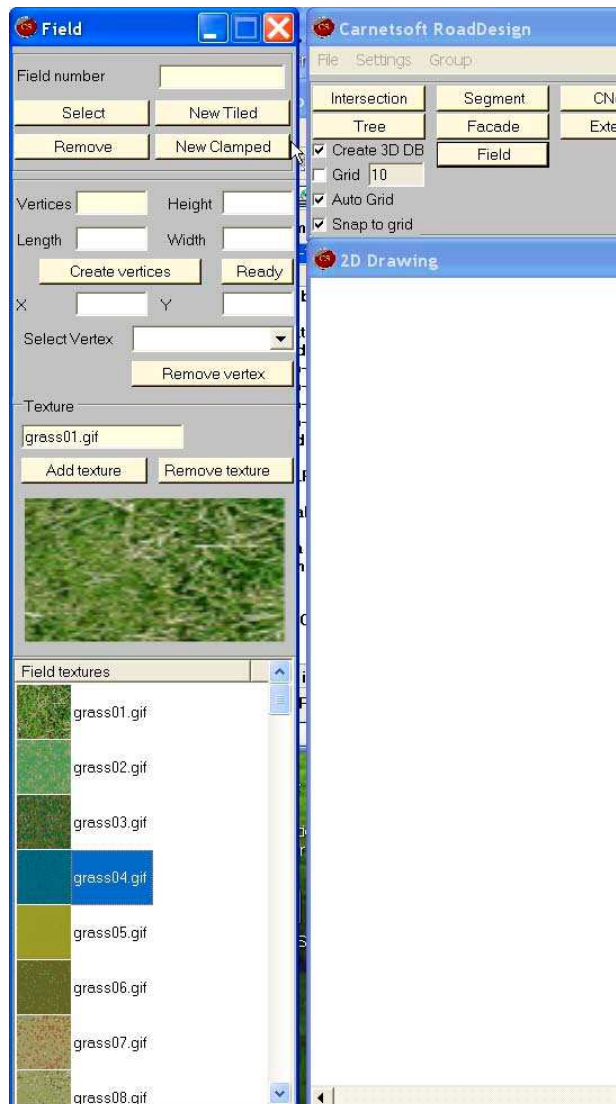
- "thumbnails\' for the thumbnail textures and
- "models\textures\fields" in the DriveSim3 folder for the textures as they are applied in the 3D databases. These must be *.jpg textures.

Then, the texture descriptions have to be added to the resource file. All The textures for all types of fields are defined in a resourcefile: 'fields.tex'. Each line contains a definition of a specific field texture:

- **ThumbnailName** : the texture filename of the corresponding thumbnail in the GUI (in the folder "thumbnails")
- **ModelTextureName** : the texture filename of the corresponding texture used in the 3D model (in the folder "textures\fields"): must be a jpg file (file extensions in ModelTextureName are ignored).
- **AspectRatio** : the ratio texture_width/texture_length for the modeltexture as it originally was. For the 3D models all model texture dimensions (width and length) have to be powers of 2 (for example 512x512). That disturbs the proportions of the texture in the 3D model, and in order to correct for that, the aspect ratio of the original texture has to be entered.
- **ScaleFactor** : the scalefactor of the texture. A scalefactor of 1 means that the texture width corresponds to 1 meter. If you make that 10, the texture is stretched to 10 meters per texture. Higher ScaleFactors improve the tiling of field textures, especially if the texture was intended for smaller areas.

Lines in 'fields.tex' then look like this:

plants2.gif	plants2.rgb	1.0	10.0
StoneStreak.gif	StoneStreak.rgb	1.0	10.0
etc.			



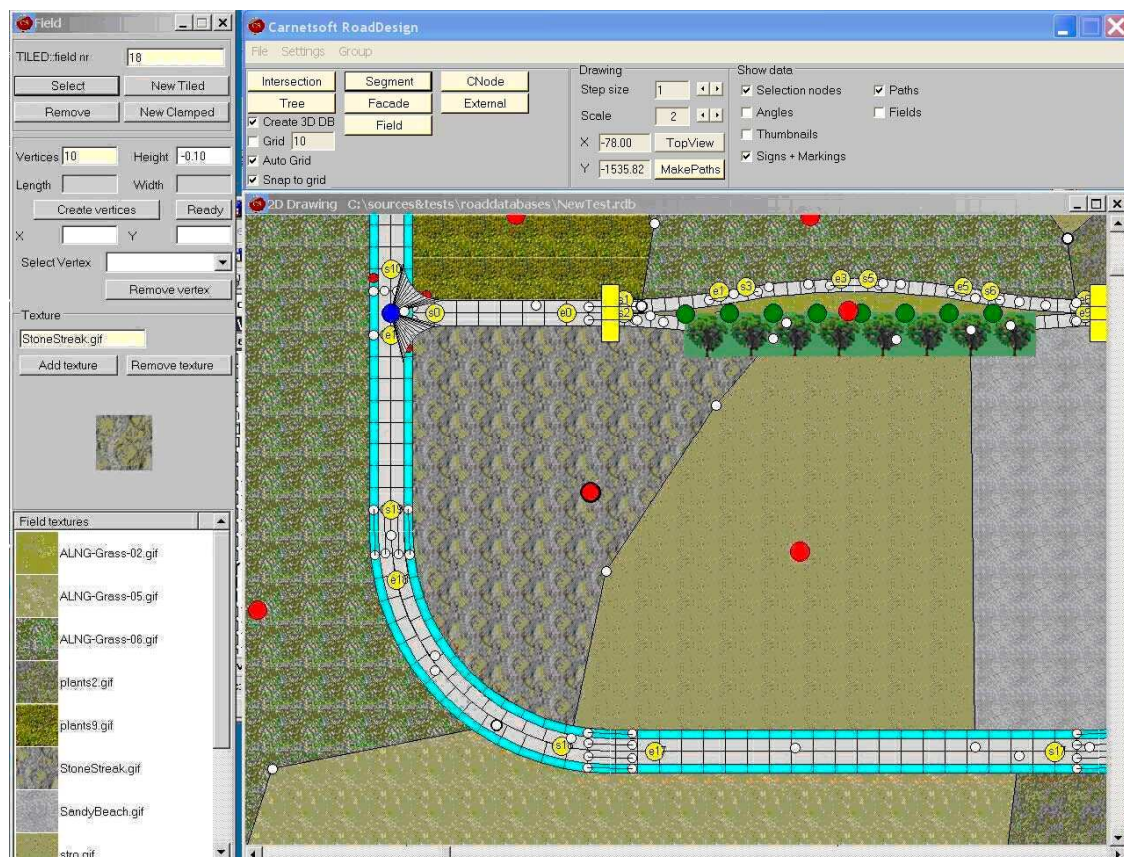
In the GUI you can see the list of textures as defined in 'fields.tex'. The thumbnails (small images of the textures) correspond to the first column. The name of the thumbnail texture is shown next to the thumbnail. The ModelTextureName, AspectRatio and ScaleFactor are solely used for generating the 3D graphical databases. If you are not satisfied with how a texture looks in the 3D database, simply change the name of the texture or add a better texture with the same name or change the ScaleFactor in the resource file. Then load the '*.rdb' database in the RoadDesign designer and Save to a new 3D database. The thumbnail texture names are included in the *.net and *.rdb databases. If you have changed the name of a thumbnail texture, then the designer cannot draw the texture. For drawing the textures in the designer, the thumbnail textures are used. In that case you see a green untextured field. You can then click on the RED hotspot of the texture, click on a texture and add the texture.

It is important to know that there are 2 types of textured fields:

- Tiled fields
- Clamped fields

These two are very different:

- Tiled fields are created by pressing the 'New Tiled' buttons. Tiled fields are the 'normal' landscape polygons that can have any number of vertices. The textures are tiled in the 3D database. The default height is -0.10 : 10 cm below the surface. This can be changed manually to create overlapping textures. In the designer, all textures are sorted on height before drawing: the lowest textures are drawn first and the higher textures are drawn after that. Lower textures are then masked by higher textures. However, the red hotspots are visible always for all fields, so you can select a field and temporarily give it a higher height.



In the example snapshot, a number of tiled fields have been designed. In the centre of each field is a red hotspot. Tiled field nr. 18 has been selected, and it has the texture 'StoneStreak.gif' on it. This field has 10 vertices. Each vertex is indicated by a small white circle. The 'Selected Vertex' combobox can be used to see all vertices and select (and change) each of them individually.

- Clamped fields, in contrast, always are rectangular: they always have 4 vertices, and are characterized by a Length and a Width (both in meters). The texture is clamped to the surface. If you create a Clamped field (by pressing the button 'New Clamped'), you first define where it is (X, Y) coordinates, and then add the appropriate texture. Now the texture has a certain Width and Length (a fixed aspectratio of the thumbnail texture). The program sorts this out and defines and fills out the Length and the Width (in meters) based on the texture dimensions. If you change any of these, the other will be changed automatically so that this aspect ratio is maintained. By this mechanism, you can scale the texture to the appropriate dimensions. Clamped fields can be used as a background on which you can design your roadnet, or you can use it to add a whole landscape to which you add other elements. For example, you can use a roadmap as a background texture. The default height is -0.20 , that is 20 cm below the surface. This can be changed manually.

To create a *Tiled* field, do the following steps:

- Press the tab page 'Fields'
- Press button <New Tiled>
- Press the button <Create vertices>
- click the vertex points with the mouse in the 2D Drawing form in a counterclockwise way. If the last point is identical with the first point the polygon is closed, and becomes a green field. Alternatively, you can press the 'Ready' button. In that case the last created vertex is connected to the first vertex.
- select a texture by clicking on the required texture in the selection list.
- Press <Add texture>

A field may be selected as follows:

- Press <Select>
- click in the red hotspot of a field.
- now the texture may be changed or vertices may be changed or removed if the field is Tiled. For Clamped fields the centerpoint may be changed or the dimensions may be changed.

A Field can be removed after it has been selected.

9.2 Trees (TreePoly)

Trees are billboards: trees or bushes with a 3D appearance, although the use is not restricted to trees and bushes. In the *.egg databases, they are referred to as "TreePoly". A tree is a vertical panel containing a texture, normally with alpha channel. A tree can be of two types: Billboard or Fixed. In the 3D Database, a billboard is represented as a panel that rotates with the position of the driver. For trees that are never very close to the viewer, Fixed trees can be used. These are panels that don't rotate with the viewpoint of the driver.

A billboard has a scaled height and width and has a fixed aspect (width/height) ratio. This width/height ratio is determined by the size of the original texture and is defined in the resource file 'trees.tex'. All textures for trees are defined in resource files.

Each line contains a definition of a specific tree texture:

- **ThumbnailName** : the texture filename of the corresponding thumbnail in the GUI (in the folder "thumbnails")
- **ModelTextureName** : the texture filename of the corresponding texture used in the 3D model (in the folder "models\textures\trees" with DriveSim3). All textures must be *.png (the *.rgb extension in trees.tex is ignored) as transparent textures.
- **AspectRatio** : the ratio texture_width/texture_height for the model texture as it originally was. For the 3D models all model texture dimensions (width and length) have to be powers of 2, typically 256x256. That disturbs the proportions of the texture in the 3D model, and in order to correct for that, the aspect ratio of the original texture has to be entered.

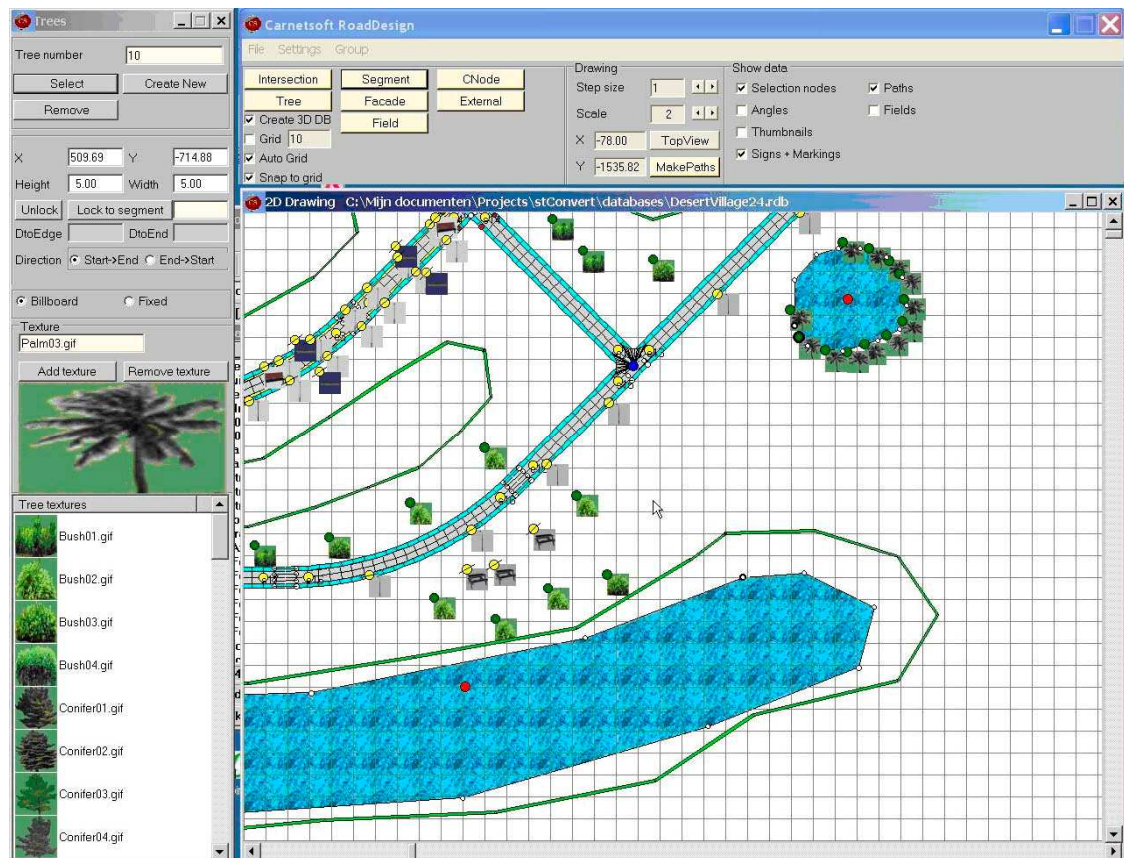
Lines in 'trees.tex' then look like this:

```
S_Deciduous01.gif      S_Deciduous01.rgb      0.75
S_Deciduous02.gif      S_Deciduous02.rgb      0.60
etc.
```

The *.png files are typically 256x256 pixels. The larger they are, the more detailed they are, but they also use more texture memory on the graphics board, especially if there are hundreds of them in a database.

Trees can be created as a row or as a single tree. A row of trees is always positioned along a segment. To create a row of trees, do the following steps:

- 1) first press <Create New>,
- 2) then press the <Row> tab.
- 3) Press the <Select Segment> button
- 4) Then select a segment by clicking in the Start or End selection node. The segment number is shown next to the select segment button.
- 5) Now you see a row of green circles that indicate the position of the trees. You now have to fill out the details of this row:
- 6) Always select a tree type by clicking on a tree texture in the tree list, an then press the <Add Texture> button
- 7) After that, always fill out the 'Height' field by entering a number (for example 12.0 to indicate 12 meters), followed by a tab. The Width is now filled in by the program (so that it matches the height/width ration of the respective tree type).
- 8) Then change the distance from the road edge (DtoEdge) or the distance between the trees (Between Distance). If one or more tree turn out to be on a wrong place, then select that tree (<Select>, <Single>, click on a green selection node), and Remove the tree.



The row information is contained with the segment. You can create as many rows along a segments as required. If you want to change a few trees in a row, then select the tree that is part of a row, by clicking on the <Select> button, then the <Single> tab, and then select an individual tree by clicking on the selection node. Then change the type of tree, height etc. So if a row is

selected, all trees in the row are changed after the user enters a change. If on the other hand a single tree is selected, only that single tree is modified.

In the example, a tree has been selected, by clicking on the GREEN hotspot. The tree is a billboard, the height is 5 meters, and the texture is 'Palm03.gif'. A tree can be positioned and after that, be attached to a segment. In this way it is easy to align trees at a precise position relative to the road:

- Position by clicking somewhere in the 2D drawing window, and then optionally changing the X and Y fields
- After that, you may lock the tree to a segment by:
 - press <Lock to Segment>
 - then click on a yellow segment selection node
 - The distance from the right edge of the segment is now computed (DtoEdge) together with the distance from the end of the segment (DtoEnd).
 - These values may be changed by the user by changing the DtoEdge and DtoEnd field values

Distances are now with respect to the right road edge, measured from the start selection node to the end selection node (Start->End). If you press the End->Start radio button then the Tree is positioned on the other side of the road.

By this procedure it is easy to add a row of trees alongside a road at, say, 2 meters from the road edge and precisely 30 meters apart (for example first tree at 30 meters DtoEnd, second tree at 60 meters DtoEnd etc.)

To create a single Tree, do the following steps:

- Press the tab page 'Trees'
- Press button <Create New>
- Press the <Single> tab
- click with the mouse in the 2D Drawing form to set the position of the tree, or alternatively, fill out the X and Y text boxes. If required, press the <Lock to Segment> button to lock the position of the tree to a segment.
- select a texture by clicking on the required texture in the selection list.
- Press <Add texture>
- The program knows the aspect ratio of the texture (because it was defined in trees.tex). If you change the height, the width is changed by the program and vice versa to match the aspect ratio.

A Tree may be selected as follows:

- Press <Select>
- if you want to select and change the properties of a whole row, then press the <Row> tab. Otherwise press the <Single> tab
- click in the green hotspot of a Tree.
- now the texture or the dimensions may be changed. If you have selected a row of trees, all trees will change if you apply a different texture, height etc.

9.3 Facades (FacePoly)

Facades can be used as graphical panels containing textures with (usually) an alpha channel, that block the horizon. In the *.egg databases, they are referred to as "FacePoly". They can also be used as fences, woods etc. Facades can then be tree lines, walls, fences, fronts of buildings (in that case no alpha channel would be required), or hills. A façade is a sequence of vertical panels with fixed orientations. A façade can consist of any number of connected panels that are indicated by points. The height can be set by the user. Facades don't have a fixed aspect ratio, The aspect ratio that is used in the file is for the 3D conversion software only. So, for a given size of the façade, any height can be specified.

A façade can be of two types: 'Repeat' or 'Clamped'.

- a 'Repeat' façade is characterized by horizontal tiling of the texture on the panels in the 3D database.
- a 'Clamp' façade is characterized by a stretched texture to fit the dimensions of the façade in the 3D database. A Clamp façade would normally have only 2 points. You can glue a housefront on that, for example.

It contains a texture with alpha channel (not required, but usual). All textures for facades are defined in the resource file 'facades.tex'. Each line contains a definition of a specific facade texture:

- **ThumbnailName** : the texture filename of the corresponding thumbnail in the GUI (in the folder "thumbnails")
- **ModelTextureName** : the texture filename of the corresponding texture used in the 3D model (in the folder "models\textures\facades" within the DriveSim3 folder). The extension has to be *.png. The extension in trees.tex (for example .rgb, is ignored).
- **AspectRatio** : the ratio $\text{texture_width}/\text{texture_length}$ for the modeltexture as it originally was. For the 3D models all model texture dimensions (width and length) have to be powers of 2. That disturbs the proportions of the texture in the 3D model, and in order to correct for that, the aspect ratio of the original texture has to be entered.

Lines in 'facades.tex' then look like this:

```
TreeLine01.gif      TreeLine01.rgb      30
TreeLine02.gif      TreeLine02.rgb      20
etc.
```

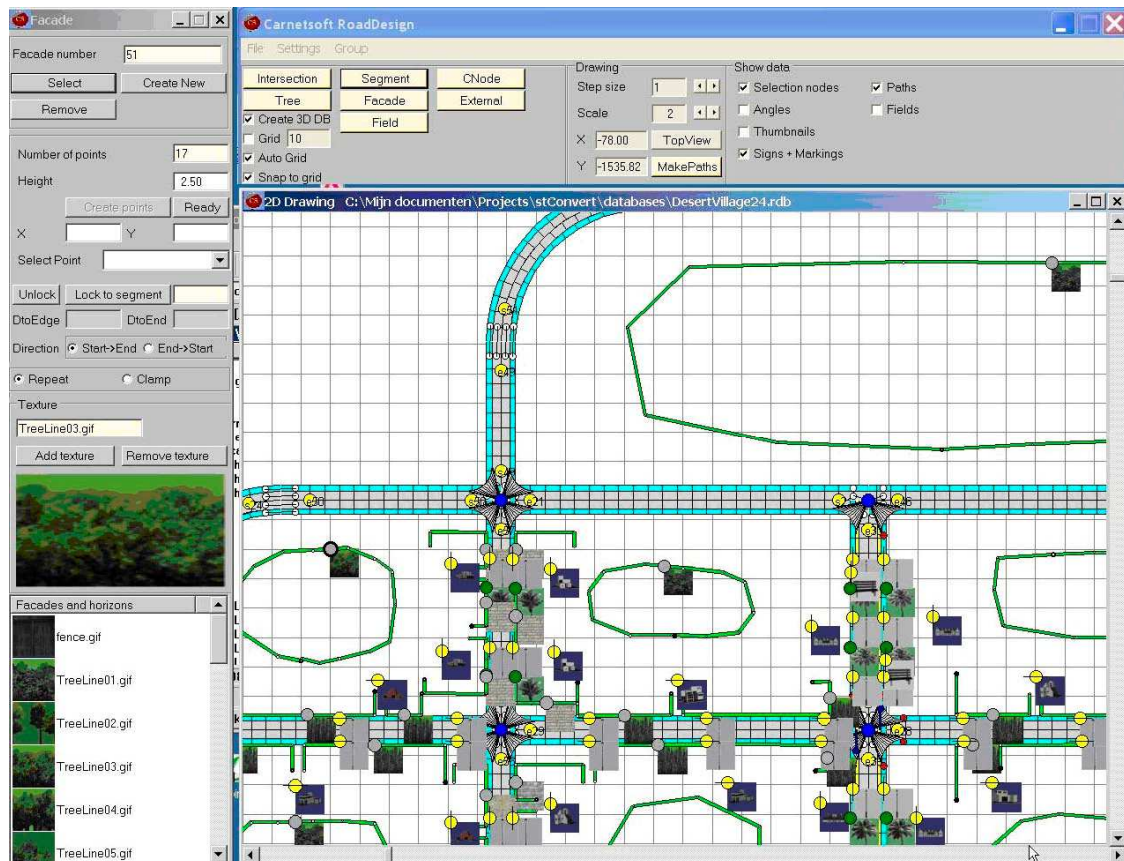
A facade can be positioned and after that, be attached to a segment. In this way it is easy to align facades at a precise position relative to the road:

- Position all points by clicking somewhere in the 2D drawing window. If all points in the façade has been defined, click the <Ready> button. Then optionally change the X and Y fields by selecting each points individually with the Select Point combobox, and then changing the X or Y values manually.
- After that, you may lock the facade to a segment by:
 - press <Lock to Segment>. Normally, you don't lock a façade to a segment.
 - then click on a yellow segment selection node
 - The distance from the right edge of the segment is now computed (DtoEdge) of all points together with the distance from the end of the segment (DtoEnd).
 - These values may be changed by the used by changing the DtoEdge and DtoEnd field values after selecting each point individually with the Select Point combobox

Distances are now with respect to the right road edge, measured from the start selection node to the end selection node (Start->End). If you press the End->Start radio button then the Facade is positioned on the other side of the road.

In the present version of RoadDesign, the locking to the segment information is not persistent. That means that if you save the network to a file, the next time the file is read the segment information is lost: only coordinate position information is retained. But then you still can lock the tree to a segment in order to position the façade more accurately.

In the example, a façade has been selected, by clicking on the GRAY hotspot: the selected façade can be seen at the bottom of the figure as 2 connecting green thick lines. The façade is of type 'Repeat', the height is 2.5 meters, and the texture is 'TreeLine03.gif'. It has 17 points. Since the number of connecting panels is $NrPoints-1$, you can see that there are 16 connected panels.



Facades are very versatile objects that can be used in a number of ways, for example:

- create horizon lines of trees, buildings etc. (Repeat façade, alpha channel)
- create closed areas with a texture of bushes, trees etc, that give the illusion of a forest (Repeat façade, alpha channel)
- create low walls or a row of bushes around a garden: in that case it is advised to work in a low scale, so that the grid is only 1 meter wide: you can the position all façade points accurately to align with the road segment. Alternatively, you may lock the façade to a segment to position it even more accurately (Repeat façade, no alpha channel)
- create rows of buildings: façade buildings are computationally inexpensive and it is an easy way to build a whole block of row houses. First make a 2-pointed façade (front façade) alongside the segment (click two points, Lock it to a segment, then click the Select Point combo box to select a point and change DtoEdge and DtoEnd, for both points individually). Then make the rear side of the building (rear façade) by defining a façade with 2 points, lock it to the same segment, make the DtoEnd values the same as the front façade, but make the DtoEdge values a few meters (say 7 meter) larger than those of the front façade. After that

make a new 2 pointed façade (side façade): Select the first point as the first point of the front façade and the second point as the first point of the rear façade. Then put a different texture of the facade (side of the house). Repeat the previous step for the second side of the building. In that case you then define 4 separate facades with different textures to represent the 4 sides of the building (Repeat façade, no alpha channel)

- etc.

To create a facade, do the following steps:

- Press the tab page 'Facades'
- Press button <Create New>
- click points that define panels with the mouse in the 2D Drawing form.
- if you are ready, click the 'Ready' button
- if required, lock the façade to a segment and alter the DtoEdge and DtoEnd of each individual point.
- All points may be selected and the coordinates altered by changing the X and Y values.
- select a texture by clicking on the required texture in the selection list.
- Press <Add texture>
- fill out the height.
- if the type should not be 'Repeat', then click 'Clamp'

A facade may be selected as follows:

- Press <Select>
- click in the gray hotspot of a facade.
- now the texture or the individual edge points may be changed, by selecting points with the 'Selected point' combobox, and changing the X and Y values.

9.4 External (3D) objects (externobject)

These are 3D objects with a fixed dimension and orientation. In the *.egg databases, they are referred to as "externobject". External objects can be models of buildings, of street furniture or any model that you may want to use in your databases. They can be either *.egg or *.bam objects: if they are included in the *.egg database files, they have to be *.egg files. If they are included in the *.ref files, they have to be *.bam files.

Position and angle can be specified as well as the specific type of object. All External objects to be used in the designer are specified in the resource file 'externalobjects.tex'. All objects themselves are located in the folder 'models\objects' within the DriveSim3 folder. Each object has its own sub-folder that must have the same name. So, if the ModelName FlatBuilding104 has been specified in 'externalobjects.tex', there must be a folder with the name 'FlatBuilding104' under the 'models\objects' folder. Within the folder 'FlatBuilding104', there must be a file 'FlatBuilding104.egg' and/or 'FlatBuilding104.bam', together with all associated textures.

It is important that you use low-polygon objects. All references to External objects are defined in the resource file 'externalobjects.tex'.

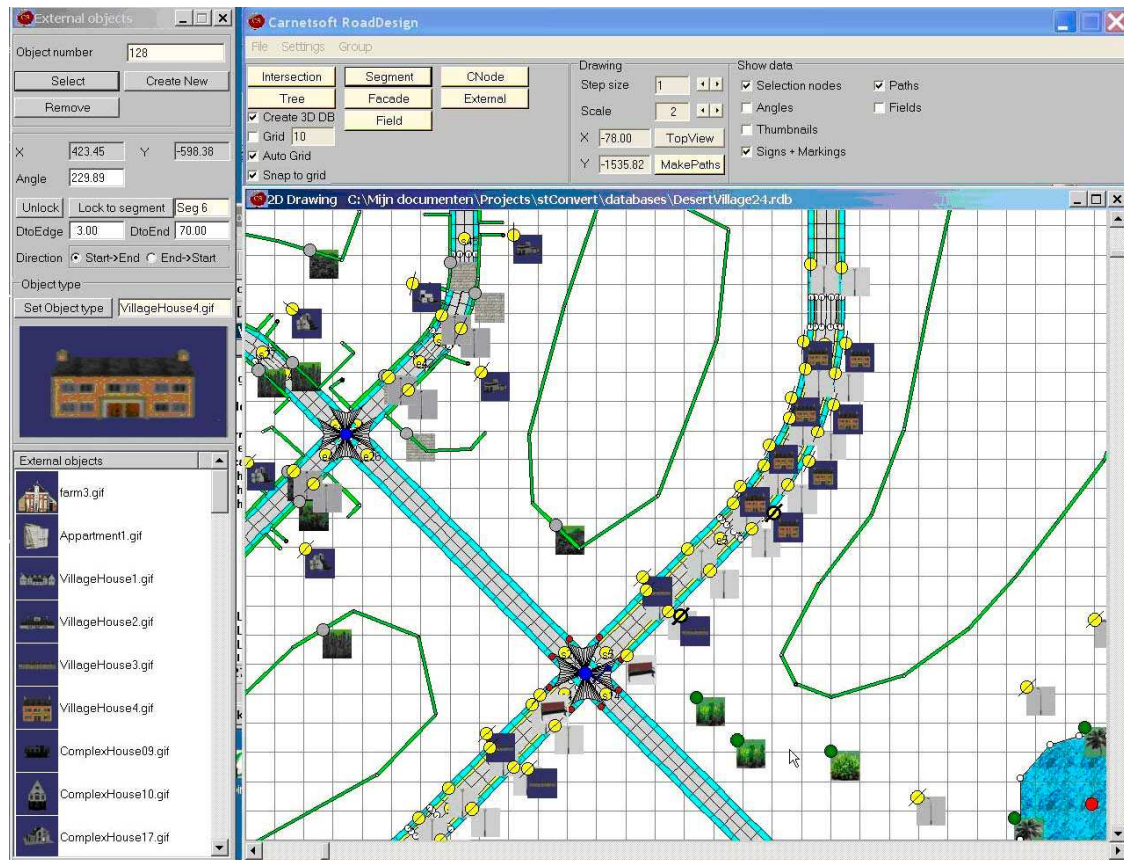
Each line contains a definition of a specific External object:

- **ThumbnailName** : the texture filename of the corresponding thumbnail in the GUI (in the folder "thumbnails")
- **ModelName** : the 3D modelname of the corresponding model used in the 3D model (in the folder "objects\<modelname>", for example "objects\FlatBuilding104")
- **Put in *.ref file** : a 1 or a 0: 1 means the object must be written to a *.ref file instead of the *.egg file. 0 means write to *.egg and not to *.ref file. In general, bigger (higher polygon) objects have to be stored in a *.ref file, while smaller repeating objects (for example street lights that are placed in high numbers beside the roads) can better be put into the *.egg files.
- **default height:** the default height of 0.0 means the object is put on ground level. Sometimes it must be raised a bit, for example if the object is a road marking that is put on the road.
- **default switch distance:** Switch distance of the object (can be changed in RoadDesign) after which the object is not rendered.

Lines in 'externalobjects.tex' then look like this:

```
OpenBarn.gif OpenBarn          1      0.0 500.0
AmericanBarn.gif AmericanBarn  1      0.0 500.0etc.
```

In the example, a 3D model has been selected, by clicking on the YELLOW hotspot: the selected External object can be seen at the bottom of the figure together with a thumbnail of the model.



As with Trees, External models can be positioned in a row, or as individual objects. To create a row of objects, do the following:

- 1) press the <Create New> button,
- 2) press the <Row> tab
- 3) press the <Select Segment> button
- 4) select a segment by clicking on a segment selection node.
- 5) Now a whole row of objects is created, as indicated by a row on yellow circles.
- 6) Always select the type of object and press <Set Object type>

For each segment more than one row of objects can be created. You can, for example, create a row of streetlights, and a row of single buildings.

The selection and modification of External objects is similar to Trees (see the explanation of Trees).

A single external 3D model can also be positioned and after that, be attached to a segment. In this way it is easy to align 3D objects, for example street lamps, houses etc., at a precise position relative to the road. Also the angle of the object is computed so that it faces the right side of the road.

- Position the external object by clicking somewhere in the 2D drawing window. Then optionally change the X and Y fields by changing the X or Y values manually.
- After that, you may lock the external 3D object to a segment by:
 - press <Lock to Segment>
 - then click on a yellow segment selection node

- The distance from the right edge of the segment is now computed (DtoEdge) of the center point together with the distance from the end of the segment (DtoEnd), based on the present X, Y coordinates. Also the angle of the object is computed so that the object is facing the right side of the road
- These values may be changed by the user by changing the DtoEdge and DtoEnd field values. The angle value is then changed automatically in case the segment is curved.

Distances are now with respect to the right road edge, measured from the start selection node to the end selection node (Start->End). If you press the End->Start radio button then the external 3D is positioned on the other side of the road. The angle value is now changed so that it faces the right side of the road, as seen from the End->Start direction.

By this procedure it is easy to add a whole row of, for example, street lights alongside a road at, say, 1 meters from the road edge and precisely 30 meters apart (for example first street light at 30 meters DtoEnd, second street light at 60 meters DtoEnd etc.)

The locking to the segment information is persistent. That means that if you save the network to a file, the next time the file is read the segment information is regained: segment information is attached to the object.

To create a single External object, perform the following steps:

- Press the tab page 'External objects'
- Press button <Create New>
- Position the object by the methods described earlier.
- select an object type by clicking on the required object in the selection list.
- Press <Set Object Type>
- fill out the angle (the angle with respect to the world in degrees) if needed. Not necessary if the object has been aligned to a segment.

An External object may be selected as follows:

- Press <Select>
 - click in YELLOW hotspot of an External object.
- now the object, angle, or position may be changed

10 Paths

After a network of roads has been build, the network can be used in a traffic simulation. In the simulation, paths play an important role. A path is a logical connection between two nodes (intersections and connection nodes). A vehicle must always be on a path. Each path is unique and each path in one direction has a counterpath in the opposite direction. Each path number is an even number and the counterpath is the next odd number. For example, P10 has P11 as its counterpath. Path numbers are created automatically by RoadDesign, if the program is saved as a .rdb and .net file. But these .net files are filled with numbers and not very accessible to the user. Although there are also files saved with the extension .path that contain more accessible path information, it still is an arduous task to extract the path numbers from the files. In order to make this task easier, there is a button <MakePaths> in the upper panel. If you press this button, the program computes path numbers, and shows these on the 2D Drawing window. You can then zoom in on smaller portions of the network and make screenshots (after toggling off information, of, for example, fields, signs and markings).