

# Auto-Routing - The Rotweiler Experience

(for what it's worth...)

**Background:** I had months of experimentation and severe frustration trying to create routable maps using GPSMapEdit and MapRoute/cGPSMapper (see “Sources” at the end of this guide for more information of these two great programs). I DID experiment with ARCVIEW (thanks for the demo, ESRI – works great but MUCH too expensive, and very difficult to master).

This “guide” is a summary of what I learned. Hopefully some of it will be useful to you. Special thanks to Kok Chi (Singapore) who greatly assisted the learning process. The process MAY look rather complex, but after one or two trials, it will become simple and clear (clearer?) Of course, this guide is aimed at maps for Garmin GPS devices only. All maps built by the author have been tested on Garmin's iQue 3600 and eTrex Vista Cx.

Finally, all brand or trade names used are the property of the various companies who own them.

## Step 1 - Create map using GPSMapEdit (Polish format #.mp)

*Note: GPSMapEdit is **definitely** the program of choice for map making and editing (unless you have tons of money, even more patience, can stand a LONG learning curve and decide to go the ARCVIEW method...). All previous issues with routing (the transfer of information between GPSMapEdit and cGPSMapper) have dramatically improved with the most recent releases by both authors. The routing is now (mostly) optimized and several of the glitches regarding one-way streets and speed limit data now works (almost) perfectly. I **strongly** recommend downloading the most recent version of GPSMapEdit **first before going any further!***

### BEFORE STARTING YOUR MAP

The routing function of a GPS map occurs only at Level 24, so while many map builders prefer multi-level maps (multi-level maps they LOOK prettier inside MapSource), you CAN build multi-level maps from scratch, or have MapRoute do it for you by defining the levels you want to have created and the contents of each level, by editing the “Dictionary” section. Especially for beginners, I strongly recommend using only two levels - one working (24) and one empty level (I use 18). This keeps the confusion between levels to a minimum and you can always bring in additional levels during the compiling stage (but more about that later).

I will assume you have already started a map or have an existing “.mp” format map to work on. Click on “Map Properties” and then the “cGPSMapper” tab. Ensure that “POI Index”, “Enable MG” and “Enable automatic routing”, “ are checked. To be perfectly safe, also select “Save objects as [RGNx0] sections (old format to ensure maximum compatibility with MapRoute) although the latest versions of MapRoute don't seem to care much which format you use.

### COMMON ERRORS IN BUILDING ROUTING MAPS (in GPSMapEdit)

**Do your roads actually connect?** When building the map, ensure that ALL roadways you intend to make routable in fact JOIN each other. The most common error are nodes that do not physically connect, making routing impossible. One method of helping to achieve this is to set the GPSMapEdit options to “Stick to neighbors” and carefully check each link to ensure the connection has been made.

Right click the overlapping node where a connection is to take place and click on “Connect to nearest node”. The selected node will change color (yellow or green are good, red CAN be good but MAY not be – it CAN be an indication that the node is ‘complex’). Do that for ALL connections within your map. Double check ALL red-colored points and try various connection options to turn the point anything except red.

**Routing across tiles.** If you are connecting several maps, or have split a larger map into smaller pieces (tiles), make sure that the nodes where the connections are to take place ACTUALLY will connect (by right clicking each connecting node, select Node Properties, and ensure that “This node is routing graph node” is checked. That’s all (except if you intent to provide turn restrictions at a certain intersection, in which case you can add those to each connecting node as well). The rest will be completed automatically.

The current (most recent versions of GPSMapEdit and cGPSMapper do a splendid job of linking maps – **BUT... there is one very important caution – irregular-shaped map tiles (individual maps) MAY not link correctly.** While I am not certain of the reason, this may be a function of the ‘backgrounds’ that cGPSMapper automatically adds to each tile during processing.

Once you have completed these steps, GPSMapEdit makes the actual routing **really simple**. Just click ‘Tools’ – ‘Generate Routing Graph’ – ‘Using coinciding nodes of Polylines’. That’s all.

**Provide SPECIFIC routing information for each road.** Each component of a roadway (each separate segment of a “line” or “polyline”) can be edited to provide specific routing options for it. Included are the type of roadway, the average vehicular speed on that segment, whether or not the segment is a one-way road, restrictions on its use by certain classes of vehicles, etc. You must ensure that each segment (separate roadway ‘piece’) has its properties changed to reflect real-life travel – and also ensure that ALL of the segments of the same road have the same routing properties well (unless, of course, you intentionally want different sections to have different travel restrictions/speeds, etc..

To create a one-way road, right click the line segment using the “Select Objects Tool in GPSMapEdit. Right click, go to “Object Properties”, select the “Properties” tab, and click-on the “Has Direction” box. You should also check that the “Direction Arrow” showing on a line after it has been made direction if going the right way. If not, right click the segment again and select “Reverse Polyline”.

**‘Data Babel’.** One of the most often-made errors in building routable maps is related to levels. (Remember the suggestion to use only ONE data level??) You **MUST** check the .mp file to ensure that none of the routable roadways have data in other levels than the base level (24, also seen in the file as “Data0=”. If you haven’t done that during the basic map-making process, you can now.

Open the file with your text editor and search for any instances of Data1=, Data2=, and so on. All routable roadways (lines) must be “Data0=...” **only**.

**Correct map Identification.** Finally identify the name correctly. Click on “Map Properties” [shortcut Alt Enter] and enter an ID number. The number can be anything you want but EACH map must have a different ID number using the Garmin 8 digit numerical system. I recommend numbers greater than 1xxxxxxx. Enter a Map Name and save your work. The naming process is simple – always name your file with the same 8 number ID you used in the “Map Properties ID” - it will save confusion later on.

## Step 2 – The “Shapes” Process

As you will surely know if you have experimented or used it before, cGPSMapper is a command-line program. To simplify the following steps, it is advisable to first copy or move the cGPSMapper folder to a new location. I use C:\gpsmap. Keep the names simple. Then copy your .mp files from wherever your editing has taken place, to a new folder under your new folder for cGPSMapper – again, keep it simple. For my Mexican .mp routing files, I used C:\gpsmap\mex . You will see why shortly. For the remainder of this document, I will assume that you have followed this advice and use those folder names for all command-line processes.

Open a command line window (aka a DOS window for us old-timers. Go to C:\gpsmap\[*name of folder where the new .mp files are located*]. In our example case, go to C:\gpsmap\mex . Our .mp file, for the sake of the following examples, will be “10101010.mp”

Create the “shape” files. Command line input:

```
..\cgpsmapper shp 10101010.mp
```

Assuming that there are no errors (if there are, cGPSMapper will provide – in most case – the line number where the error is located. Notepad++ is really handy for the debugging process since it number all lines). This will produce a number of files with various extensions including the ESRI-format definition files required to do routing calculations. As noted earlier, this is the time we **can** check the one-way roads to see if the directional information has been passed on to cGPSMapper.

If you wish to check these, open the line.dbf file with Excel and go to the “one way column. (*Now you will know why we moved all of the one-way roads to the beginning of the file...*). If any of the “one-way” cells for your one-way roads do not have a 1, change the “0” to a “1”. Close Excel and save (*in the original file format, NOT as an excel file!*).

## Step 3 – The “MapRoute” Process

Maproute.exe will be located in your “gpsmap” (original name cgpsmapper, remember?) folder. If not, download it from:

<http://www.cgpsmapper.com/tool.htm>

It will also contain a file called “maproute.ini” (also called map.ini in other area’s of the cGPSMapper site). Copy those files to your working map file (in our example, c:\gpsmap\mex). Then create the routing files. Command line input:

```
..\maproute maproute.ini ##
```

What the heck is ## you ask? It is actually a complex input to tell MapRoute exactly what to process. Run the command line input “maproute” and see the options. I have had advice from many map-builders on which numerical input value to use, and have tried pretty well all combinations. I recommend using the value “39” since it combines the “Merge and Correct wrong elements”; “Polish Format output, and “No generalization”. It seems to work well for me. Some day there will be a GUI for it and that would resolve a lot of real or apprehended issues.

The odd map will NOT process using “39” as an input value. Usually, this is a matter of “too much detail”, but not always. In that case, try “7” or as a last ditch effort, “3”.

```
..\maproute maproute.ini 39
```

[NOTE: As of this writing (May, 2007) the author of cGPSMapper and MapRoute (<http://cgpsmapper.com>) has a new version of MapRoute (April 2007). This new version of MapRoute and the latest version of the “.ini” file substantially improve the routing process and provide vastly improved error reporting as well.]

This will produce the “processed” line and segments files, the routing input into your final map.

Assuming it has in fact produced the proceed files without reporting errors, (creating the processed\_line.dbf, processed\_line.shp, processed\_line.shx, and processed\_segments.dbf files), at this point you should delete the line.dbf, line.shp, and line.shx as well as all of the marine.xxx files. They are not required for routing. And you can move the original .mp file (in our example, 10101010.mp) to your “map working” area. It has served its purpose.

## **Step 2 – Building the “build.mp” File**

### **General**

This is the most misunderstood aspect of routable mapping. The build.mp file will take the place of the original number .mp file to provide the information cGPSMapper requires to process the various files created in the “shapes” and “MapRoute” steps.

In order to show this complex process as simply as possible, the build.mp file really has only two major components – a descriptive section and a “Dictionary”. The “Dictionary” tells the cGPSMapper program exactly **what** to process and how to process it.

First, the “descriptive” portion. I will use one of the Mexican map examples again.

[IMG ID]





```
;nodeID3=  
;roadID1=  
;roadID2=  
[END-DEFINITIONS]
```

```
[SHP]  
Name=processed_line  
Type=64  
LabelField=label  
Label2Field=descr  
TypeField=type  
;DirField=oneway
```

```
CityName=city  
RegionName=region  
CountryName=country  
houseNumber=house  
streetDesc=descr  
phoneNumber=phone  
zip=zip
```

```
roadId=roadid  
speedType=speed  
RoadClass=roadclass  
OneWay=oneway  
Toll=toll  
;VEHICLEC=not_car
```

```
Level=0  
EndLevel=3  
[END-SHP]
```

```
[SHP]  
Name=poi  
Type=RGN10  
labelField=label  
typeField=type  
streetDesc=descr  
houseNumber=house  
phoneNumber=phone  
zip=zip  
highway=highway  
cityName=city  
regionName=region  
countryName=country
```

```
Level=0  
EndLevel=2  
[END]
```

```
[SHP]  
Name=point  
Type=RGN20  
labelField=label  
typeField=type  
regionName=region  
countryName=country
```

```
Level=0  
EndLevel=3  
[END]
```

```
[SHP]
Name=polygon
Type=RGN80
labelField=label
typeField=type

Level=0
EndLevel=3
[END]
```

These are discussed in much greater detail in the cGPSMapper manual (a **must read**).

## Customizing the Build.mp File

Once you have a “build.mp” file that is to your liking, two changes are required **each** time you process a file or send that file to the MapCenter for processing (creating the actual Garmin-compatible “.img” file). First, the “Header” or [IMG ID] section must be changed for the *specific* map you are having processed. Change the following:

```
ID= ; put in the unique 8-digit number that you used in GPSMapEdit
Name=A Map Name
```

A sample build file is available on my site at <http://rotweilermaps.com/files/build.mp>

## Submitting the map to the MapCenter

If you are submitting the map to the MapCenter for processing, you need to zip up and send the build.mp file you have customized for that specific map (NOT the numbered.mp file!), all of the poi, point, and polygon files (each with a .dbf, .shp and .shx file), as well as all of the files starting with “processed\_” (processed\_line, processed\_mapping, and processed\_segments files).

If you are fortunately enough to have purchased your own copy of Stanislaw Kozicki’s “*cGPSmapper*”, (and I strongly recommend you do!) then the process is only:

```
..\cgpsmapper ac build.mp
```

This will produce the image(s) ready for uploading to your Garmin GPS – whether via MapSource or SendMap or using MalSing’s M3 installer.

## Conclusion:

The process is lengthy – but if you **DO** follow the instructions, it is actually quite simple. Yes you say, for you... Well, I went through the entire learning process without a guide and managed to master it (I think...). And so can you.

Rotweiler  
<http://rotweilermaps.com>

(I would appreciate any criticisms or suggestions for improvements for this guide. Please email me at **rotweiler00[at]gmail.com** (replacing the [at], of course...).

## **SOURCES**

CGPSMapper            <http://cgpsmapper.com>

The MapCenter:      <http://mapcenter.cgpsmapper.com/>

GPSMapEdit:         <http://geopainting.com/en/>

MalSing:             <http://www.malsingmaps.com/forum/index.php>

NotePad++:          <http://notepad-plus.sourceforge.net/uk/site.htm>

**VERY large greetings and thanks** to Stanislaw Kozicki (cGPSMapper) and Konstantin Galichsky (GPSMapEdit)